Nápověda Kaloka

Jak Kaloko nastavit, projít aplikaci s agentem, zkontrolovat, co postavil, a z výsledku vést changelog a dokumentaci. Obrázky pocházejí ze schválených běhů samotného Kaloka, takže ukazují obrazovky tak, jak vypadají dnes. I tahle nápověda je živá dokumentace publikovaná v Kaloku.

Tutoriály

- První běh za deset minut: instalace, přihlášení, průchod flow, sdílení a kontrola
- Navrhněte flow se svým agentem: živé HTML kroky, verze a reference designu

Kontrola a schválení

- Projít běh: obrazovky běhu na plachtě jednu po druhé
- Schválit nebo vrátit běh: schválit kroky, které sedí, a vrátit ty ostatní
- Kontrolovat jen to, co se změnilo: režim kontroly, jeho klávesy a převzatá schválení
- Co je nového: novinky s obrázky před a po
- Předat podepsaný záznam o přijetí: doklad, kdo co převzal, balíček dokladů a jak je ověřit

Průchod a kontroly

- Napsat scénář: flow, jeho kroky a co má každý krok ukázat
- Přidat kritéria a packy kontrol: automatické kontroly, otázky pro evaluátor, packy
- Kontrola kondice aplikace: SEO, přístupnost a výkon mimo přijetí, převzetí kontroly, upozornění
- Projít víc prohlížečů a zařízení: výchozí velikosti, WebKit, Firefox, telefony, tablety, tmavý režim
- Projít aplikace pro Android, iOS a desktop: emulátory, simulátory, Electron, macOS a Windows
- Natočit video průchodu: video a průběh každého kroku, jedno video celého běhu
- Skrýt osobní údaje ve snímcích: masky, automatické vyhledávání, smyšlené údaje, kontrola před sdílením
- Spouštět Kaloko v CI a komentovat pull requesty: průchod každého preview nasazení
- Porovnat s referenčním během a ignorovat, co se mění pořád: rozdíly bez hodin a karuselů
- Kontrolovat stránky podle plánu (hostované běhy): prohlížeče Kaloka otevřou vaše stránky každý den nebo týden
- Každý týden hlídat SEO a odpovědní vyhledávače: stacky z jazykových sitemap, packy seo, aeo a locale, kontroly celého webu, týdenní workflow na GitHubu, výsledky pro dashboard

Tým a přístup

- Pozvat lidi a hosty: členové, hosté v jednotlivých projektech, čtenáři z vaší domény
- Role a API tokeny: kdo smí co, tokeny pro agenty a CI
- Nastavit firemní přihlášení: přihlašovací domény, dvoufázové ověření, SAML, OIDC a SCIM
- Připojit Jiru, Linear, Slack a Teams: úkoly z kroků a kanály po projektech

Changelog a dokumentace

- Publikovat changelog s obrázky: připravit záznam, zapsat release, nasdílet ho
- Publikovat živou dokumentaci a hlídat zastaralé části: návody, jejichž obrázky drží krok s aplikací
- Exportovat dokumentaci a changelog: Markdown, Docusaurus, VitePress, MkDocs, HTML, PDF nebo JSON

Reference

• Příkazy CLI: všechny příkazy kaloko

• Soubor scénáře: všechny klíče scénáře

• Kritéria, kontroly a packy: všechny typy kontrol a co který pack přidá

• Kontroly kondice: všechny packy a jejich kontroly, proč na každé záleží, její závažnost a kde platí

• Konfigurační soubor: kaloko.config.yml

• Nástroje MCP: co agenti zvládnou přes MCP

• Shrnutí pro posouzení dodavatele: jedna stránka pro posouzení dodavatele

• Webhooky: události, jejich obsah a ověření podpisu

• API kondice pro dashboardy: kondice sečtená po jazycích, krocích a kontrolách a co se změnilo od minulého běhu

Vysvětlení

• Jak Kaloko odpovídá vašim aplikacím: organizace, produkty, projekty, moduly, prostředí a verze

• Jak vznikají verdikty: kontroly, evaluátor a lidé

• Bezpečnost a vaše data: co opustí váš počítač a kdo to uvidí

• Doklady o kvalitě pro váš systém kvality: co záznam dokládá, podpisy, dohledatelnost, jak zapadá do ISO 9001, ISO 13485, IEC 62304, ISO 27001 a GxP a známé mezery

Jak Kaloko odpovídá vašim aplikacím

Jeden repozitář je málokdy jedna aplikace. V monorepu bývá aplikací několik, jedna aplikace se táhne přes víc repozitářů, agentura má projekt pro každého klienta a každý tým pojmenovává prostředí jinak. Kaloko proto pracuje s několika pojmy, které jednou přiřadíte ke své situaci.

Pojmy

| Pojem | Co to je | Obvyklé přiřazení | |---|---|---| | Organizace | účet vaší firmy nebo agentury | jedna na firmu | | Produkt | to, co se vydává a verzuje jako celek, s vlastním changelogem a dokumentací | e?shop, administrace, mobilní aplikace, balíček | | Projekt | jednotka práce a přístupu: kdo ho vidí a kdo ho kontroluje | klient, tým, modul, celá aplikace | | Modul | část produktu s vlastní sekcí v dokumentaci a changelogu | košík, účet, fakturace | | Scénář a krok | flow a jeho obrazovky | patří k projektu; může uvést produkt a modul | | Prostředí | kde flow běží | local, preview, staging, produkce, production-cz | | Verze | vydaná verze produktu | z git tagů, package.json, vašeho nástroje na release nebo z kaloko release |

Produkty jsou volitelné

Bez produktů je produktem projekt a jeho changelog je changelog projektu. Tak začíná většina týmů. Až budete potřebovat víc, vypište produkty pod products: v kaloko.config.yml a scénáře uvedou svůj přes product: a module:. Nic se nemusí přesouvat.

Projekt může pokrývat několik produktů a na jednom produktu může pracovat víc projektů. Dokumentace a záznamy v changelogu odkazují na kroky, takže produkt a modul znají ze scénáře.

Verze patří produktům

Prostředí jen říká, kde běh proběhl. Běh je svázaný s verzí produktu a dokumentace pro verzi 2.4 ukazuje přijaté běhy verze 2.4. Obrázky pocházejí z prostředí, které produkt určí jako referenční (reference_env), třeba staging u interního nástroje nebo production-cz u portálu jen pro Česko.

Čtyři obvyklá uspořádání

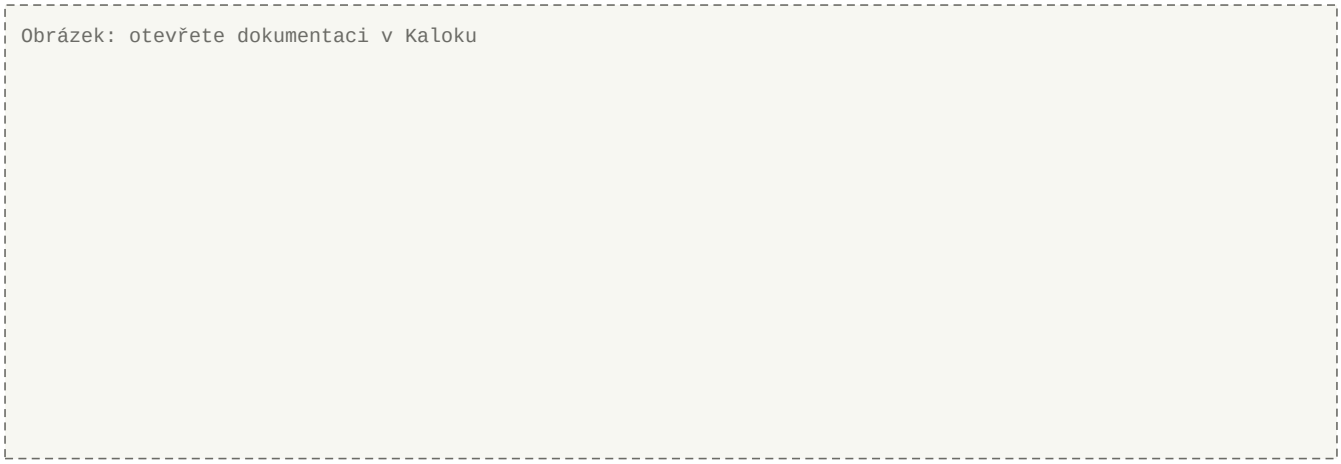
1. Jedna aplikace, jeden tým. Žádné produkty; projekt je aplikace a má jeden changelog. To je výchozí stav.

2. Monorepo s několika aplikacemi. Produkt pro každou aplikaci; projekty podle týmů nebo aplikací.

3. Agentura. Projekt pro každého klienta a produkt pro každou dodávku. Hosté od klienta čtou jen changelog a dokumentaci svého produktu.

4. Velká aplikace s moduly. Jeden produkt s moduly podle týmů. Každý tým se stará o dokumentaci svého modulu a release má jeden společný changelog se sekcí pro každý modul.

kaloko init navrhne produkt pro každou aplikaci, kterou najde: pro každou složku v apps/ s vlastním package.json a pro složky workspaces (npm, Yarn nebo pnpm) a packages/, které se spouštějí jako aplikace (mají skript dev, start nebo serve). Sdílené knihovny vynechá. kaloko products sync je pošle do Kaloka ještě před prvním vydáním a Produkty v menu aplikace ukážou každý produkt, který čtete.



Stránka Produkty · cs · desktop

Odebrání produktu

Produkt, který už nevydáváte, se archivuje, nemaže. Spusťte kaloko products remove <product>, nebo otevřete jeho novinky, sjeďte k nastavení na konci a zvolte Archivovat produkt. Smí to jen admin projektů produktu (u produktu celé organizace admin organizace).

Archivovaný produkt zmizí z Produktů a jeho novinky, dokumentace i veřejná stránka hlásí, že neexistují. Vydání a dokumentace zůstanou uložené. kaloko products sync ho znovu nezaloží a napíše, že je archivovaný, takže ho smažte i z products: v kaloko.config.yml. Když si to rozmyslíte, kaloko products restore <product> ho vrátí tak, jak byl. kaloko products list archivované produkty vyjmenuje.

Každá část funguje sama

Můžete používat jen changelog, jen dokumentaci, jen dokumentaci jednoho modulu, jen obrázky pro web s dokumentací, který už máte (kaloko export), nebo jen e?mail "Co je nového“. Nic nevyžaduje celou sadu.

Jak vznikají verdikty

Každé kritérium na plachtě skončí jako splněno, částečně, nesplněno, nebo čeká na člověka. Rozhodují o tom tři druhy soudců v pevném pořadí: kontroly, které počítají, AI evaluátor na to, co chce úsudek, a lidé, kteří mají poslední slovo.

Nejdřív kontroly, které počítají

Co jde změřit, se zkontroluje přímo na živé stránce při zachycení kroku: je tam text, pole má popisek, adresa po kroku sedí, barva odpovídá tokenu z designu, stránka se načte v časovém limitu. Ke každému výsledku zůstane důkaz, třeba prvek, který odpovídal, nebo naměřená hodnota, takže vidíte, proč kontrola prošla nebo neprošla. Tyto kontroly běží bez AI a nikdy nespotřebují žádný dotaz na evaluátor.

Evaluátor na úsudek

Kritéria jako "chybová hláška říká, co má člověk udělat“ jdou sémantickému evaluátoru. Výchozí je JEV od TypeSafe. Čte zredukovanou osnovu stránky, ne screenshot, a vrátí skóre mezi 0 a 1. Nad prahem pro splnění (výchozí 0,85) je kritérium splněné, pod prahem pro nesplnění (0,15) nesplněné a mezi nimi čeká, až rozhodne člověk. Prahy nastavíte v kaloko.config.yml.

Kritéria označená vision: true a odpovědi v nejistém pásmu dostanou druhý pohled od evaluátoru, který vidí screenshot. Ten navíc jednou větou řekne, co rozhodlo; na plachtě ji najdete jako "Proč“.

Evaluátor běží s vaším vlastním klíčem v každém plánu, nebo přes Kaloko od plánu Team v rámci měsíčního počtu dotazů. Odpovědi se ukládají, takže nezměněnou obrazovku vyhodnotíte znovu bez nového dotazu. Bez klíče i hostované evaluace zůstanou tato kritéria "nehodnoceno“, dokud nerozhodne kontrolující. kaloko calibrate porovná evaluátor s verdikty vašich kontrolujících a navrhne prahy.

Na webu s tisíci stránek můžete omezit, kolik placených dotazů smí jedno vyhodnocení položit (evaluate.budget.calls). Nejdřív přijdou na řadu běžné kroky, pak stránky ze stacků, a to nejprve ty, které neprošly kontrolou nebo se kontrolovaly nejdéle před tím. Co se nevejde, zůstane "nehodnoceno“ a počká na další vyhodnocení. Jako první může odpovídat i váš vlastní model, třeba v Ollamě nebo v LM Studiu. Nic to nestojí a JEV dostane jen odpovědi, ve kterých si model nebyl jistý. Viz konfigurační soubor.

Třídění: o jaký problém jde (experiment)

Verdikty odpovídají na vaše kritéria. Třídění odpovídá na jinou otázku: co na obrazovce vypadá špatně, ať se na to některé kritérium ptá, nebo ne. Jakmile ho admin zapne v Nastavení -> Zabezpečení -> Experimenty, kaloko evaluate ukáže každou obrazovku a pár snímků z její nahrávky modelu Clef na Cloudflaru. Clef pojmenuje druh problému: překrývající se prvky, oříznutý text, překladový klíč místo textu, zástupný text, chybu, nekonečný spinner, poskakující rozvržení. Přetékání obsahu přidá sama stránka, která ho změřila.

Nálezy se ukážou na plachtě v části "Možné problémy“ a verdikt nikdy nezmění. Kontrolující jedním kliknutím převede správný nález na běžný komentář, označí ho jako správný, nebo řekne, že nejde o problém. Z těchto odpovědí poznáme, jestli vám třídění šetří čas; v Nastavení uvidíte, kolik jich zatím bylo užitečných. Každá obrazovka se počítá jako jedna otázka z měsíčního limitu.

Kondice: jen radí, stejně jako třídění

Kontroly z packů (SEO, přístupnost, výkon, použitelnost, konzole, chybové stránky) odpovídají na to, jestli je aplikace v dobrém stavu, ne jestli práce dělá, co chtělo zadání. Vyhodnocují se stejně, ale tvoří kondici běhu a nikdy nemění verdikt kroku ani běhu, frontu ke kontrole, převzatá schválení ani záznam o přijetí. Když má některá z nich rozhodovat, tým ji převezme jako kritérium; v přijetí může zůstat i celý pack. Víc v návodu Kontrola kondice aplikace.

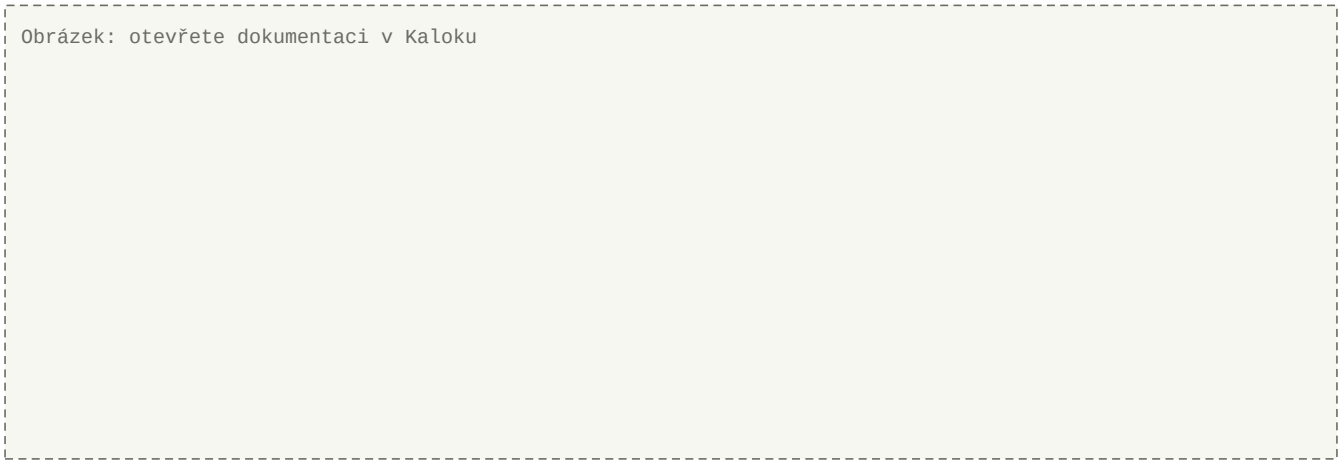
Rozhodují lidé

Kontrolující krok schválí, vrátí s poznámkou, nebo rozhodne kritérium, kterým si evaluátor nebyl jistý. Nakonec celý běh přijme nebo vrátí. Každé rozhodnutí nese jméno.

• Povinní schvalovatelé (od plánu Team): běh je přijatý, až ho určení lidé schválí.

• Schválení zůstává. Když se nasdílí nový běh, krok, jehož screenshoty a kritéria se od posledního přijatého běhu nezměnily a jehož kontroly pořád procházejí, si ponechá schválení, které mu dal člověk, i s tím, kdo a kdy. V režimu kontroly čeká jen zbytek. Převzaté schválení může kdokoli zrušit. Převzaté schválení nikdy běh nepřijme a nepočítá se za povinné schvalovatele.

• Agenti za lidi neschvalují. Rozhodnutí agenta nebo tokenu jsou označená jako jejich a za rozhodnutí člověka se nepočítají.



Režim kontroly · cs · desktop

Proč v tomhle pořadí

Počítání je levné a přesné, proto jde první. Úsudek je drahý a občas se mýlí, proto řeší jen to, co spočítat nejde, a řekne, když si není jistý. Lidé vidí u každého verdiktu důkaz i důvod a za přijetí se zapíše jejich rozhodnutí.

Doklady o kvalitě pro váš systém kvality

Stránka je pro lidi z kvality, validace a vydávání, jejichž tým pracuje v systému řízení kvality: podle ISO 9001, ISO 13485, IEC 62304, ISO 27001, nebo podle GxP ve farmacii a zdravotnických prostředcích, kde jsou pravidla nejpřísnější. Kaloko z vás neudělá organizaci ve shodě ani certifikovanou a samo podle žádné z těchto norem certifikované není. Dává vám doklady: záznam, který se dá ověřit, data, která vaše vlastní dokumenty potřebují, a poctivý přehled toho, co dělá a co ne, aby ho váš QMS mohl použít a vy jste si to použití mohli validovat po svém.

K čemu tu Kaloko slouží

Kaloko prochází obrazovky webové nebo mobilní aplikace podle kritérií, která jste napsali, zaznamená, co vidělo (snímky, stav stránky, výsledky), a sbírá, co o tom rozhodli lidé: schválení, vrácení, komentáře a přijetí běhu. Z toho vznikne záznam podepsaný službou, který si kdokoli může později ověřit.

Není to váš eQMS. Nespravuje řízené dokumenty, odchylky, CAPA, školení ani žádosti o změnu a sám nic neuvolňuje. Hodnotitelé s AI mohou výsledek navrhnout, jejich verdikty jsou ale jen doporučení: rozhodují lidé.

Co Kaloko zaznamená a na co se můžete spolehnout

Záznam obsahuje:

• běh, scénář a prostředí, s verzí scénáře, SHA-256 souboru scénáře tak, jak se procházel, a základní URL

• každé kritérium tak, jak se procházelo (jeho tehdejší text, ne dnešní), výsledek v každém jazyce a velikosti obrazovky a kdo ho hodnotil: člověk, deterministická kontrola, nebo model AI (jmenovitě, jako doporučení, pokud nerozhodl člověk)

• odkazy na požadavky (refs:) u scénáře, jeho kroků a kritérií

• záznamy, ke kterým běh patří, a proč se šlo, pokud je běh uvádí (viz níže)

• SHA-256 každého snímku a manifestu běhu

• rozhodnutí o krocích a o běhu, se soubory přiloženými k rozhodnutí o kroku (název, velikost, SHA-256, kdo a kdy) a s dřívějšími rozhodnutími o kroku; schválení převzatá z dřívějších běhů i s tím, odkud pocházejí, a schválení tokenů a agentů odděleně od lidských

• elektronické podpisy a důvody změn, pokud podepisujete v Kaloku

• verze CLI, Node.js, operačního systému, prohlížečů a služby

• výsledky mimo akceptační kritéria (například kontroly kondice), označené jako doplňkové

Balíček dokladů k tomu přidá matici dohledatelnosti, auditní stopu běhu, snímky a soubory přiložené k rozhodnutím o krocích (decisions/<krok>/), s manifestem SHA-256 všech souborů podepsaným stejným klíčem.

Na co se můžete spolehnout:

• Ověříte si to bez nás. kaloko signoff --verify ověří záznam i balíček proti veřejnému klíči na /.well-known/kaloko-signoff.json, offline a bez účtu. Změněný bajt ověřením neprojde.

• Staré záznamy jdou ověřit dál. Vyřazené klíče zůstávají zveřejněné.

• Formáty se nemění pod rukama. qawalk.evidence.v1 a kaloko.signoff.v1 v rámci verze jen přibírají pole; nekompatibilní změna dostane novou verzi a starou půjde dál číst.

• Lidé a stroje jsou oddělení. Výsledek AI, o kterém nikdo nerozhodl, je označený jako doporučení, a tokeny ani agenti se nikdy nepočítají jako lidé a nepodepisují.

Dvě cesty

Kaloko jako záznam o provedení, podpis ve vašem eQMS. Je to výchozí stav a nic se nenastavuje. Kaloko zaznamená, co se prošlo a kdo co rozhodl; vaše vlastní nástroje si doklady stáhnou (kaloko evidence, balíček dokladů nebo webhook evidence.ready), udělají z nich vaše dokumenty a vaši lidé je podepíšou ve vašem

eQMS. Každý doklad to říká přímo: acceptance je none, accepted nebo returned a signing je none, takže generátor může napsat "provedeno v Kaloku, schváleno v našem eQMS“, aniž by hádal. Váš systém pak pokrývá elektronický podpis, jeho význam, opětovné ověření, vazbu mezi podepsaným dokumentem a doklady (uložte do něj SHA-256 záznamu) a jak dlouho balíček uchováte.

Podpis přímo v Kaloku. V plánech Business a Enterprise může admin zapnout elektronické podpisy v Nastavení -> Zabezpečení, pro všechny projekty nebo jen pro některé. Kdo pak přijímá běh (a pokud chcete, i schvaluje krok), vybere, co podpis znamená, z vlastního seznamu vaší organizace, znovu potvrdí druhý krok ověření a záznam uvede jeho jméno, e-mail, čas v UTC i v jeho časovém pásmu a význam podpisu. Podepisující vidí významy ve svém jazyce (Reviewed, Approved a Verified jsou přeložené, vlastním připíšete český název za svislou čáru: QA release | Uvolnění QA) a záznam uchová název, který viděl, jeho id a jazyk. Podpis pokrývá i samotné rozhodnutí: poznámku a SHA-256 souborů, které k němu někdo přiložil. Změna podepsaného rozhodnutí chce důvod; podpis v záznamu zůstane, označený jako odvolaný, i s tím, kdo, kdy a proč. Tokeny a agenti nepodepisují nikdy. Když jsou podpisy vypnuté, nic se nemění: lidé přijímají a schvalují jako dosud.

Dohledatelnost

Identifikátory požadavků napište do scénáře tak, jak je vede váš QMS:

```
refs: [URS-12]
nodes:
- id: release-form
refs: [FS-3.4]
criteria:
- id: AC1
text: Před uvolněním je vidět číslo šarže
refs: [URS-12.1, RA-7]
```

Kaloko je nijak nevykládá. Matice dohledatelnosti má řádek pro každý požadavek, kritérium, krok, jazyk a velikost obrazovky, se snímkem a jeho SHA-256, výsledkem, tím, kdo ho hodnotil, a rozhodnutími o kroku a běhu. Je pro každý běh i release, jako JSON (qawalk.evidence.v1, stabilní kontrakt pro generátory) a CSV.

Každý řádek matice navíc říká, kdo krok provádí a do které části procesu patří: session: kroku (role, třeba expert QC) a jeho řádek na plachtě (lanes:, dílčí proces, třeba sběr dat). Když scénář lanes: nemá, dostane každá session vlastní řádek. Validátor si tak matici vyfiltruje na jednu roli v jednom dílčím procesu. Odkaz končící #lane=<číslo> (počítá se od 0) otevře plachtu na tomto řádku, #session=<klíč> zvýrazní kroky jedné role.

Snímek kroku si ve vlastním portálu s dokumentací zobrazíte vedle testovacího kroku tak, že ho portál stáhne tokenem jen pro čtení, omezeným na projekt: GET /api/v1/orgs/<org>/runs/<run>/steps/<step>/screen?locale=cs&viewport=desktop&size=thumb&sha256=<hash>. Uložte si běh, krok a SHA-256 snímku z matice dohledatelnosti. Když běh obsahuje jiný soubor, odpověď je 409, nikdy novější obrázek. Stránky Kaloka nejde vložit do jiných webů, takže portál ukáže stažený obrázek a odkáže na krok na plachtě, kde se čtenář přihlásí vlastním účtem.

Jeden scénář, víc záznamů

Scénář se napíše jednou a prochází se znovu u každé změny: zkušební běh na lokálu pro změnové řízení CC-2026-001, formální běh na stagingu pro totéž řízení, později smoke test na produkci pro CC-2026-002. Záznam a účel uveďte při startu běhu:

```
kaloko start --scenario qa/batch.yml --env staging --ref CC-2026-001 --purpose formal
```

V CI udělají totéž KALOKO_REFS=CC-2026-001 (víc hodnot oddělte čárkou nebo mezerou) a KALOKO_PURPOSE=formal. Běh unese až 20 odkazů, každý nejvýš 80 znaků: písmena, číslice a . _ - / : # +. Účel je dry_run, formal nebo smoke; je to štítek pro lidi a portály a na tom, jak se o běhu rozhoduje, nic nemění.

Plátno ukáže obojí v hlavičce běhu. Váš portál si s tokenem pro čtení vypíše běhy daného záznamu, aniž by musel číst soubory scénářů nebo hádat z názvů: GET /api/v1/orgs/<org>/runs?ref=CC-2026-001 vrátí každý běh s verdiktem, prostředím, refs a purpose; &purpose=formal ponechá jen formální. Doklady je nesou jako run.refs a run.purpose a doklady releasu berou stejné ref a purpose, aby v nich zůstaly jen tyto běhy.

Jak to zapadá do vašeho rámce

U uvedených článků auditoři takové záznamy obvykle hledají. Co se počítá jako doklad, rozhoduje vaše QA.

| Rámec | Jak Kaloko pomůže | Co zůstává na vás | |---|---|---| | ISO 9001 | Podepsané záznamy, které nejde nepozorovaně změnit, s auditní stopou (7.5); záznamy o ověření návrhu a vývoje: běhy podle kritérií, rozhodnutí lidí, odkazy na požadavky (8.3.4); záznam o přijetí říká, podle jakých kritérií, na jaké verzi a kdo přijal (8.6) | Řízení dokumentů, uchování záznamů ve vlastním archivu, samotné rozhodnutí o uvolnění | | ISO 13485 | Podklady k validaci softwaru použitého v QMS: shrnutí pro posouzení dodavatele, pevná verze CLI, naše řízení změn, verze nástrojů v každém záznamu (4.1.6); čitelné, identifikovatelné záznamy chráněné podpisem a otisky (4.2.5); výsledky ověření a validace navázané na požadavky (7.3.6, 7.3.7) | Validace vašeho použití Kaloka podle rizika, doba uchování a archiv, plán ověření a validace | | IEC 62304 | Dohledatelnost od požadavku přes test k výsledku v matici dohledatelnosti; záznamy o testování softwarového systému: co se testovalo, na jaké verzi, s jakými nástroji, kdy a kdo rozhodl; podklady k záznamu o vydání | Plán vývoje, bezpečnostní klasifikace, řízení rizik, zbylé anomálie a samotné vydání | | ISO 27001 | Akceptační kritéria a jejich výsledky jako záznam o bezpečnostním testování při vývoji a přijetí (A.8.29); přijetí navázané na verzi, PR a lidi pro řízení změn (A.8.32); balíček, který si auditor ověří offline | ISMS, posouzení rizik, prohlášení o aplikovatelnosti a váš proces změn | | GxP (21 CFR Part 11, EU GMP Annex 11) | Viz oddíl níže | Validace (CSV nebo CSA), SOP, školení, archivace a váš výklad jednotlivých požadavků | | ISO 26262, IEC 61508 | Podpůrný nástroj: záznamy z kontrol UI jako doplňkový doklad | Kvalifikace nástroje a bezpečnostní případ; Kaloko samo o sobě nástrojem pro bezpečnostní ověření není |

Kontroly, které platí dnes

• Přístup: role v organizaci a v projektech, omezené projekty a hosté; dvoufázové ověření lze vyžadovat po všech (Business); délka relace a odhlášení po nečinnosti (Business); firemní přihlášení, SCIM a povolené IP adresy (Enterprise). API tokeny mají omezený rozsah, vyprší a dají se zrušit.

• Auditní stopa: bezpečnostně důležité akce, verdikty běhů, podpisy, smazaná rozhodnutí o krocích a stažení záznamů jdou do auditního logu, do kterého se jen přidává, i s tím, kdo, kdy, odkud a jak (prohlížeč, token nebo agent). Rozhodnutí o krocích a verdikty kritérií se ukládají s autorem a časem. Auditní stopa běhu v balíčku dokladů spojuje obojí. Záznamy o rozhodnutích běhu zůstávají, dokud existuje běh; ostatní dva roky.

• Neměnnost záznamu: podpisy Ed25519 nad kanonickým JSON, SHA-256 každého souboru, zveřejněné veřejné klíče; vyřazené klíče zůstávají zveřejněné, aby starší záznamy šlo ověřit dál.

• Uchování: v plánech Business a Enterprise zůstávají přijaté běhy (a běhy, které si ponecháte) i s jejich záznamy, dokud plán trvá. Ostatní běhy vyprší po době uchování podle plánu. Člověk s odpovídající rolí může běh smazat i tak (7 dní zůstane v koši a smazání je v auditním logu). Balíčky dokladů, které potřebujete, si uložte ve vlastním systému.

• Zálohy: databázi lze obnovit k libovolné minutě posledních 30 dnů; soubory změněné za posledních 30 dnů se navíc každou noc kopírují do odděleného úložiště.

• Čas: časy služby jsou v UTC z hodin služby, které synchronizuje poskytovatel hostingu. Časy průchodu jsou z hodin stroje, který procházel.

Naše řízení změn

CLI vychází v npm se sémantickými verzemi a changelogem; kdy aktualizovat, rozhodujete vy, a verzi můžete zafixovat. Datové formáty mají verze (qawalk.run.v1, qawalk.evidence.v1, kaloko.signoff.v1): verze jen přibírá pole a nekompatibilní změna dostane novou verzi, přičemž starou půjde dál číst. Služba se aktualizuje průběžně; každý záznam uvádí verzi služby, která ho podepsala.

GxP: 21 CFR Part 11 a EU GMP Annex 11

GxP je nejpřísnější profil, se kterým Kaloko počítá. Kaloko není validovaný systém a netvrdí shodu s Part 11 ani Annex 11; tady je, co dává k požadavkům, na které se lidé ptají nejdřív.

| Požadavek | Co dává Kaloko | Co musí vyřešit vaše validace | |---|---|---| | Part 11 §11.10 (kontroly uzavřeného systému) | Přístup podle rolí, auditní stopa s tím, kdo, kdy a jak, kontrola oprávnění při přijetí a podpisu, přesné kopie záznamů (JSON, PDF, balíček) | Jestli tyto kontroly stačí pro vaše zamýšlené použití; SOP a školení | | Part 11 §11.50 (projev podpisu) | Jméno, čas s časovým pásmem a význam u každého podpisu v záznamu | Váš seznam významů | | Part 11 §11.70 (vazba podpisu a záznamu) | Podpis je navázaný na běh i na rozhodnutí (jeho poznámku a přiložené soubory přes SHA-256); podepsaný záznam nejde nepozorovaně změnit | Uložit SHA-256 záznamu do dokumentů, které podepisujete jinde | | Part 11 §11.200 (složky podpisu) | Relace člověka a při podpisu čerstvý druhý krok, který je jeho vlastní (klíč nebo aplikace s kódy) | Jestli tahle dvojice odpovídá vašemu výkladu §11.200; hesla nejsou | | Annex 11 §4 (validace) | Shrnutí pro posouzení dodavatele, pevná verze CLI, verzované formáty a naše řízení změn | Validace vašeho použití | | Annex 11 §9 (auditní stopa) | Auditní log a auditní stopa běhu v balíčku | Kontrola auditní stopy tak, jak ji chtějí vaše SOP | | Annex 11 §12 (bezpečnost) | Role, povinné dvoufázové ověření, limity relací, firemní přihlášení, povolené IP adresy | Správa uživatelů a kontroly přístupů | | Annex 11 §14 (elektronický podpis) | Volitelné podpisy s významem a novým ověřením, odvolat jdou jen s důvodem | Kde podepisujete: v Kaloku, nebo ve vašem eQMS |

Jak to napojit na váš přístup k validaci:

| Vaše činnost | Co dává Kaloko | |---|---| | Posouzení dodavatele | Shrnutí pro posouzení dodavatele, changelog a stránka o důvěře | | Instalace (IQ) | Pevná verze CLI z npm (kaloko@x.y.z); každý záznam uvádí verze CLI, Node.js, prohlížeče a služby, se kterými běh proběhl | | Provoz (OQ) | Vlastní scénář nad stránkou, kterou máte pod kontrolou, se známými úspěchy a známými chybami: deterministické kontroly musí chyby zachytit, záznamy a balíčky musí projít ověřením a změněný soubor musí ověření neprojít | | Výkon nebo UAT (PQ) | Vaše skutečné scénáře s refs:, procházené na validačním prostředí; záznam nebo balíček je doklad o provedení, rozhodnutí lidí jsou kontrola | | Zajištění podle rizika (CSA) | Deterministické kontroly pro kritéria s vysokým rizikem, kritéria s AI jen jako doporučení s rozhodnutím člověka, matice dohledatelnosti pro pokrytí |

Známé mezery

• Kaloko nikdo necertifikoval ani nevalidoval a netvrdíme shodu s žádným z rámců výše.

• Zprávu SOC 2 zatím nemáme; jak daleko je příprava, popisuje stránka o důvěře.

• Podpis v Kaloku stojí na relaci člověka (odkaz z e-mailu nebo firemní přihlášení) a čerstvém druhém kroku, který je jeho vlastní (klíč nebo aplikace s kódy). Hesla nejsou. Podpis za člověka vytvoří služba, ne jeho vlastní certifikát, a není to kvalifikovaný elektronický podpis podle eIDAS.

• Službu nejde podržet na určité verzi; zafixovat jde jen CLI.

• Zdroj času není validovaný a záznamy zatím nenesou časové razítko od nezávislé autority.

• Uchování se řídí plánem: není možné běh právně zablokovat proti smazání a PDF není PDF/A. Balíček archivujte sami.

• Auditní log drží jiné záznamy než rozhodnutí dva roky. Přibývají do něj jen přes službu, nic mimo službu ho ale nezapečetí; nezávisle si ověříte podepsaný balíček.

• Úprava textu komentáře nenechává předchozí verzi a změna rozhodnutí, které není podepsané, nechce důvod.

• Maskování osobních údajů mění to, co snímek ukazuje. Rozhodněte, jestli běhy, které si necháváte jako doklad, maskovat, nebo ne.

• Hotový balíček pro audit dodavatele zatím nemáme; na vyžádání vyplníme dotazník a popíšeme řízení změn, zálohy a obnovu.

• Hodnotitelé s AI běží u externích poskytovatelů a jejich modely se mění; jejich verdikty zůstávají doporučením.

Co si vyjasnit s vaším QA

• Kterému rámci mají doklady sloužit a u kterých článků pomáhají.

• Kterou cestou jdete: podpis v Kaloku, nebo podpis vygenerovaných dokumentů ve vašem eQMS.

• Jaké významy podpisu používáte a jestli se podepisuje i schválení kroků.

• Kde se balíčky dokladů archivují a jak dlouho.
• Která kritéria smí vůbec hodnotit AI a kdo o nich rozhoduje.
• Jak naložíte s aktualizacemi služby, které nejde odložit.

Bezpečnost a vaše data

Kaloko běží tam, kde je váš kód a váš agent: CLI prochází, zachycuje a kontroluje na vašem počítači nebo ve vašem CI. Služba na kaloko.app ukládá to, co nasdílíte, zobrazuje plachtu a sbírá rozhodnutí. Tahle stránka vysvětluje, co kam putuje. Kde jsou data uložená a které firmy je zpracovávají, najdete na stránce o důvěře a bezpečnosti, kterou průběžně aktualizujeme.

Co opouští váš počítač

Dokud nesdílíte, nic. kaloko walk, evaluate a preview pracují lokálně a místní plachta se otevře z disku. kaloko share nahraje běh: screenshoty, zachycené HTML, kritéria s důkazy a videa tam, kde je váš plán uchovává. Soubory se posílají podle otisku obsahu, takže zachycení, které se nezměnilo, se znovu neposílá.

Data o návštěvnosti jsou volitelná. S traffic.gsc nebo traffic.ga4 v konfiguraci se CLI zeptá API Search Console a Analytics od Googlu na čísla po stránkách, a to přímo z vašeho počítače nebo CI, přihlášené jako servisní účet, jehož klíč určíte proměnnou prostředí. Klíč ani přístupový token se nikdy nezapíšou do běhu, nenahrají se a zmizí z každé zprávy, která by je ukázala. Odpovědi zůstanou v mezipaměti ve složce běhů (.cache) na vašem počítači. Čísla ke stránkám (kliknutí, zobrazení, CTR, pozice, relace) skončí v items.json stacku, takže se při sdílení dostanou s během na kaloko.app. Export CSV v traffic.file se čte lokálně jako kterýkoli jiný soubor.

Co vidí evaluátor

U textových kritérií dostane evaluátor zredukovanou osnovu stránky se zamaskovanými e?maily a tokeny, surové HTML nikdy. Screenshot jde evaluátoru na obrázky jen u kritérií označených vision: true a u odpovědí, kterými si textový evaluátor nebyl jistý. S vlastním klíčem jde požadavek z vašeho počítače přímo k danému poskytovateli, bez něj přes Kaloko. Když si nastavíte vlastní model (kind: openai_compatible), osnova stránky odchází jen na adresu, kterou jste zapsali do base_url, třeba do Ollamy na vašem počítači.

Tajné údaje zůstávají venku

Tajné údaje se do kaloko.config.yml nikdy nepíšou. Konfigurace jen pojmenuje proměnnou prostředí podle vašeho výběru (password_env: STAGING_PASSWORD) a hodnota se načte z prostředí, z .env projektu nebo z ~/.config/kaloko/.env. Hodnoty těchto proměnných se ze zachyceného HTML, textu stránky i výstupu konzole odstraní dřív, než se cokoli uloží. Přihlášení uložená přes kaloko auth save zůstávají na vašem počítači a nikdy se nenahrávají.

Osobní údaje zůstanou skryté

Masky skryjí osobní údaje ve všem, co krok uloží, nejen ve screenshotu: v zachyceném HTML, v osnově pro evaluátory, v průběhu kroku, ve videu i v zachycených e?mailech. S pii: redact najde Kaloko e?maily, telefony, rodná čísla, IBAN a adresy samo; pii: pseudonymize místo nich dosadí smyšlené údaje, stejné pro stejný údaj. Prostředí s pii: required bez toho běh nedostane a kaloko share odmítne nahrát běh, ve kterém zůstal nezamaskovaný e?mail nebo telefon. Víc v návodu Skrýt osobní údaje ve snímcích.

Produkce jen pro čtení

Prostředí označené jako produkce je v CLI vždy jen pro čtení: veřejné stránky jako návštěvník, žádné přihlašování do administrací, žádná vytvořená data. Aby tam scénář mohl běžet, musí uvést readonly: true.

Běhy vyprší

Každý sdílený běh má datum expirace, výchozí je 30 dní a nejvýš retence vašeho plánu, a potom se smaže. kaloko keep běh ponechá natrvalo; v plánech Business a Enterprise přijaté běhy nevyprší, dokud plán trvá. Od plánu Team si běh můžete kdykoli stáhnout jako ZIP přes kaloko export.

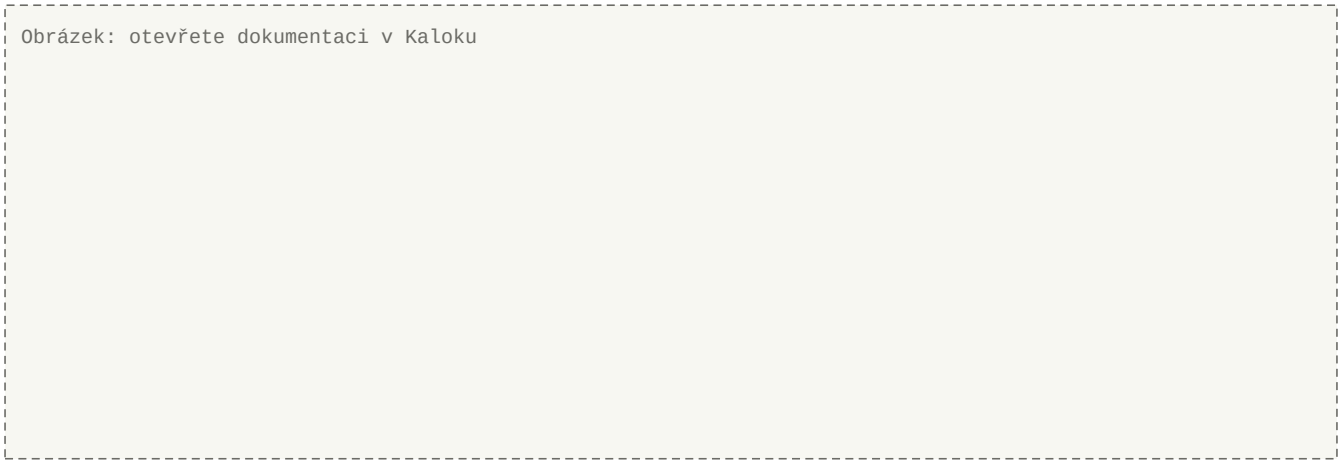
Jak se servírují soubory

Nahrané soubory se servírují z oddělené domény přes krátkodobě podepsané odkazy. Zachycené HTML se ukazuje v sandboxu bez skriptů.

Kdo se dostane dovnitř

Lidé se přihlašují jednorázovým odkazem z e?mailu nebo přes Google, Microsoft či GitHub a jako druhý krok si můžou přidat passkey nebo autentizační aplikaci. Kdo se přihlásí přes Google, Microsoft či GitHub a ještě nepatří do žádné organizace, může si ji založit; adresu už ověřil poskytovatel. Passkey platí, jen když se zeptá na PIN, otisk prstu nebo obličej. Enterprise přidává firemní přihlášení přes SAML nebo OIDC a SCIM.

Co člověk smí, určuje role: viewer čte, komentuje, schvaluje a vrací; designer a vývojář / tester navíc nahrávají; admin navíc spravuje členy, tokeny a nastavení. Vieweři jsou v každém plánu zdarma. API tokeny mají vlastní rozsah a platnost a token, který CLI dostane přes kaloko login, platí nejvýš 30 dní.



API tokeny v Nastavení · cs · desktop

Na co se ptá bezpečnostní prověrka

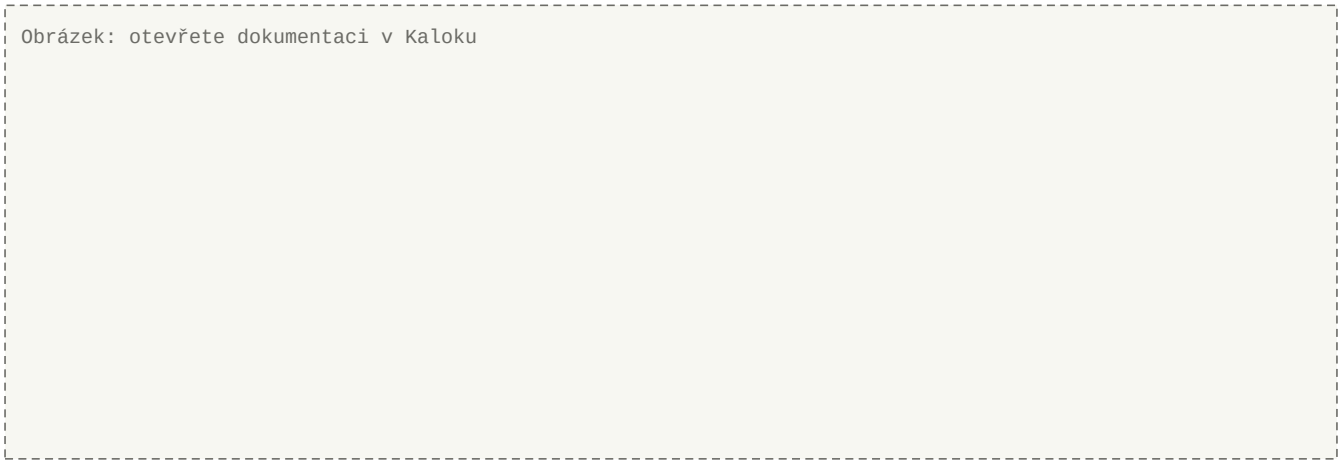
Stránka o důvěře a bezpečnosti odpovídá, kde jsou data, jak jsou šifrovaná, kdo je zpracovává a jak daleko je příprava na SOC 2. Kvůli nahlášení zranitelnosti nebo smlouvě o zpracování osobních údajů napište na kontakt, který tam najdete.

Předat podepsaný záznam o přijetí

Když se klient nebo auditor zeptá, kdo release převzal, dejte mu záznam o přijetí běhu. Je v něm, kdo co a kdy schválil, výsledky kritérií a otisk každého screenshotu, a Kaloko ho podepíše. Záznamy jsou součástí plánů Business a Enterprise.

Stáhněte ho

1. Otevřete přijatý běh a jeho panel Záznam o přijetí.



Záznam o přijetí u přijatého běhu · cs · desktop

2. Pro lidi zvolte Stáhnout PDF, pro systém, který záznamy archivuje, JSON. PDF nese stejný JSON uvnitř.

Běh, který přijatý není, záznam o přijetí nemá. Nejdřív ho přijměte, nebo o to požádejte toho, kdo to má udělat.

Podepište přijetí, pokud to vaše organizace chce

Pokud admin zapnul elektronické podpisy (Nastavení -> Zabezpečení), přijetí běhu otevře krátký dialog: vyberete, co váš podpis znamená, zkontrolujete jméno a potvrdíte druhý krok ověření, pokud o něj Kaloko požádá znovu. Záznam pak uvede vaše jméno, e-mail, čas s vaším časovým pásmem a význam podpisu. Když podepsané rozhodnutí později měníte, Kaloko se zeptá proč a důvod v záznamu zůstane. S vypnutými podpisy přijímáte jako obvykle a nic se neptá.

Předejte balíček dokladů

Pro vlastní dokumentaci, nebo když auditor chce všechno pohromadě, stáhněte balíček dokladů: otevřete u běhu panel Doklady běhu a zvolte Balíček dokladů (ZIP), nebo spusťte kaloko signoff <run-id> --package. Funguje pro každý nahraný běh, přijatý i nepřijatý, a obsahuje podepsaný záznam (JSON a PDF), evidence.json, matici dohledatelnosti (trace.csv, trace.json), auditní stopu běhu, snímky a podepsaný seznam všech souborů s jejich SHA-256. Matice dohledatelnosti (CSV) stáhne jen matici. Pokud výsledné dokumenty píšou vaše vlastní nástroje, přečtěte si Doklady o kvalitě pro váš systém kvality: co doklady říkají a co musí pokrýt váš systém.

Ověřte záznam, který jste dostali

Záznam ověří kdokoli i bez účtu. Vývojáři spustí kaloko signoff --verify zaznam.pdf; příkaz zkontroluje podpis proti veřejnému klíči Kaloka a řekne, jestli se v záznamu něco změnilo. Stačí upravit jediný znak a ověření neprojde. Stejný příkaz ověří i balíček dokladů (kaloko signoff --verify balicek.zip): každý soubor musí odpovídat podepsanému seznamu a soubor přidaný později příkaz vyjmenuje.

Přidat kritéria a packy kontrol

Každý krok má svá kritéria: co musí obrazovka splňovat. Kaloko zná tři druhy a dobrý scénář použije ten nejjednodušší, který na otázku odpoví.

Dá se spočítat: deterministická kontrola

Co se dá spočítat nebo přečíst ze stránky, se ověří přímo ve stránce a k výsledku se přiloží důkaz:

```
criteria:
- id: AC1
text: Košík ukazuje dvě položky.
check: { type: deterministic, assert: count, selector: ".cart-item", min: 2, max: 2 }
- id: AC2
text: Po zaplacení se zobrazí poděkování.
check: { type: deterministic, assert: url_matches, pattern: "/thanks$" }
- id: AC3
text: Tlačítko Zaplatit má barvu značky.
check: { type: deterministic, assert: style_equals, selector: "button.pay", property:
background-color, token: color.brand.primary }
```

Další kontroly čtou text, atributy, HTTP status, přetečení na telefonu nebo hodnotu, kterou stránka vystavuje (js_equals). Celý seznam je v referenci kontroly a packy. Skript kroku nikdy nemusí nechávat značku, kterou by kritérium hledalo.

Otázka: sémantická kontrola

Co chce úsudek, dostane evaluátor jako otázku ke stránce:

```
- id: AC4
text: Hlavní výzva k akci je zřejmá.
check: { type: semantic, question: Ukazuje stránka jednu zřetelnou hlavní výzvu k akci?,
scope: main }
```

Evaluátor si přečte zredukovanou osnovu stránky a odpoví skóre. Jasné ano projde, jasné ne neprojde a všechno mezi tím rozhodne člověk. Když jde o vzhled (rozložení, obrázky, "vypadá to jako naše značka“), přidejte ke kontrole vision: true: na screenshot se pak podívá ještě model, který vidí obrázky. Používejte to jen u kroků, kde je to potřeba.

Rozhodne člověk: ruční kontrola

```
- id: AC5
text: Právníci schválili znění souhlasu.
check: { type: manual }
```

Ruční kritérium zůstane "nehodnoceno“, dokud o něm kontrolující nerozhodne na plachtě.

Packy kontrol

Pack přidá ke kroku hotovou sadu kontrol, takže běžné kontroly nepíšete ručně. Jsou to kontroly kondice: běží při každém průchodu a na plachtě se ukazují zvlášť, ale o verdiktu kroku nerozhodují (výjimkou je pack design). Víc v návodu Kontrola kondice aplikace.

```
nodes:
- id: home
packs: [seo, a11y, perf, console]
```

| Pack | Co kontroluje | |---|---| | seo | jeden H1, titulek a popis, canonical, hreflang, Open Graph, indexaci, html lang | | usability | účel, jednu hlavní akci, texty odkazů, pole bez popisku | | a11y | alternativní texty, názvy, popisky, nadpisy, kontrast, landmarky, průchod tabulátorem a viditelný focus | | perf | LCP, CLS, TTFB a váhu stránky naměřené při průchodu | | console | jestli chyby v konzoli souvisejí s tím, k čemu krok slouží | | errors |

chybové stránky: skutečný status 4xx nebo 5xx, cestu zpět, žádný výpis zásobníku | | app | obrazovky aplikací: popisky ovládacích prvků, velikost dotykových ploch, useknutý text, pády | | design | implementaci proti přijatému návrhu: pixely, strukturu, tokeny, kontrast |

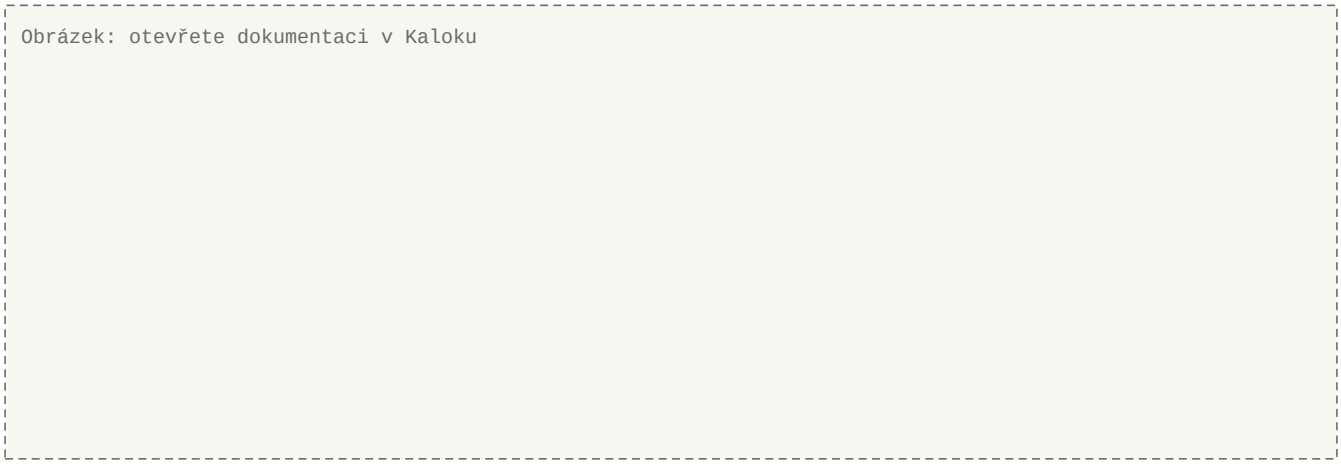
Limit změníte vlastním kritériem s číslem kontroly jako id AC (třeba AC401 pro kontrolu LCP HC401); vaše kritérium má přednost, a protože je vaše, počítá se do přijetí. kaloko health adopt <krok>:<kontrola> takové kritérium zapíše za vás, s from: HC401, aby bylo jasné, odkud pochází. Jak se z výsledků skládá verdikt, vysvětluje stránka jak vzniká verdikt.

Schválit nebo vrátit běh

Teprve vaše rozhodnutí udělá z "agent říká, že je hotovo“ opravdu hotovou věc. Rozhodujete po krocích a pak za celý běh.

Rozhodněte o každém kroku

1. Otevřete krok a přečtěte si jeho kritéria. Zelené kritérium zkontrolovalo Kaloko; u kritéria, které potřebuje člověka, to stojí přímo u něj a rozhodnutí je na vás.



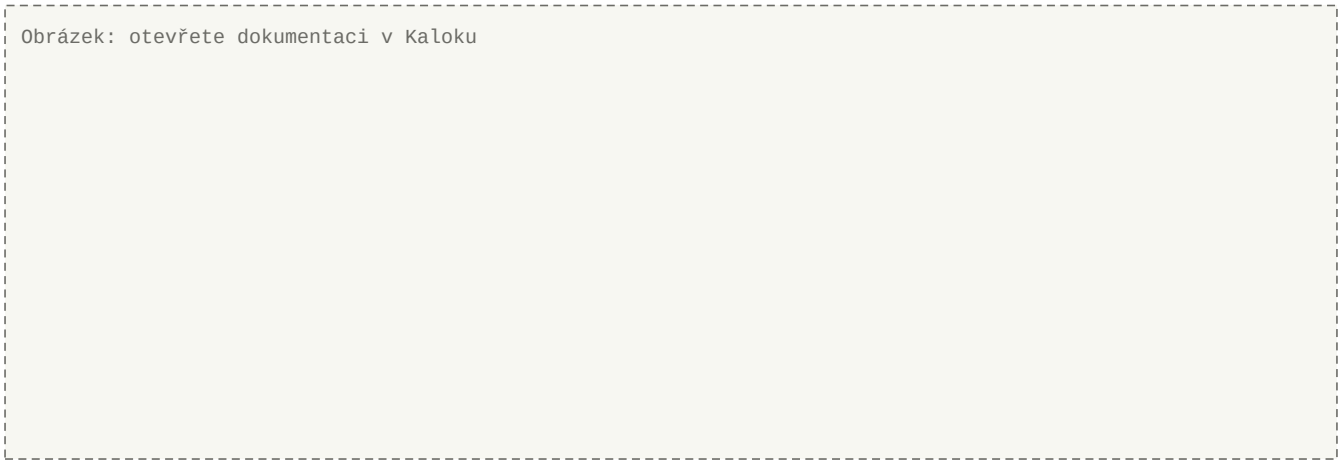
Krok s kritérii a komentáři · cs · desktop

2. Když krok sedí, zvolte pod komentáři Schválit. Když ne, napište, co se má změnit, a zvolte Zamítnout: "celková cena poskočí, když se načte cena“, ne "oprav to“. Agent si poznámku přečte a podle ní pokračuje.

Rozhodněte o běhu

1. Když jsou schválené všechny kroky, zvolte Přijmout běh. Běh se stane obrázkem této verze: novinky i dokumentace od té chvíle ukazují jeho obrazovky.

2. Pokud se má něco nejdřív změnit, zvolte Vrátit běh. Kdo ho sdílel, dostane vaše poznámky; až nasdílí další běh, komentář v pull requestu ukáže na něj.



Běh na plachtě · cs · desktop

Příště to bude kratší

V dalším běhu si kroky, jejichž obrazovky se nezměnily, vaše schválení ponechají. Díváte se jen na to, co je nové nebo jiné, a na plachtě je vidět, která schválení se přenesla a z kterého běhu.

Agenti mohou komentovat a navrhovat, ale běh přijímá jen člověk.

Projít víc prohlížečů a zařízení

Běh ve výchozím stavu zachytí každý krok v Chromiu, v každém jazyce a viewportu scénáře. Přidat můžete další prohlížeče, předvolby zařízení, tmavý režim a několik dalších nastavení. Každá kombinace je samostatný snímek, takže přidávejte jen to, o čem úkol je: chyba v Safari potřebuje WebKit u dotčených kroků, ne všechna nastavení u všech kroků.

Výchozí nastavení snímků

Nový scénář (z kaloko init, kaloko draft nebo ze šablony) snímá ve dvou velikostech:

| Název | Velikost | Proč | |---|---|---| | desktop | 1280 × 800 | běžný notebook | | mobile | 360 × 800 | nejužší běžná šířka telefonu, na které se jako první ukáže, co se nevejde |

Každý krok se snímá jako celá stránka, odshora dolů. Tablet ve výchozím nastavení chybí, protože ho většina úkolů nepotřebuje a každá velikost přidá snímky. Vypnuté jsou i očíslované značky na kontrolovaných prvcích (capture: { annotate: true }).

Změníte to ve viewports: ve scénáři. Tablet přidáte jako tablet: [744, 1133] (iPad mini na výšku); od šířky 900 už většina webů ukazuje desktopovou verzi. capture: { full_page: false } u kroku nechá jen první obrazovku. Scénář, který už máte, si své velikosti nechá, takže jeho baseline zůstanou porovnatelné; scénář bez viewports: se snímá v 1280 × 800 a 390 × 844 jako dřív.

Ve scénáři

```
viewports: [iphone-se, iphone-15, pixel-8, ipad-landscape, laptop]
browsers: [chromium, webkit, firefox]
color_schemes: [light, dark]
reduced_motion: [no-preference, reduce] # volitelné
forced_colors: [none, active] # volitelné, vysoký kontrast ve Windows
display_modes: [browser, standalone] # volitelné, stránka jako nainstalovaná webová aplikace
```

Výchozí hodnoty pro celý projekt patří pod walk: v kaloko.config.yml. Příkazová řádka má přednost před scénářem a scénář před konfigurací.

Pro jeden běh

```
npx kaloko start --scenario qa/flows/checkout.yml --env staging \
--browsers chromium,webkit --color-schemes light,dark --viewports
iphone-se,pixel-8,desktop-hd
npx kaloko walk # jeden průchod pro každý prohlížeč a zařízení
npx kaloko walk --browsers webkit # jen snímky běhu ve WebKitu
```

Předvolby zařízení

npx kaloko devices vypíše předvolby: telefony (iphone-se, iphone-15, pixel-8, galaxy-s24…), skládací telefony (fold-folded, fold-open), tablety (ipad-portrait, ipad-landscape…), počítače (laptop, desktop-hd, desktop-wide) a tv-1080p. Přípona -landscape otočí telefon, skládačku nebo tablet na šířku (iphone-15-landscape). Předvolba nastaví velikost, poměr pixelů, dotyk a user agent zařízení. Odhalí chyby rozložení, zbytek najde až skutečné zařízení.

Nainstalujte prohlížeče

WebKit (jádro Safari) a Firefox stáhnete jednou na každý počítač nebo job v CI:

```
npx kaloko browser install --browsers webkit,firefox
```

npx kaloko doctor vypíše prohlížeče, které vaše scénáře potřebují, a řekne, jak doinstalovat chybějící. V CI na Linuxu přidejte --with-deps kvůli systémovým knihovnám.

Co čekat

• Několik kontrol běží jen v Chromiu: názvy ze stromu přístupnosti, viditelný focus a posuny rozložení, ve WebKitu navíc průchod tabulátorem. V ostatních prohlížečích jsou "nehodnoceno“ i s důvodem a krok kvůli nim není částečně splněný.

• Kontrast se kontroluje pro každé barevné schéma zvlášť, takže chyba kontrastu v tmavém režimu neprojde na tmavém snímku.

• Běh odmítne víc než 400 kombinací.

Na plachtě najdete přepínač prohlížeče, barevného schématu a dalších nastavení vedle přepínače jazyka. Detail kroku s předvolbou zařízení vykreslí jeho rámeček a volba Vedle: ukáže stejný krok v jiném prohlížeči nebo schématu vedle něj.

Kontrola kondice aplikace

Klient chce nový košík. Běh se vrátí červený, protože stránce s objednávkou chybí meta popis, o který nikdo nežádal. Kaloko proto takové kontroly drží stranou: kontroly z packů (SEO, přístupnost, výkon, použitelnost, konzole, chybové stránky, obrazovky aplikací) tvoří kondici aplikace. Běží při každém průchodu, ale nikdy nerozhodují, jestli krok nebo běh prošel. O přijetí dál rozhodují kritéria ze zadání.

| | Přijetí | Kondice | |---|---|---| | Otázka | dělá práce, co se zadalo? | je aplikace v dobrém stavu? | | Odkud jsou kritéria | ze zadání, z ticketu, od lidí | z packů, automaticky | | Počítá se do verdiktu | ano | ne, jen radí | | Hlavně pro | člověka, který schvaluje | agenta před review; tým v čase |

Jak ji zapnout

Kondice vzniká z packů kroku, stejně jako dřív:

```
nodes:
- id: home
packs: [seo, a11y, perf, console]
```

Výjimkou je pack design: kontroluje práci proti návrhu, který tým přijal, a proto se do přijetí počítá vždy. Všechny packy a to, na co se která jejich kontrola dívá, najdete v Kontrolách kondice.

Když se scénář prochází na designovém prostředí, packy, které splní jen kód (seo, aeo, perf, console, errors), jsou vypnuté: HTML od designéra nemá skutečný server, meta značky ani chybové stavy. Pokud některý chcete zachovat, uveďte ho v health: { design_packs: [...] }. Které packy jsou zapnuté a které ne, ukazuje stránka scénáře v aplikaci.

Jak ji číst

Po průchodu vypíšou kaloko walk a kaloko evaluate jeden řádek s počty. Podrobnosti:

```
npx kaloko health # aktuální lokální běh
npx kaloko health --remote # sdílený běh na kaloko.app
npx kaloko health --json # pro skripty a agenty
```

Každý nález má závažnost. Chyba znamená, že lidé nebo vyhledávače zůstanou venku: stránka vyřazená z indexace, chybějící viewport pro telefony, chyby v konzoli, tlačítko bez názvu, past pro klávesnici. Varování je všechno ostatní, třeba příliš dlouhý titulek nebo pomalá stránka. Nejisté znamená, že si evaluátor nebyl jistý. Výpis také řekne, co přibylo nebo se spravilo od minulého běhu scénáře, a ke každé kontrole, jak ji opravit.

Na plachtě má kondice vlastní kartu. V panelu běhu je Kondice vedle Souhrnu: počty, změna proti minulému běhu a nálezy podle packů nebo kroků; kliknutím se krok otevře na kartě Kondice, vedle Kritérií, kde jsou jen akceptační kritéria. Kontroly kondice mají vlastní id (HC103), takže si je nikdo nesplete s akceptačními kritérii (AC1, AC2…). Seznamy běhů ukazují počty jako malý odznak a stránka Kondice u projektu kreslí vývoj v čase.

S --fail-on error skončí kaloko health kódem 1, když jsou v kondici chyby. Použijte to jen tam, kde má kondice zastavit pipeline; nic jiného v Kaloku kvůli kondici neselže.

Jak z kontroly udělat kritérium přijetí

Někdy kontrola kondice do zadání patří ("každá produktová stránka má meta popis do 155 znaků“). Převezměte ji, beze změny nebo vlastními slovy:

```
npx kaloko health adopt product:HC103 --text "Každá produktová stránka má meta popis do 155 znaků"
```

Kontrola se zapíše do souboru scénáře jako kritérium AC103 kroku, s from: HC103, a od dalšího průchodu rozhoduje o jeho verdiktu. V kondici kroku už HC103 nebude. Starší běhy, které kontrolu označují AC103, přijmou i toto id. Na plachtě udělá totéž tlačítko Převzít do kritérií u nálezu, pro lidi s rolí Developer / Tester; agent ho do scénáře zapíše příkazem kaloko health pull.

Chcete-li nechat v přijetí celý pack, jako tomu bylo před zavedením kondice, uveďte ho ve scénáři:

```
health:
acceptance: [seo]
```

Jak se dozvědět, že se kondice zhoršila

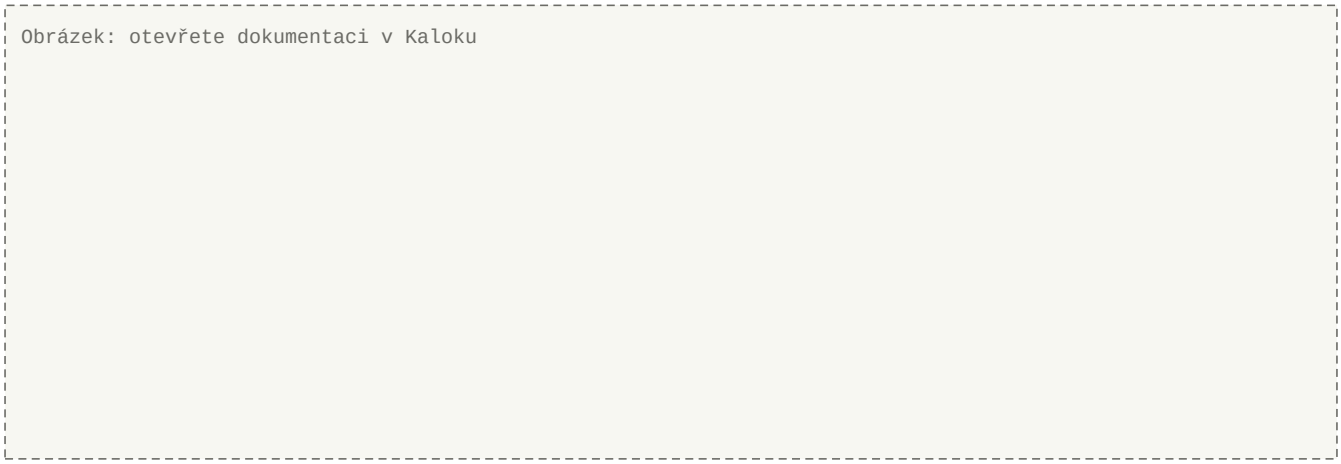
Hostované běhy a běhy sdílené z CI porovnají kondici s minulým během scénáře. Když přibude nová chyba, Kaloko dá vědět sledujícím scénáře do schránky a e?mailem, napíše do kanálů ve Slacku a Teams, které odebírají "Zhoršená kondice“, a pošle webhook health.worse. U každého plánu si vyberete, jestli hlásit nové chyby (výchozí), i nová varování, nebo nic: Nastavení -> Integrace -> Běhy na serveru Kaloka.

Nastavit firemní přihlášení

Pro adminy organizace. Kaloko se přizpůsobí přihlášení, které ve firmě už máte: osobní přihlášení v každém plánu, povinný druhý krok pro celou organizaci od plánu Business a vašeho poskytovatele identit se SCIM v plánu Enterprise.

Přihlášení v každém plánu

Lidé se přihlašují jednorázovým odkazem z e-mailu, nebo přes Google, Microsoft či GitHub. Každý si v nastavení účtu může přidat passkey nebo autentizační aplikaci jako druhý krok.



Přihlašovací stránka · cs · desktop

V plánu Business může admin v Nastavení -> Zabezpečení vyžadovat druhý krok po celé organizaci a nastavit limity relací. Kdo druhý krok nemá, musí si ho přidat, než bude pokračovat.

Nejdřív ověřte doménu

Firemní přihlášení i SCIM se týkají jen adres z ověřených domén.

1. Otevřete Nastavení -> Lidé a přidejte doménu v části Domény pro přihlášení.

2. Přidejte do DNS záznam TXT, který Kaloko ukáže (název _kaloko.<doména>, hodnota kaloko-verify=…), nebo doménu prokažte přihlášením přes Google Workspace či Microsoft 365. Záznam _qawalk přidaný před přejmenováním platí dál.

3. Stiskněte Ověřit. Dokud doména není ověřená, nikdo se přes ni nepřipojí.

Připojte poskytovatele identit (Enterprise)

1. Otevřete Nastavení -> Zabezpečení -> Firemní přihlášení.

2. Zvolte SAML (Okta, Entra ID, Google Workspace a další) a vložte metadata poskytovatele, nebo OIDC s issuerem, client ID a secretem. Metadata Kaloka i s certifikátem najdete na stejné stránce.

3. Stiskněte Vyzkoušet přihlášení. Ukáže, co poskytovatel poslal a jestli by to Kaloko přijalo; test nikoho nepřihlásí.

4. Přihlášení vynuťte. Lidé z ověřených domén se pak přihlašují jen přes vašeho poskytovatele. Admini s passkey nebo autentizační aplikací mají cestu dovnitř i v den, kdy poskytovatel nefunguje.

Další volby na stejné stránce:

• Šifrované assertion: vytvořte pár klíčů organizace; nový pár nechá starý fungovat, dokud ho neodeberete.

• Jednotné odhlášení: odhlášení u poskytovatele odhlásí i z Kaloka a naopak.

• Přihlášení z dlaždice aplikace u poskytovatele, vypnuté, dokud ho nezapnete, s výchozí stránkou, kam lidé

přijdou.

• Druhý krok u vašeho poskytovatele může Kaloko uznat jako svůj, pro přihlášení přes vašeho poskytovatele.

Hosté z jiných firem se přihlašují jako dřív a vidí jen projekty, do kterých byli pozváni.

Zřizujte lidi přes SCIM (Enterprise)

Základní adresu SCIM a token z Nastavení vložte do Okty nebo Entra ID. Přiřazení lidé přibudou s výchozí rolí své domény, skupiny se mapují na projekty a role a odebrání člověka ukončí jeho relace, zneplatní jeho tokeny a vyřadí ho ze všech projektů. Admina SCIM nikdy nevytvoří.

Token pro SCIM platí rok. Nastavení ukazuje do kdy a "Prodloužit o rok“ si token nechá, takže u poskytovatele identit nic neměníte. Adminům 30 dní před koncem přijde připomínka do schránky v Kaloku.

Omezte, odkud se lidé připojují (Enterprise)

V Nastavení -> Zabezpečení povolíte rozsahy IP adres pro aplikaci a zvlášť pro API, CLI a MCP. Audit log může proudit na podepsaný webhook, do Splunku nebo na endpoint kompatibilní s Datadogem.

Kde jsou vaše data a kdo je zpracovává: Bezpečnost a data.

Porovnat s referenčním během a ignorovat, co se mění pořád

Před vydáním chcete znát jednu odpověď: co vypadá jinak než verze, kterou tým přijal, a jestli je to záměr. Dobrý běh scénáře se stane jeho referencí a každý další běh se s ní porovná.

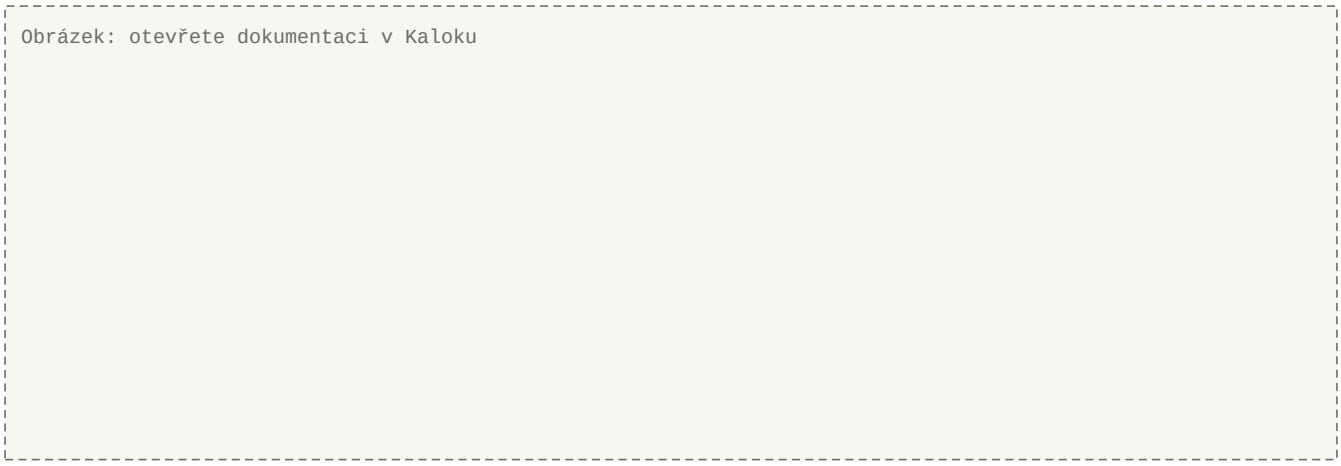
Nastavte referenci

Na plachtě otevřete běh, se kterým jste spokojení, zvolte Přijmout běh a potom Nastavit jako referenci. Každý scénář má jednu referenci a ta se změní, jen když někdo nastaví novou. Z terminálu:

```
npx kaloko reference --scenario checkout --baseline --run <run-id>
npx kaloko reference --scenario checkout # ukáže současnou
```

Porovnejte běh

Na plachtě zvolte Porovnat a referenci (nebo předchozí běh, případně jakýkoli jiný běh scénáře). Každý krok dostane štítek s podílem změněných pixelů a označí se kritéria, kterým se změnil verdikt nebo důkaz.



Běh porovnaný s referencí · cs · desktop

V detailu kroku se na změnu podíváte čtyřmi způsoby: Vedle sebe, Rozdíl, Posuvník (táhnete čáru přes oba screenshoty) a Prolnutí (jeden přechází do druhého). Klávesa G ukáže jeden krok ve všech jazycích a velikostech obrazovky najednou.

Stejné porovnání jako text, pro agenta nebo poznámky k vydání:

```
npx kaloko compare # aktuální běh proti referenci, jinak proti předchozímu běhu
npx kaloko compare --run <id> --baseline <id> --json
```

Ignorujte, co se mění pořád

Hodiny, reklamy, karusely a "před 3 minutami“ se mění při každém běhu a skutečné rozdíly v nich zapadnou. Vynechte je z porovnání pixelů pomocí ignore: u kroku:

```
nodes:
- id: home
ignore:
- { selector: ".ticker", note: živé ceny }
- { rect: [0, 620, 1280, 180], viewport: desktop }
```

Hledat je ručně nemusíte:

```
npx kaloko walk --stability 3 # tři screenshoty každého snímku krátce po sobě
npx kaloko stability # co se pohnulo, ve srovnání s dřívějšími běhy scénáře
npx kaloko stability --apply # zapíše návrhy do ignore:
```

Než návrhy použijete, podívejte se na screenshoty: oblast, která se změnila, protože se opravdu změnila stránka, není šum.

Šum mohou označit i kontrolující. V detailu kroku na kaloko.app zvolí Označit proměnlivou oblast a nakreslí na screenshot obdélník. Ten se pak čárkovaně ukáže u každého běhu flow a rozdíl na plachtě ho hned vynechá. Kreslit jde s rolí Vývojář / tester nebo Admin, v běhu (ne v konceptu) a v režimu Snímek; jinak je tlačítko šedé a řekne proč. Do scénáře tyto oblasti dostanete příkazem:

```
npx kaloko stability --pull
```

Dotčené kroky pak projděte znovu: snímek si zapamatuje, které oblasti vynechal.

Připojit Jiru, Linear, Slack a Teams

Pro adminy organizací v plánu Team a vyšších. Po připojení lidé založí z vráceného kroku úkol přímo na plachtě a každý projekt dá vědět do svého kanálu, co se stalo.

Jira nebo Linear

1. Otevřete Nastavení -> Integrace a připojte Jira Cloud (přihlášením přes Atlassian, nebo vložením API tokenu) či Linear (přihlášením, nebo vložením API klíče).

2. Každý projekt v Kaloku přiřaďte k projektu a typu úkolu v Jiře, nebo k týmu v Linearu. Výchozí řádek pokryje projekty, které nevyjmenujete.

3. U Jiry přidejte do svého webu Jira webhook, který vám Kaloko ukáže, aby dokončený úkol hned uzavřel svůj komentář v Kaloku. Bez něj to Kaloko kontroluje každých 30 minut.

Lidé pak u kroku zvolí Nové issue. Úkol dostane screenshot, kritérium a poznámky kontrolujících i s odkazem zpět.

Scénář, který hlídá kondici webu (SEO, přístupnost), umí taky sám založit jedno issue na každý nový nález, zavřít ho, když je nález opravený, a okomentovat ho, když zasáhne o hodně víc nebo méně stránek. Zapíná to Developer / Tester u každého scénáře zvlášť: API kondice pro dashboardy, oddíl Nálezy.

GitHub Issues nebo Plane pro nálezy

Nálezy můžou jít i do GitHub Issues a Plane; issues z kroků zůstávají v Jiře a Linearu.

• GitHub: aplikace Kaloko na GitHubu zakládá issues na repozitářích, kde je nainstalovaná, jakmile na GitHubu přijmete její oprávnění Issues. Pro ostatní repozitáře vložte fine-grained token v bloku GitHub Issues pro nálezy (jen repozitáře, do kterých se synchronizuje, Issues: read and write).

• Plane: v bloku Plane vložte slug workspace a API klíč (Profile settings -> Personal access tokens). Pro Plane Cloud nechte adresu prázdnou; Plane na vlastním serveru musí mít veřejnou https adresu.

Kaloko token nebo klíč před uložením ověří a už ho nikdy neukáže. Pak u každého projektu v Kaloku vyjmenujte repozitáře a projekty v Plane, kam smí jeho scénáře posílat nálezy (výchozí řádek pokryje ostatní); Developer / Tester může vybírat jen z nich. Kam který scénář posílá které nálezy, se nastavuje u scénáře: repozitář, projekt v Plane a cesty podle kontroly nebo balíčku.

Kanály ve Slacku nebo Teams

1. V Nastavení -> Integrace přidejte k projektu kanál: pro Slack tlačítkem Add to Slack, pro Microsoft Teams adresou toku z Workflows.

2. Zaškrtněte události, které kanál chce: nový běh, vrácení, přijetí, komentář, zmínka, zapsaný release, čeká na schválení, zhoršená kondice.

3. Nastavte tiché hodiny, když kanál nemá od Kaloka v noci nic slyšet; zprávy počkají, až skončí.

Omezený projekt hlásí jen do kanálů, které ho výslovně jmenují. Kanál ve Slacku pro celou organizaci funguje dál.

Ptát se schvalovatelů ve Slacku a Teams

Když běh čeká na povinné schvalovatele svého scénáře, každý z nich dostane e-mail a položku v inboxu. Je-li připojený Slack, Kaloko navíc napíše každému schvalovateli, jehož e-mailová adresa patří někomu ve vašem workspace. Odkaz ve zprávě otevře běh rovnou v režimu kontroly.

• Workflow v Teams neumí napsat jednomu člověku, proto Kaloko v Teams jmenuje schvalovatele v kanálech, které odebírají Čeká na schválení. Kanály ve Slacku s touto událostí je jmenují také.

• Když nikdo nerozhodne do 24 hodin, každý schvalovatel dostane jednu připomínku. Počet hodin změníte nebo

připomínku vypnete v Nastavení -> Integrace -> Ptát se schvalovatelů ve Slacku a Teams.

• Klidové hodiny tamtéž podrží přímé zprávy do rána v časovém pásmu každého schvalovatele.

• Pokud je Slack připojený z doby, kdy Kaloko nemohlo psát lidem, ten blok nastavení to řekne. Připojte Slack jednou znovu.

• Hosté, kteří si u projektu vypnuli žádosti o schválení, zprávu nedostanou.

Exportovat dokumentaci a changelog

Váš web s dokumentací už možná má vlastní build, nebo klient chce návod jako PDF. kaloko export zapíše dokumentaci i changelog s obrázky jako obyčejné soubory, takže nic nezůstane zamčené v Kaloku.

Vyberte formát

```
npx kaloko export docs --format docusaurus --out website/docs
```

| --format | Co dostanete | Plán | |---|---|---| | md | Markdown se složkou assets/ s obrázky, pro jakýkoli statický web | každý plán | | json | changelog jako strukturovaný JSON | každý plán | | docusaurus | složka pro Docusaurus s front matter, kategoriemi a postranním menu | od plánu Business | | vitepress | složka pro VitePress s konfigurací | od plánu Business | | mkdocs | složka pro MkDocs s mkdocs.yml | od plánu Business | | html | HTML pro offline čtení: stránka na každý návod, changelog a stránka pro tisk | od plánu Business | | pdf | jedno PDF vytištěné prohlížečem | od plánu Business |

Z odkazů na kroky se stanou obyčejné obrázky v jazyce a velikosti obrazovky exportu. Krok, jehož záznam se v Kaloku přehrává, přijde i se záznamem: přehrávač s obrázkem jako náhledem a soubory MP4 a WebM v assets/ (v MkDocs obrázek odkazuje na záznam). V PDF je jen obrázek.

Vyberte, co do exportu patří

• --what docs nebo --what changelog vyexportuje jen jedno z nich; výchozí jsou obě.

• --version 2.4.0 vyexportuje dokumentaci vydání místo té nejnovější.

• --lang cs vyexportuje jeden jazyk.

• --product shop vybere produkt, když jich konfigurace má víc.

• --out určí složku nebo soubor, kam se export zapíše.

Publikované, nebo lokální

Bez dalších voleb export vezme to, co jste publikovali přes kaloko docs publish. S --local vezme soubory ve vašem repozitáři, aniž by je publikoval. Hodí se to, když chcete změnu zkontrolovat dřív, než půjde ven.

Udržujte web s dokumentací v souladu

Existující web si nechá vlastní build. Buď do jeho složky vyexportujete a výsledek commitnete, nebo export spustíte v jeho CI před buildem:

```
npx kaloko export docs --format vitepress --what docs --out docs-site/guide
```

Token v CI potřebuje jen čtení; viz Role a API tokeny.

Stáhněte export ze stránky dokumentace

Čtenáři, kterým vyhovuje soubor, použijí Stáhnout na stránce dokumentace nebo changelogu. Nabídne stejné formáty; v plánech pod Business jsou i tam jen Markdown a JSON.

Každý export se otevře i bez Kaloka. Odkud stránky pocházejí a jak je držet aktuální, popisuje Publikovat živou dokumentaci a hlídat zastaralé části.

Skrýt osobní údaje ve snímcích

Když procházíte kopii produkce nebo čtete produkční stránky, dostanou se do prohlížeče skutečná jména a e?maily. Kaloko je umí skrýt ve všem, co krok uloží: ve screenshotu, v zachyceném HTML, v osnově stránky pro evaluátory, v průběhu kroku, ve videu i v zachycených e?mailech. Kontroly přitom běží nad skutečnou stránkou, takže kritérium "rekapitulace ukazuje jméno z formuláře“ dál dostane správný verdikt.

Zamaskovat, co znáte

Prvky s osobními údaji vyjmenujte v capture.mask, u scénáře nebo u jednoho kroku:

```
capture:
mask: [".customer-name", "[data-qa=email]", ".payer"]
```

Masky scénáře a kroku se sčítají. Na screenshotu dostane každý prvek tmavý pruhovaný obdélník. V HTML se jeho text změní na [skryto] doplněné na stejnou délku ([hidden] v anglických běžích), takže rozložení kopie se nerozpadne. Zakryjí se i formulářová pole (také skrytá), odkazy mailto: a tel: a atributy title, alt a data-*. Atributy, které nikdo nevidí, třeba data-*, se změní jen v uloženém HTML: samotná stránka si ničeho nevšimne.

Nechat je najít Kaloko

pii: redact ve scénáři zapne vyhledávání ve všech krocích. Najde e?mailové adresy, telefonní čísla (s +420 a 00420, a české 603 123 456 vedle slova jako Telefon, Mobil nebo Kontakt), rodná čísla (s kontrolním součtem; starý devítimístný tvar a čísla bez lomítka jen vedle popisku jako RČ), IBAN a čísla účtů, data narození u popisku, adresy s PSČ a jména v oslovení ("Dobrý den, Jano,“). Částky jako 250 000 000 Kč a čísla objednávek nechá být. Formulářová pole pozná podle typu, autocomplete a názvu. Počítají se i údaje v adresách URL, také zakódované (jan%40firma.cz). Pro jeden krok udělá totéž capture: { mask: auto }.

Vyhledávání čte i text v otevřeném shadow DOM a v rámech ze stejného originu. Uzavřené shadow rooty, rámy z jiného originu a text vykreslený do obrázků nebo canvasu přečíst neumí: takové prvky zamaskujte selektorem.

Údaje, které jako osobní jen vypadají, třeba vaši zákaznickou linku, zapište do pii.allow v kaloko.config.yml. Hvězdička * znamená jakýkoli text:

```
pii:
allow: ["info@acme.cz", "*@support.acme.cz", "+420 800 100 200"]
```

Smyšlené údaje místo obdélníků

pii: pseudonymize nahradí osobní údaje vymyšlenými: eva.svobodova42@example.com, +420 000 731 205, Petr Novák. Stejný údaj má stejnou náhradu v celém běhu i v dalších běžích scénáře, takže se screenshoty dají porovnat pixel po pixelu a schválení zůstane platit. Dva různé údaje nikdy nedostanou stejnou náhradu. Obrazovky zůstanou pro klienta čitelné a pořád z nich poznáte, že se jméno z formuláře propsalo do rekapitulace i do potvrzovacího e?mailu. Smyšlená telefonní čísla mají za předvolbou 000 a smyšlená rodná čísla měsíc 00, aby je nikdo nepovažoval za skutečná.

Náhradu vybírá otisk (hash) údaje, scénáře a projektu. Zůstane v něm z údaje tak málo, že se z náhrady zpátky odvodit nedá. Pokud chcete do výpočtu přidat vlastní tajemství, nastavte KALOKO_PII_SALT (všude stejné, jinak se náhrady na vašem počítači a v CI budou lišit).

Údaje zamaskované v jednom kroku se nahradí i v dalších, také když další krok běží v novém procesu kaloko: běh si je drží v souboru .pii-known.json ve své složce, který zůstává na vašem počítači. Krátké údaje se dál nepřenášejí, takže zamaskované 2024 nebo Praha se jinde v běhu nezmění na zástupný text.

E?maily

Příjemci zachyceného e?mailu (Komu a Kopie) se skryjí vždy, když krok něco maskuje. Oslovení a údaje plátce zamaskujte selektorem u e?mailového kroku (capture: { mask: [".greeting", ".payer"] }), nebo nechte zapnuté vyhledávání. E?mail jen s textovou částí pokryje vyhledávání a údaje, které se v běhu zamaskovaly dřív.

Vyžadovat to u prostředí

```
environments:
production:
base_url: https://shop.acme.cz
pii: required
```

S pii: required vytvoří kaloko start běh jen tehdy, když scénář říká pii: redact nebo pseudonymize, nebo když běh spustíte s --pii redact. pii: redact nebo pii: pseudonymize přímo u prostředí zapne skrývání pro každý běh v něm.

Kontrola před sdílením

kaloko share projde, co se chystá nahrát. V běhu, který skrývá osobní údaje (pii u běhu, scénáře nebo prostředí, nebo mask: auto), nahrávání zastaví nezamaskovaný e?mail, telefon, rodné číslo nebo IBAN a Kaloko vypíše krok a prvek:

```
Not shared: the run holds 1 unmasked personal data value(s):
summary (cs, desktop) <p.contact> email ja….cz
```

Údaj zamaskujte a krok zachyťte znovu, nebo ho přidejte do pii.allow, pokud osobní není. kaloko share --allow-pii nahraje běh i tak. Běh, který osobní údaje neskrývá, se nahraje a stejný seznam vypíše jako upozornění.

Kontakty samotného webu se nepočítají: e?maily na doméně prostředí, e?maily a telefony v patičce stránky a kontakty Organization nebo ContactPoint v JSON-LD. Nepočítají se ani ukázkové adresy jako you@company.com, text v placeholderech a čísla v kresbách SVG. Pokud chcete kontakty webu kontrolovat jako cokoli jiného, nastavte pii: { site_contacts: false } v kaloko.config.yml. Screenshoty a videa se nečtou: ty chrání maska.

Na plachtě

Běh, který skrývá osobní údaje, to říká vedle názvu prostředí ("Osobní údaje skryty“). V detailu kroku jsou zamaskovaná místa orámovaná fialově a poznámka vysvětlí, že tmavý pruhovaný obdélník je záměr, ne chyba aplikace.

Kontrolovat stránky podle plánu (hostované běhy)

Marketingový web se mění několikrát týdně a nic z toho neprochází vašimi pull requesty. Hostovaný běh otevře stránky scénáře jen pro čtení v prohlížečích Kaloka každý den nebo týden, pořídí screenshoty celých stránek ve všech jazycích a velikostech obrazovky, zkontroluje, co se dá ze stránky přečíst, a nasdílí běh pod vaším jménem. CLI nemusí nikdo spouštět. Hostované běhy patří k plánu Business (2 000 načtených stránek měsíčně) a Enterprise (10 000).

1. Připravte scénář jen pro čtení

Kroky scénáře otevírají stránky. Začít můžete přímo od stránek:

```
npx kaloko draft https://example.com/ https://example.com/pricing
```

2. Naplánujte ho

```
npx kaloko schedule add --scenario qa/flows/public.yml --env production --every day --at 5
npx kaloko schedule add --scenario qa/flows/public.yml --env production --every week
--weekday mon
```

--at je hodina v UTC. --every manual vytvoří plán, který poběží, jen když ho spustíte.

Fungují i stránky za přihlášením. Přihlašovací údaje HTTP Basic z prostředí se pošlou jednou a uloží zašifrované. Pro stránky přihlášeného uživatele si člověk uloží relaci přes npx kaloko auth save --env production --account member a plán ji použije s --account member.

3. Spravujte ho

```
npx kaloko schedule list # příští a poslední běh, stránky využité tento měsíc
npx kaloko schedule run <id> # spustit hned
npx kaloko schedule pause <id>
npx kaloko schedule resume <id>
npx kaloko schedule remove <id>
```

Stejné plány najdete na kaloko.app v Nastavení -> Integrace, i s dalším spuštěním a s tím, kolik výsledků kondice poslední běh našel nových a opravených.

Spustit po nasazení

Plán se může spustit i ve chvíli, kdy se web nasadí. Vytvořte mu adresu pro spuštění a tajemství:

```
npx kaloko schedule hook <id> # vypíše adresu a tajemství, jen jednou
```

nebo v Nastavení -> Integrace -> Běhy na serveru Kaloka -> Spustit po nasazení. Tajemství uložte do tajných proměnných CI nebo hostingu, nikdy do repozitáře. Další volání příkazu vytvoří nové tajemství a to staré přestane platit; --remove spouštění odebere.

Každé volání s tajemstvím spustí běh v nejbližší čtvrthodině. Volání, která přijdou, než běh začne, jsou jeden běh; volání během průchodu dostane vlastní běh po něm. Jeden plán přijme nejvýš 30 volání za hodinu a každé je v auditním logu.

Z jakéhokoli jobu v CI, po kroku s nasazením:

```
curl -fsS -X POST "$KALOKO_HOOK_URL" -H "Authorization: Bearer $KALOKO_HOOK_SECRET"
```

Vercel. Vercel hlásí každé nasazení do GitHubu, takže krátké workflow zavolá spouštění, jakmile je produkce nahoře. Přidejte do repozitáře tajné proměnné KALOKO_HOOK_URL a KALOKO_HOOK_SECRET a commitněte .github/workflows/kaloko-after-deploy.yml:

```
name: Kaloko after deploy
on: deployment_status
jobs:
kaloko:
```

```
if: github.event.deployment_status.state == 'success' && github.event.deployment.environment
== 'Production'
runs-on: ubuntu-latest
steps:
- run: curl -fsS -X POST "$KALOKO_HOOK_URL" -H "Authorization: Bearer $KALOKO_HOOK_SECRET"
env:
KALOKO_HOOK_URL: ${{ secrets.KALOKO_HOOK_URL }}
KALOKO_HOOK_SECRET: ${{ secrets.KALOKO_HOOK_SECRET }}
```

Netlify. Site configuration -> Notifications -> Deploy notifications -> Add notification -> Outgoing webhook. Událost "Deploy succeeded“, jako URL adresa pro spuštění, tajemství jako "JWS secret token“. Kaloko podpis od Netlify ověří.

Vlastní odesílatel. Místo tokenu můžete podepsat tělo požadavku: pošlete X-Kaloko-Signature: sha256=<hex>, HMAC-SHA256 těla s tajemstvím jako klíčem.

Přečtěte si výsledek

Běh se objeví mezi ostatními, porovnaný s referencí scénáře, se stejnou plachtou, komentáři i schvalováním. Kontroly selektorů, textů, titulků a meta tagů se vyhodnotí. Kritéria, která potřebují plný průchod, zůstanou "nehodnoceno“ i s důvodem.

Kontroly z packů (SEO, přístupnost) patří do kondice běhu, ne do jeho verdiktu. Když má běh proti minulému novou chybu v kondici, plán to oznámí sledujícím scénáře, kanálům, které odebírají "Zhoršená kondice“, a webhooku health.worse. U každého plánu si vyberete, jestli hlásit nové chyby (výchozí), i nová varování, nebo nic: v Nastavení -> Integrace -> Běhy na serveru Kaloka, nebo volbou --health-alert error|warning|off u kaloko schedule add. Víc v návodu Kontrola kondice aplikace.

Stack kroky: mnoho stránek jedné šablony

Stack krok zastupuje celou rodinu stránek, třeba všechny produkty. Hostovaný běh vezme jeho stránky z items.urls, items.sitemap nebo items.robots (i zabalené sitemapy a indexy sitemap). match, exclude, limit a rotate: week fungují stejně jako v kaloko walk, takže oba ve stejném týdnu vyberou stejné stránky.

Každou stránku otevře jednou, v první velikosti obrazovky kroku, a přečte ji bez screenshotu. Kontroly, kterým stačí prvky stránky (selektory, texty, titulky, meta značky), rozhodne; indexová pole titulek, H1, popis, počet slov a typy JSON-LD vyplní; u všeho ostatního napíše "nevyhodnoceno“ i s důvodem. Stavový kód HTTP zůstane prázdný. Na plachtě je krok stack s indexem, stejně jako po průchodu.

Velký stack se čte během několika čtvrthodinových kol, v každém nejvýš 150 stránek. kaloko schedule list a Nastavení -> Integrace ukazují, kolik už je přečteno, a běh se nasdílí, až jsou přečtené všechny stránky. Než doběhne, Spustit teď, spouštěč po nasazení i uložená změna počkají a přijdou na řadu po něm; pozastavení plánu ho ukončí. Jeden běh přečte nejvýš 1 000 stránek stacků a ze sitemap nejvýš 5 × limit vyhovujících adres pro každý jazyk (nejvýš 20 000). Když kolo selže nebo se nikam nedostane (server se sitemapou neodpovídá), další kolo to zkusí znovu; po třech takových kolech za sebou se běh vzdá a plán počká na další termín.

Sitemapy můžou ležet na jiném serveru, třeba na CDN. Kaloko je stahuje bez hesla a cookies prostředí, a jen když se jméno překládá na veřejné adresy, a to i po každém přesměrování. Stránky otevírá jen na adresách prostředí; ostatní ze sitemapy vynechá a napíše to. match a exclude můžou mít nejvýš 300 znaků, bez zpětných odkazů a lookaroundu.

Stack s items.variants: hreflang seskupí jazykové verze stránky stejně jako kaloko walk: seznam, výřez na tento týden i id se berou ze source_locale a v ostatních jazycích se otevře hreflang alternativa každé stránky. Před stránkami si běh u každé zdrojové stránky přečte hlavu stránky (<head>) kvůli hreflang. Každé takové čtení se počítá jako jedna stránka, do měsíčního limitu i do 150 stránek jednoho kola, takže stack se 100 produkty ve třech jazycích spotřebuje 400 stránek: 100 čtení a 300 stránek. Stránka bez překladu se v indexu stacku označí a započítá do HC1005, stejně jako po průchodu. Čtení zůstává na adresách prostředí, i po přesměrování, a nese tam heslo nebo cookies plánu, tytéž, které dostane hostovaný prohlížeč. Když hreflang míří na produkci a běh prochází staging, otevře se cesta každé alternativy na stagingové adrese daného jazyka. Když zdrojová stránka

přesměruje mimo prostředí, běh za ní nejde: na rozdíl od kaloko walk ji nahlásí jako chybějící překlad (cannot read the page).

Jedna adresa pro každý jazyk

Když má prostředí base_url pro každý jazyk zvlášť, hostovaný běh ji vybere stejně jako kaloko walk: vlastní adresu jazyka, jinak default.

```
environments:
production:
base_url: { default: "https://www.example.com", cs: "https://www.example.cz/cs" }
readonly: true
```

Co hostované běhy neumějí

Stránky jen otevřou a přečtou. Nic neklikají, nepíšou, neodesílají ani nevytvářejí, takže kroky, které dělají víc než otevření stránky, vynechají a vyjmenují. Plný průchod potřebují i nativní aplikace, místní nebo neveřejné adresy, přihlašovací údaje posílané v hlavičkách a stack kroky, které berou stránky z crawl, script nebo file. Odkazy do CMS z items.resolve hostované běhy nevyplní.

Pro ně plán vyexportujte jako workflow pro GitHub Actions, které scénář projde s plným CLI:

```
npx kaloko schedule export github --scenario qa/flows/checkout.yml --env staging --every week
```

Workflow vypíše, které tajné proměnné potřebuje.

Jak se stránky počítají

Počítá se každá načtená stránka: každý krok v každém jazyce a velikosti obrazovky. Scénář s 10 stránkami ve 2 jazycích na desktopu a mobilu načte za jeden běh 40 stránek, takže denní běh spotřebuje zhruba 1 200 stránek měsíčně.

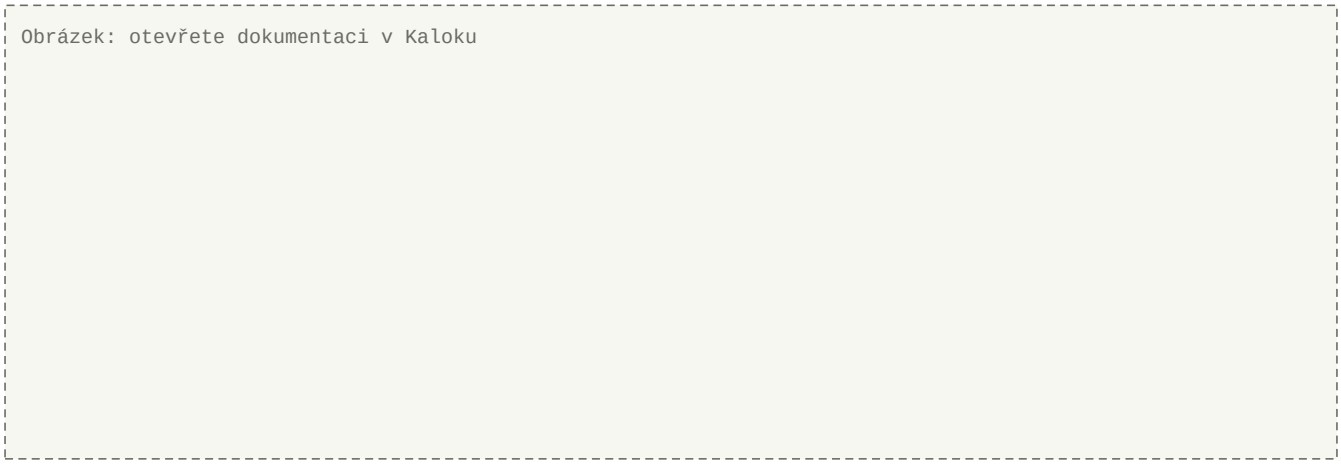
Každá stránka stacku se počítá jednou, když se přečte. Týdenní stack se 100 stránkami ve 3 jazycích spotřebuje zhruba 1 300 stránek měsíčně. Když měsíční rozpočet dojde uprostřed stacku, běh až den počká, jestli nějaký přibude (nový měsíc, vyšší limit), pak nasdílí stránky přečtené do té doby a u ostatních napíše, že je nepřečetl.

Pozvat lidi a hosty

Běh k něčemu je, až když ho uvidí ti, kdo rozhodují. Přiveďte kolegy do organizace, pozvěte klienta do jednoho projektu, nebo nechte číst každého z firemní domény. Vieweři jsou v každém plánu zdarma, takže nemusíte šetřit tím, kdo se smí dívat.

Pozvěte kolegu do organizace

1. Otevřete Nastavení -> Lidé a zvolte Pozvat do organizace.



Členové v Nastavení · cs · desktop

2. Zadejte e-mail a vyberte roli. Viewer čte, komentuje a schvaluje; ostatní role popisuje stránka Role a API tokeny.

3. Člověk dostane e-mail s odkazem a přes něj se k organizaci připojí.

Seznam v Nastavení ukazuje každého člena s rolí a s tím, kdy byl naposledy vidět, a nepřijaté pozvánky s tím, kdo je poslal a kdy vyprší.

Když pozvánku nejde doručit, u nepřijaté pozvánky stojí Adresa nefunguje: zkontrolujte, jestli je správně napsaná a ten, kdo ji poslal, dostane položku do inboxu. Znovu ji Kaloko neposílá: zrušte ji a pozvěte správnou adresu.

Pozvěte někoho do jednoho projektu

Klienti, externisté i kolegové z jiných týmů často potřebují jeden projekt, ne celou organizaci.

1. Otevřete projekt a zvolte Sdílet.

2. Napište e-mailovou adresu, klidně i mimo vaši organizaci, a vyberte roli.

3. Zvolte Pozvat. Pozvánka platí 14 dní.

Adresa mimo organizaci přijde jako host. Host vidí jen projekty, do kterých byl pozván: ne ostatní projekty, ne lidi z organizace, ne nastavení ani fakturaci. Dialog Sdílet také ukáže, kdo už přístup má a jestli je projekt otevřený všem členům, nebo omezený jen na uvedené lidi.

V plánu Business můžete hostovi nastavit datum, kdy přístup skončí; ten den projekt opustí a přijde mu e-mail. Hosté s rolí designer nebo vývojář / tester patří k plánu Enterprise; v ostatních plánech je host viewer.

Pusťte dovnitř firemní doménu

Místo zvaní kolegů po jednom přidejte firemní doménu v Nastavení -> Lidé -> Domény pro přihlášení. Kontrolu nad ní prokážete TXT záznamem, který Kaloko ukáže; dokud doména není ověřená, nikdo se přes ni nepřipojí. Pak se každý s e-mailem na ověřené doméně přihlásí s výchozí rolí domény (viewer, pokud nezvolíte jinou). Domény veřejných schránek, třeba gmail.com, si přivlastnit nejde.

Čtenáři z vaší domény

V plánu Business přepínač Čtenáři z naší domény na stejném místě pustí každého, kdo se přihlásí z ověřené domény, ke changelogu a dokumentaci všech produktů, i když je projekt omezený. Vstoupí jako vieweři a místo nezabírají. Víc na stránce Publikovat changelog s obrázky.

Kdo zabírá místo

Vieweři nikdy. Designer, vývojář / tester a admin zabírají jedno místo, jednou za celou organizaci, ať jsou to členové, nebo hosté. Pozvánka s takovou rolí drží místo, dokud ji člověk nepřijme nebo ji nezrušíte.

Publikovat changelog s obrázky

Vaše přijaté běhy už obsahují každou obrazovku vydání. Kaloko z CHANGELOG.md ve vašem repozitáři udělá stránku s obrázky před a po, kterou si přečtou klienti, podpora i marketing. Takhle vzniká i changelog samotného Kaloka a je veřejný, poznámky k vydání v něm píšeme anglicky: kaloko.app/p/sinfin/kaloko/changelog.

Jednou to nastavte

```
npx kaloko init --docs=changelog
```

Přidá CHANGELOG.md (Keep a Changelog) a blok changelogu do kaloko.config.yml. Repozitář s několika aplikacemi je vypíše pod products:; npx kaloko products sync je dá do Kaloka ještě před prvním vydáním.

Dejte do záznamu obrázky

Obrázek odkazuje na krok místo na soubor:

```
- Ceník srovnává všechny plány v jedné tabulce.
![Předtím](kaloko:KALOKO-public/pricing?version=prev) ![Potom](kaloko:KALOKO-public/pricing)
```

KALOKO-public je id scénáře a pricing id kroku. Bez ?version= odkaz ukáže nejnovější přijatou obrazovku; ?version=prev tu z předchozího vydání.

Připravte koncept záznamu

```
npx kaloko changelog draft # vypíše záznam Unreleased
npx kaloko changelog draft --apply # zapíše ho do CHANGELOG.md
```

Koncept posbírá pull requesty a commity od posledního vydání, roztřídí je do Added, Changed, Fixed a Removed a přidá obrázky před a po u obrazovek, které přijaté běhy ukazují jinak. Řádky pak přepište pro lidi: co teď můžou udělat, jedna změna na řádek. --since <tag> začne od jiného vydání.

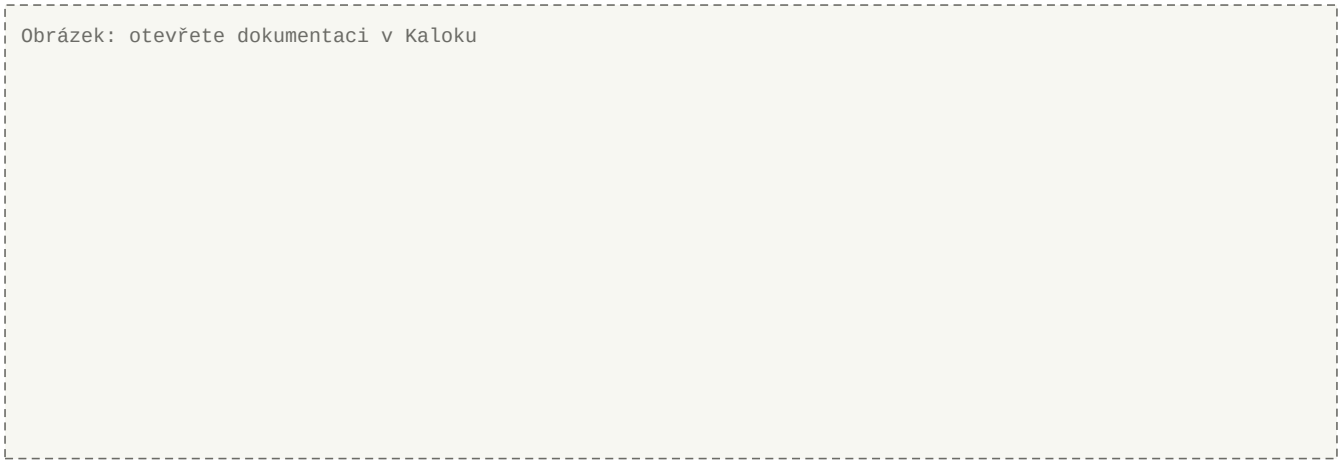
Zapište vydání

```
npx kaloko release 2.4.0
```

Z Unreleased se stane 2.4.0, vydání se zapíše i s přijatými běhy, které dávají obrázky, a příkaz vypíše adresu stránky. Bez čísla verze si ji Kaloko přečte z gitových tagů, package.json, release-please nebo changesets.

Když vydání dělá release-please, changesets nebo semantic-release, píše CHANGELOG.md dál ten nástroj a Kaloko ho jen čte. npx kaloko ci github --release přidá workflow, které zapíše každé zveřejněné vydání na GitHubu.

Kdo ho čte



Stránka Produkty · cs · desktop

Changelog najdete v aplikaci pod Produkty. Čte ho každý, kdo vidí některý z projektů produktu, i hosté. V plánech Business a Enterprise může admin těch projektů přímo na stránce zvolit, kdo ho čte dál:

• Čtenáři z naší domény: každý, kdo se přihlásí z ověřené domény vaší organizace.

• Kdokoli s odkazem: veřejná stránka s odběrem Atom. Vyhledávače ji indexují, jen když to povolíte.

Čtenáři si na stránce changelogu můžou zapnout i e-mail "Co je nového“ (od plánu Business): každé nové vydání jim přijde s obrázky v jejich jazyce.

Jak stránku používají čtenáři: Co je nového. Návody stejně aktuální udržíte podle stránky Publikovat živou dokumentaci.

Publikovat živou dokumentaci a hlídat zastaralé části

Návody ve vašem repozitáři můžou místo vložených screenshotů ukazovat obrazovky z přijatých běhů, takže obrázky nikdy nezestárnou. Když se obrazovka po napsání textu změní, Kaloko řekne, kterou část zkontrolovat. Stránky, které právě čtete, vznikají přesně takhle: kaloko.app/p/sinfin/kaloko/docs.

Připravte složku

```
npx kaloko init --docs=full
```

Vytvoří docs/ se čtyřmi složkami podle Diátaxis, v každé šablonu:

| Složka | Čtenář chce | |---|---| | tutorials/ | naučit se to od začátku do konce | | how-to/ | udělat jeden konkrétní úkol | | reference/ | dohledat fakt | | explanation/ | pochopit proč |

index.md je úvodní stránka. Front matter je nepovinný: title (jinak první nadpis) a order (pořadí v postranním menu). Stránky propojujte relativními odkazy na .md.

Napište stránku

Obrázky jsou odkazy na kroky, stejně jako v changelogu:

```
1. V horní liště zvolte **Přihlásit**.
![Přihlašovací stránka](kaloko:KALOKO-public/login)
```

Překlad leží vedle originálu: pay.cs.md vedle pay.md, jazyky vyjmenuje konfigurace (docs.languages: [en, cs], jazyk originálů první). Obrázky se řídí jazykem textu.

Některé věci je lepší vidět než číst: přetažení, dialog, který se otevře, přepnutí jazyka. Když má přijatý běh záznam kroku (kaloko walk --record), je na obrázku tlačítko pro přehrání. Kliknutí nebo klepnutí záznam přehraje bez zvuku a na konci se vrátí obrázek. Samo se nic nepřehrává. Přidejte ?media=video, když krok svůj záznam potřebuje (kaloko docs check upozorní, když ho běh nemá), nebo ?media=image, když má zůstat jen obrázek:

```
![Přeložit kategorii](kaloko:DOC-konzole/category-edit?media=video)
```

Záznamy, které vaše dokumentace nebo changelog přehrávají, zůstanou, dokud je přehrávají, a počítají se do úložiště.

Ukažte jen část obrazovky

Když text mluví o jednom tlačítku, celá obrazovka je často zbytečně velká. Části, které návod potřebuje, pojmenujte ve scénáři u kroku pod details:. Každý běh je pak vyfotí ve všech jazycích a velikostech obrazovky a osobní údaje v nich skryje stejně jako na snímku:

```
- id: cart
title: Košík
path: /cart
details:
- id: add-to-cart
name: { en: Add to cart button, cs: Tlačítko Přidat do košíku }
selector: "#add"
- id: summary
name: Souhrn objednávky
selector: .summary .total
context: section
```

Kaloko ukáže prvek i s okolím a obtáhne ho rámečkem. Bez context je okolím nejbližší sekce nebo karta, která má aspoň 480 × 240 px (na telefonu méně). context: card, context: section, context: page (jedna obrazovka kolem prvku) nebo CSS selektor nadřazeného prvku ho určí přesně a padding přidá okraj (výchozí 16 px). Dlouhou sekci Kaloko ořízne na jednu obrazovku kolem prvku. states: [error] vyfotí detail jen v některých stavech kroku.

V textu přidejte id za #:

```
2. Zvolte **Přidat do košíku**.
![Přidat do košíku](kaloko:shop/cart#add-to-cart)
```

Ostatní parametry fungují dál (kaloko:shop/cart?locale=cs#add-to-cart). Na plachtě najdete detaily kroku v Přílohách kroku na kartě Stránka, každý i s odkazem. Dokud detail není v žádném přijatém běhu, čtenáři vidí celou obrazovku a kaloko docs check upozorní.

Když má každá položka seznamu jednu akci a obrázek hned pod sebou, dá se stránka projít i jako průvodce: čtenář zvolí Projít krok za krokem, vidí vždy jeden krok a může si odškrtnout Mám vyzkoušeno.

Publikujte

```
npx kaloko docs publish # dokumentace pro latest
npx kaloko docs publish --version 2.4.0 # dokumentace vydání (Business)
```

Čtenáři dostanou výběr verze, přepínač jazyka a obrazovky a hledání a každý obrázek otevře svůj krok na plachtě. kaloko release publikuje složku s dokumentací pro svou verzi sám, pokud nepřidáte --no-docs. Kdo dokumentaci čte, se řídí stejnými pravidly jako changelog: viz Publikovat changelog s obrázky.

Hlídejte zastaralé části

Při publikování si Kaloko zapamatuje obrazovku za každým odkazem. Když pozdější přijatý běh ukáže jinou obrazovku, uvidí lidé, kteří dokumentaci píšou, u části poznámku "Obrazovka se změnila ve verzi 2.4 — zkontrolujte text.“ (Business).

```
npx kaloko docs check # proti publikované dokumentaci; potřebuje token
npx kaloko docs check --offline # jen lokální soubory
```

Chyby: rozbité odkazy, kroky, které scénáře nemají, odkazy bez přijatého obrázku a zastaralé části. Varování: chybějící nebo neaktuální překlady, položky changelogu, které lidé uvidí, ale nemají obrázek, a odkazy s media=video, jejichž přijatý běh záznam kroku nemá, a odkazy na #detail, který scénář nemá nebo který zatím žádný přijatý běh nevyfotil. Při chybě příkaz skončí kódem 1; s --strict i při varování, což se hodí do CI.

U každé zastaralé části otevřete krok, porovnejte ho s textem a přepište, co už nesedí; poznámka zmizí při dalším kaloko docs publish. Pokud text pořád sedí, kaloko docs publish --reviewed poznámky smaže beze změny textu. Použijte ho, až když jste je přečetli.

Jak dokumentaci dostat ven jako soubory pro jiný web: Exportovat dokumentaci a changelog.

Co je nového

Každé vydání produktu má v novinkách záznam psaný pro lidi, kteří produkt používají, s obrazovkami, které změnilo. Obrázky jsou ze schválených běhů, takže ukazují to, co někdo schválil.

Kde novinky najdete

1. V hlavní nabídce otevřete Produkty a u produktu zvolte Novinky, nebo otevřete odkaz, který vám někdo poslal. Každý produkt má své novinky a nejnovější vydání je nahoře.

2. Verze v horní části vás přenesou na své vydání. Pod každou verzí je, co se v ní změnilo, rozdělené na Nové, Změny, Opravy a Odebráno.

Porovnejte předtím a potom

1. U změněné obrazovky leží obrázek předtím a potom přes sebe. Posuvníkem posunete předěl.

2. Vedle sebe dá oba obrázky vedle sebe (na telefonu pod sebe).

3. Jazyk a Obrazovka přepnou všechny obrázky do jiného jazyka nebo velikosti obrazovky, pokud je běh zachytil.

4. Otevřít flow na plachtě ukáže celé flow, pokud máte k běhu přístup.

Buďte v obraze

• Posílat e-mail o novinkách: ke každému novému vydání vám přijde jeho záznam s obrázky ve vašem jazyce. Odkaz na konci každého e-mailu je zase vypne.

• Stáhnout nabídne novinky jako JSON, Markdown, HTML pro offline čtení nebo PDF, třeba pro poznámky k vydání nebo zprávu pro klienta.

• Veřejné novinky mají i odběr pro čtečky.

Natočit video průchodu

Screenshot ukáže, kde krok skončil. Video ukáže, jak se tam dostal: točící se kolečko, které se nezastavilo, menu, které se otevřelo dvakrát. Nahrávejte, když screenshot neukáže, co má kontrolující posoudit: animace, načítání, přetahování, krok, který jednou projde a jednou ne, nebo ukázku pro klienta.

Nahrajte průchod

```
npx kaloko walk --record # každý krok od předchozí obrazovky po jeho snímek
npx kaloko walk --record flow # celý průchod pro každý jazyk, kapitola na krok
npx kaloko walk --record failed # jen kroky, kterým selže skript nebo kontroly
npx kaloko walk --record --record-quality low --steps checkout
```

Pokud chcete nahrávat vždy, nastavte to ve scénáři nebo v kaloko.config.yml:

```
capture:
record: { mode: failed, quality: low, max_seconds: 45 }
nodes:
- id: checkout
record: true # tento krok vždy; record: false nikdy
```

Kvalita je low, medium (výchozí) nebo high. Videa jsou mnohem větší než screenshoty, takže nenahrávejte každý průchod.

Průběh má každý webový krok

Každý prošlý webový krok si nechá průběh: kliknutí a psaní, síťové požadavky se statusem a časem a zprávy z konzole. Je malý a zapnutý ve výchozím stavu; vypne ho --no-trace.

```
npx kaloko video list # co se nahrálo a zaznamenalo, i s velikostí
npx kaloko video trace --step checkout # průběh kroku jako časová osa v terminálu
```

Na plachtě najdete v detailu kroku vedle snímku přepínač Video a na záložce Technické průběh. Klikněte na řádek a video na to místo skočí. Videa jdou jedno za druhým: když jedno pustíte, po jeho konci se samo spustí video dalšího kroku. Stejně se samo posouvá i přehrání běhu s videem. Krok, o kterém máte rozhodnout, na vás počká, a krok bez videa ukáže na tři sekundy svůj screenshot.

Udělejte z běhu jedno video

```
npx kaloko video export --format mp4 # nebo webm, gif
npx kaloko video export --locale cs --steps cart,payment --lang cs --out checkout.mp4
```

Export seřadí videa podle flow a před každý krok dá úvodní kartu s jeho názvem a verdiktem. Takový soubor pošlete klientovi nebo přiložíte k poznámkám k vydání.

Potřebné nástroje

Videa z webu a z Electronu potřebují jen Chrome. S nainstalovaným ffmpeg dostanete i MP4 pro Safari, videa aplikací (Android, iOS Simulator, macOS) a export celého průchodu. npx kaloko video tools vypíše, co na počítači je. Průchody ve WebKitu a Firefoxu se nenahrávají a průběh se u nich nezaznamenává.

Sdílení

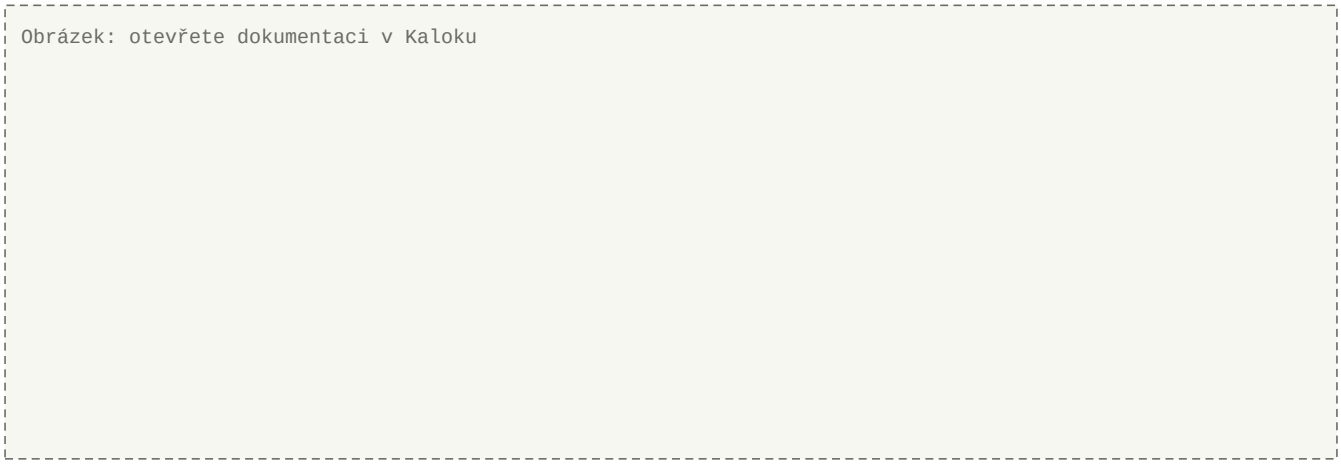
Nahrávat můžete v každém plánu. kaloko share nahraje videa spolu s během v plánech Business a Enterprise; ve Free a Team zůstanou v místním náhledu a sdílí se screenshoty a průběhy. Sdílená videa se uchovávají 30 dní (Enterprise si dobu nastaví v Nastavení -> Úložiště). Video větší než 25 MB zůstane jen u vás: snižte kvalitu nebo max_seconds.

Projít běh

Běh je jeden průchod agenta flow: každá obrazovka, na kterou se dostal, ve všech jazycích a velikostech obrazovky, se vším, co zkontroloval. Projdete ho v prohlížeči, nic se neinstaluje.

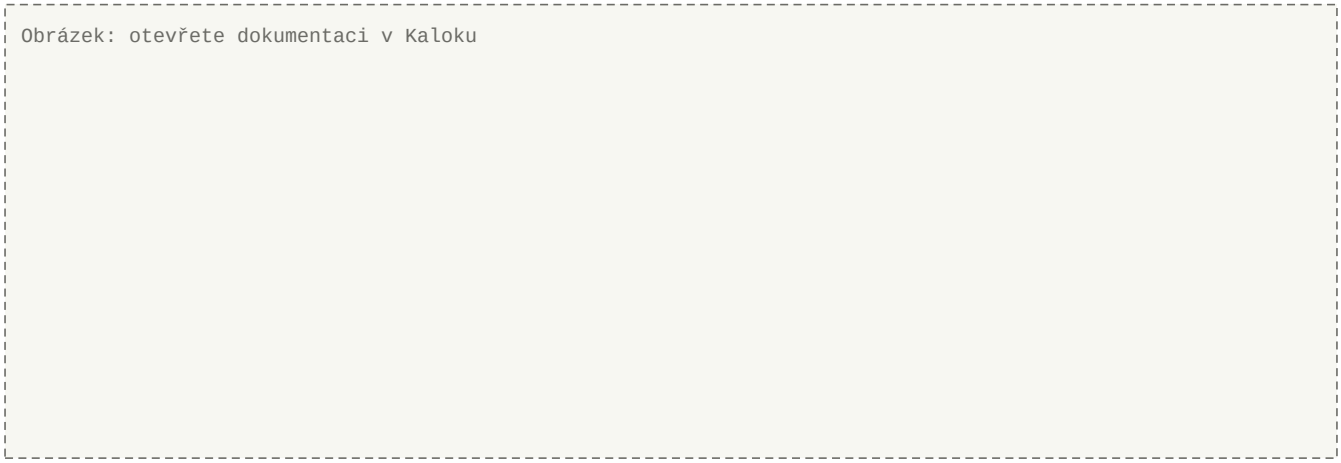
Otevřete běh

1. Otevřete odkaz z pull requestu, z issue nebo z e-mailu. Běh se otevře na plachtě: všechny kroky flow jako mapa se šipkami, které ukazují cestu mezi nimi.



Mapa běhu · cs · desktop

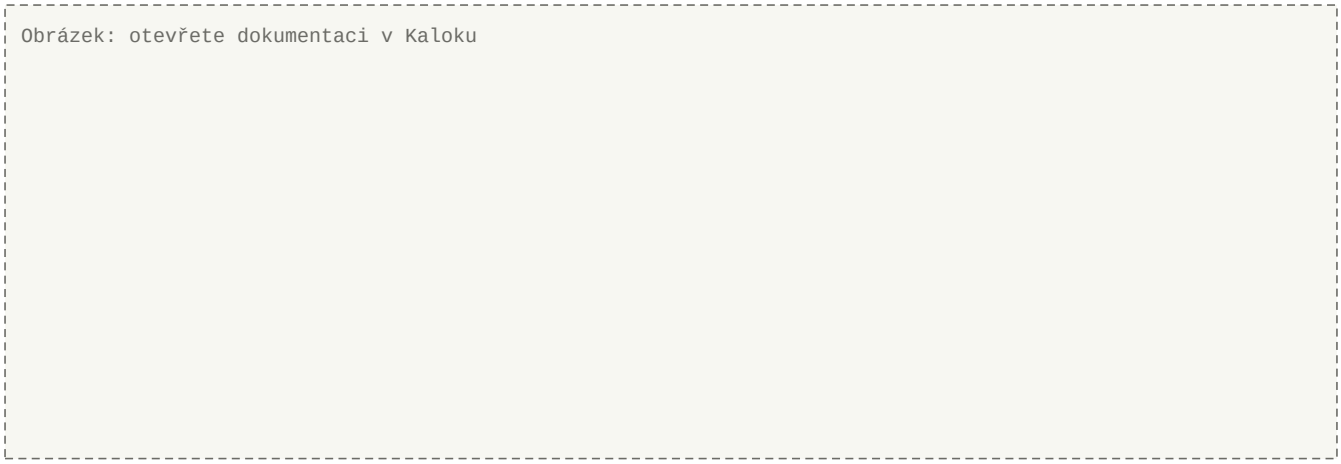
2. Pokud vám víc vyhovuje seznam, stiskněte I. Stejný běh se ukáže jako seznam kroků s výsledky a dá se v něm hledat i filtrovat podle stavu.



Běh jako seznam · cs · desktop

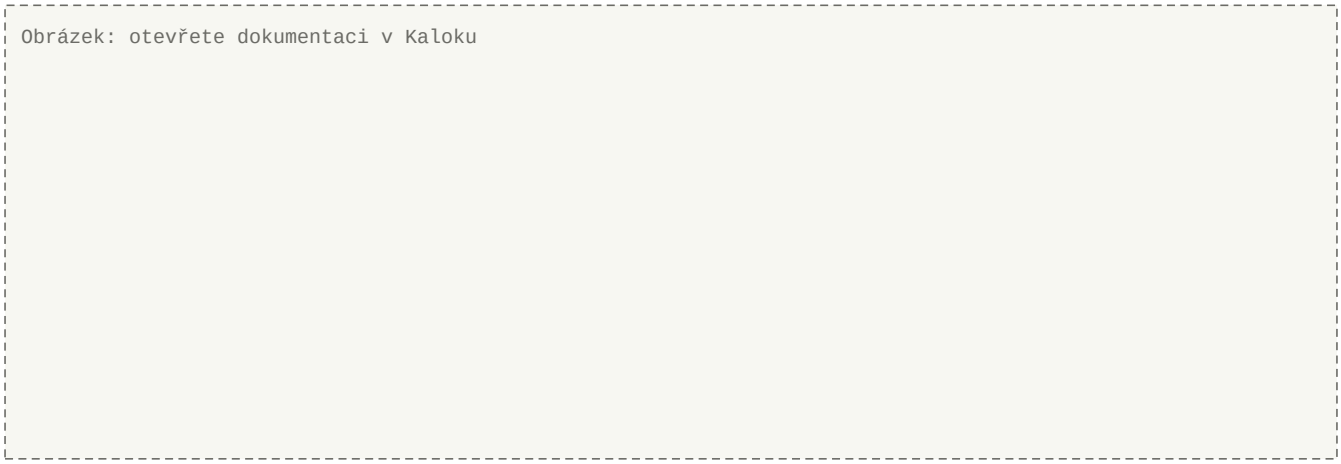
Podívejte se na krok

1. Klikněte na krok. V detailu uvidíte snímek obrazovky, kritéria s tím, co se zjistilo, a komentáře, které k němu lidé napsali. Šipkami <- a -> přejdete na předchozí a další krok.



Krok s komentáři · cs · desktop

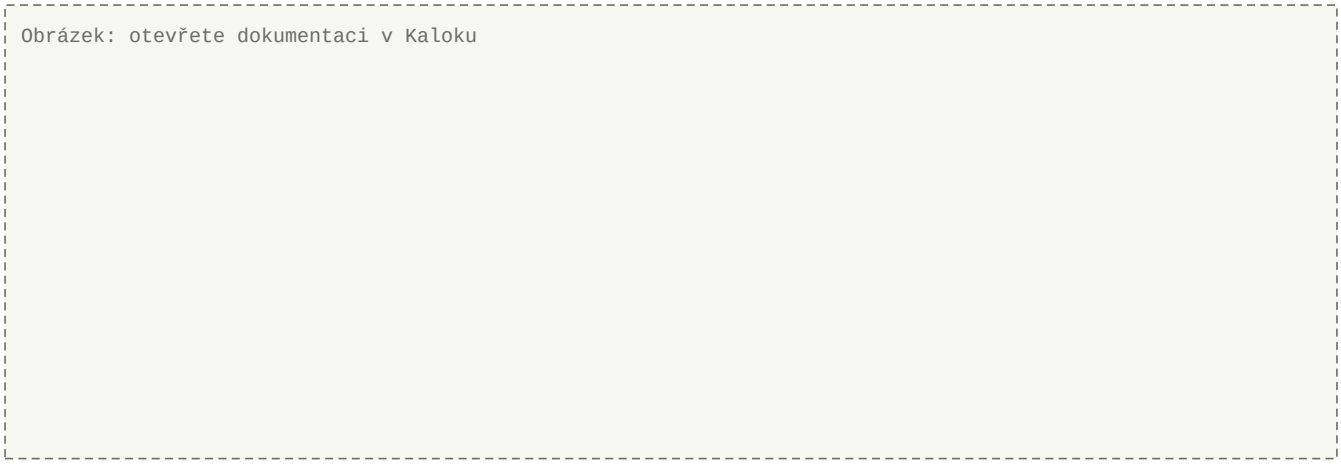
2. Něco nesedí? Napište ke kroku komentář. Všechny otevřené komentáře běhu najdete v panelu komentářů, takže se mezi kroky nic neztratí.



Panel komentářů · cs · desktop

Projděte ho jako uživatel

Klávesou R spustíte průchod během: jedna obrazovka po druhé, v pořadí, v jakém je uvidí člověk. Tak nejrychleji ukážete flow někomu, kdo plachtu ještě neviděl.



Průchod během · cs · desktop

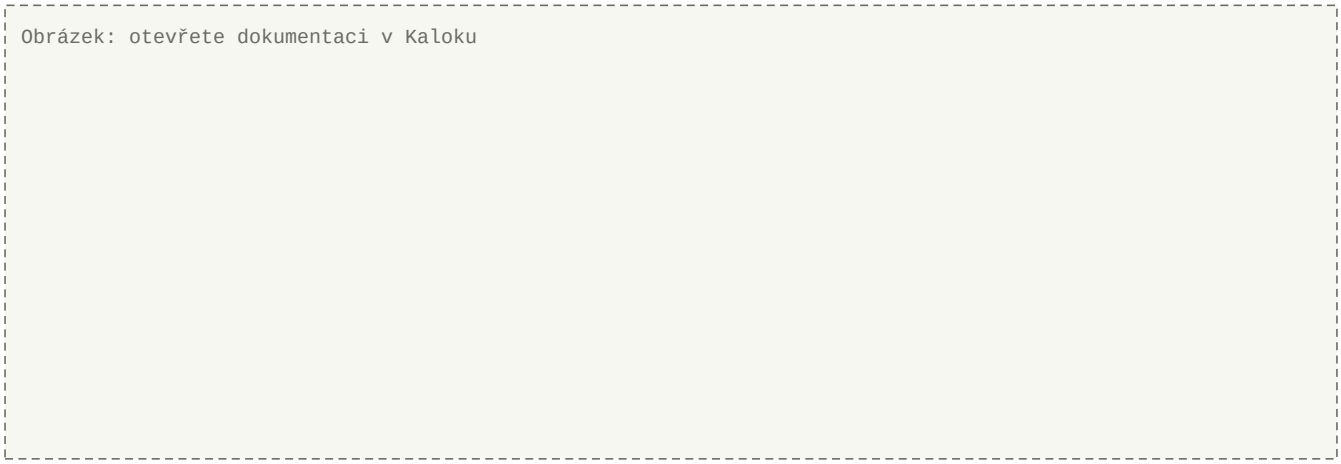
Až budete vědět, co si myslíte, schvalte nebo vraťte běh.

Kontrolovat jen to, co se změnilo

Režim kontroly ukazuje jeden krok po druhém a začíná těmi, které chtějí člověka: nesplněné a vrácené kroky, kritéria, kterými si AI nebyla jistá, kroky změněné od posledního přijatého běhu a kroky s otevřenými komentáři. Kroky, které se nezměnily, si ponechají schválení, které jim někdo dal dřív.

Otevřete režim kontroly

1. Otevřete odkaz na kontrolu z pull requestu nebo z e?mailu (končí #review), nebo otevřete běh a stiskněte Q.



Režim kontroly · cs · desktop

2. Krok, o kterém rozhodujete, je vedle přijatého běhu, takže vidíte, co se pohnulo.

Rozhodujte z klávesnice

| Klávesa | Co udělá | |---|---| | J / K | další a předchozí krok | | A | schválí krok | | R | vrátí krok s poznámkou | | C | přidá komentář | | G | krok ve všech jazycích a velikostech obrazovky |

Když jsou rozhodnuté všechny kroky, celý běh přijměte, nebo vraťte.

Podívejte se zblízka

Ve velkém náhledu porovnáte oba běhy vedle sebe, jako rozdíl, posuvníkem (táhnete dělicí čáru přes oba screenshoty) nebo prolnutím (tento běh průsvitně přes ten druhý). Když chcete ukázat na problém, nakreslete při psaní komentáře do screenshotu rámeček, šipku nebo čáru, nebo vložte obrázek.

Převzatá schválení

U převzatého schválení je vidět, kdo ho dal a ve kterém běhu. Když si myslíte, že se má krok projít znovu, schválení zrušte a krok bude znovu čekat. Převzaté schválení za vás běh nikdy nepřijme.

Role a API tokeny

Každý člověk v organizaci má roli a každý token jedná za člověka, nikdy s většími právy, než ta role dovolí. Agenti a CI používají tokeny; místo v plánu nezabírají.

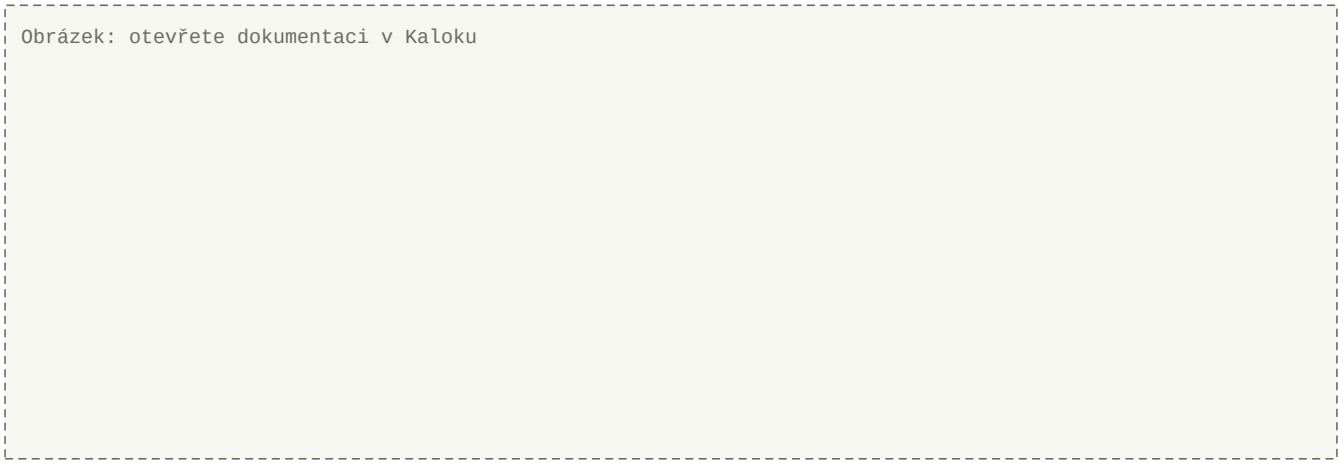
Role

| Role | Smí | Místo | |---|---|---| | Viewer | číst běhy a návrhy, komentovat, schvalovat a vracet | zdarma | | Designer | navíc pracovat s koncepty, verzemi, větvemi a sadami tokenů v Kaloko Design | jedno místo | | Vývojář / tester | navíc nahrávat a spravovat běhy a své tokeny | jedno místo | | Admin | navíc spravovat členy, domény, všechny tokeny a nastavení organizace | jedno místo |

Projekt může mít vlastního admina projektu, který do projektu zve lidi, mění jim role a projekt může omezit nebo otevřít. Projekty zakládají vývojáři / testeři a admini; kdo projekt založí, stane se jeho adminem. Roli člena změníte v Nastavení -> Lidé.

Vytvořte token v aplikaci

1. Otevřete Nastavení -> API tokeny.



API tokeny v Nastavení · cs · desktop

2. Pojmenujte token podle toho, kde bude žít, třeba "CLI na notebooku“ nebo "GitHub Actions“.

3. Zaškrtněte, co smí. Číst může vždycky.

• komentáře a schvalování: komentáře a verdikty vaším jménem

• design: kaloko sync designových konceptů

• nahrávání běhů: kaloko share z CLI nebo z CI

• vše, co dovolí moje role

4. Vyberte, jak dlouho platí: 30 dní, 90 dní nebo rok.

5. Volitelně ho omezte na vybrané projekty; pak vidí jen ty a nové nezaloží.

6. Token jednou zkopírujte a uložte do .env projektu jako KALOKO_TOKEN.

Seznam ukazuje u každého tokenu vlastníka, roli, prefix, poslední použití a platnost. Zneplatnit token ukončí okamžitě.

Přihlaste se z terminálu

Na vlastním počítači to jde rychleji:

```
npx kaloko login
```

Otevře se prohlížeč, kde kód schválíte, vyberete organizaci a to, co token smí. Stránka ukáže, odkud žádost

přišla, který terminál ji poslal a kdy: schvalte ji, jen když jste kaloko login právě spustili sami, a odkaz, který vám někdo poslal, odmítněte. Token platí nejvýš 30 dní a uloží se do ~/.config/kaloko/.env. S --print ho jen vypíše, s --no-open ukáže odkaz bez otevření prohlížeče.

Tokeny pro CI

Vytvořte token s rozsahem nahrávání běhů, nejlépe omezený na projekt, a uložte ho v CI jako secret KALOKO_TOKEN. npx kaloko ci github a ci gitlab napíšou workflow, které ho čte; víc v referenci CLI.

Tokeny pro dashboardy

Dashboard, který si jednou týdně stáhne kondici, nepotřebuje ani celé čtení: vyberte read:health. Takový token čte kondici běhů a nejnovější běh každého scénáře a na všechno ostatní, screenshoty včetně, dostane 403. Co smí volat, popisuje API kondice pro dashboardy.

Pravidla, která nastaví admin

V Nastavení -> Zabezpečení může admin povolit tokeny se všemi právy jen adminům, vypnout tokeny omezené na projekty a rozhodnout, jestli si tokeny smějí vytvářet hosté. Token hosta vidí jen jeho projekty. V plánu Enterprise drží tokeny pro automatizaci, která nepatří žádnému člověku, servisní účty.

Spouštět Kaloko v CI a komentovat pull requesty

V CI spouští Kaloko stejné příkazy jako na vašem počítači. Obvyklé nastavení projde na preview nasazení (Vercel, Netlify, Cloudflare Pages nebo jakákoli review app) to, co pull request změnil, a odkaz na plachtu napíše do komentáře v pull requestu.

1. Napište workflow

```
npx kaloko ci github # zapíše .github/workflows/kaloko.yml
npx kaloko ci gitlab # zapíše kaloko.gitlab-ci.yml; vložte ho do .gitlab-ci.yml přes include
npx kaloko ci github --print # jen ukáže, nic nezapíše
```

--scenario a.yml,b.yml ho omezí na vybrané scénáře. Příkaz zároveň přidá do kaloko.config.yml prostředí preview s base_url: ${KALOKO_PREVIEW_URL}. Scénář, který vyjmenovává environments:, musí obsahovat i preview, jinak ho kaloko start odmítne; příkaz vám řekne, kterých scénářů se to týká.

2. Přidejte token

V Nastavení -> API tokeny vytvořte token s rolí tester a přidejte ho do tajných proměnných CI v repozitáři jako KALOKO_TOKEN. Na GitLabu přidejte ještě KALOKO_GITLAB_TOKEN: project access token s rozsahem api, aby Kaloko mohlo napsat poznámku do merge requestu. Workflow commitněte.

Co workflow dělá

Po každém úspěšném preview nasazení pull requestu:

1. Nainstaluje projekt a prohlížeč (npx kaloko browser install --with-deps), a když o ně scénář žádá, i WebKit a Firefox.

2. npx kaloko ci preview-url --wait 600 --github-env najde adresu preview, pull request a jeho základní větev a předá je dál jako KALOKO_PREVIEW_URL, KALOKO_PR a KALOKO_BASE.

3. U každého scénáře kaloko affected --fail-if-none zjistí, jestli změna zasahuje některý jeho krok. Pokud ne, scénář přeskočí, jinak spustí:

```
npx kaloko start --scenario qa/flows/checkout.yml --env preview
npx kaloko walk --ephemeral --affected
npx kaloko share --pr
```

share --pr přidá do pull requestu jeden komentář za každý scénář a při dalších pushích ho aktualizuje. Soubory se posílají podle obsahu, takže nezměněné screenshoty se znovu nenahrávají.

Kontrola Kaloko v pull requestu

S GitHub App Kaloko (nainstaluje ji správce v Nastavení -> Integrace -> GitHub, od plánu Team) dostane každý pull request v připojeném repozitáři jednu kontrolu Kaloko na posledním commitu. Zezelená, když běhy sdílené pro ten commit projdou nebo když se změna žádného scénáře netýká. Čeká, dokud scénář, kterého se změna týká, nemá běh, a zčervená, když některý selže. Běhy můžou přijít z tohoto workflow, od kódovacího agenta v cloudu i z notebooku: rozhoduje commit. Běh s necommitnutými změnami se nepočítá.

Pokud má merge počkat, až bude zelená, vyžadujte v ochraně větve status check Kaloko. Kaloko samo ochranu větví nikdy nemění.

• Režimy (Nastavení, pro každý repozitář): vypnuto, ručně (jen po štítku kaloko:run nebo /kaloko walk), automaticky (výchozí) a přísně (všechny scénáře, povinné posouzení, časový limit).

• Jeden pull request: štítky kaloko:skip, kaloko:all, kaloko:review, komentář /kaloko skip <důvod> (jen aktuální commit) nebo tlačítka přímo v kontrole. Přeskočit smí jen lidé s právem, které zvolil správce, nikdy aplikace ani boti, a kontrola ukáže, kdo ji přeskočil a proč.

• V repozitáři: sekce github: v kaloko.config.yml na cílové větvi může kontrolu zpřísnit, nikdy uvolnit.

• npx kaloko ci plan řekne kontrole přesně, kterých scénářů se commit týká. Vygenerované workflow ho posílá před procházením.

• Kontrola se neobjevuje? Nastavení -> Integrace -> GitHub ukáže repozitáře a řekne proč, třeba že repozitář ještě není v instalaci aplikace.

Dobré vědět

• walk --affected potřebuje historii gitu: na GitHubu checkout s fetch-depth: 0, na GitLabu GIT_DEPTH: "0".

• --trigger pull_request spouští workflow při každém pushi místo události nasazení a na preview počká.

• kaloko ci preview-url vypíše jen adresu, takže funguje v jakémkoli CI: export KALOKO_PREVIEW_URL="$(npx kaloko ci preview-url)".

• Preview chráněné Vercel deployment protection potřebuje tajnou proměnnou VERCEL_AUTOMATION_BYPASS_SECRET a blok access.headers prostředí preview, který je ve vygenerované konfiguraci zakomentovaný.

• Běhy sdílené z CI také porovnají kondici (kontroly z packů, které jen radí) s minulým během scénáře a při nové chybě dají vědět. npx kaloko health --fail-on error zastaví job kvůli chybám v kondici, pokud to tým chce.

• V pull requestu se komentář neobjevil? Obvykle chybělo preview nasazení nebo token. Log jobu řekne, co z toho.

Pokud vaše vydání publikuje release-please, changesets nebo semantic-release, přidá npx kaloko ci github --release druhé workflow, které každé vydání zapíše do Kaloka. Všechny volby najdete v referenci CLI.

Každý týden hlídat SEO a odpovědní vyhledávače

Ve čtvrtek vyjde nová verze překladů. V pondělí si někdo všimne, že polské produktové stránky přišly o strukturovaná data a polovina řeckého magazínu nemá meta popis. Týdenní průchod všech veřejných stránek by to řekl už v pátek ráno. Tahle stránka ho nastaví na příkladu Squared: www.squared.one ve 20 jazycích a www.vyvolej.to v češtině, slovenštině a ukrajinštině, zhruba 400 stránek na jazyk.

Plán:

• jeden stack krok na každou obsahovou rodinu (e-shop, magazín, FAQ, ostatní), stránky z jazykových sitemap webu;

• lanes podle obsahových rodin, takže plachta ukáže jeden řádek na rodinu;

• sessions podle toho, čí pohled kontrolujete (návštěvník z vyhledávače, přihlášený člen klubu);

• packy seo a aeo na každé stránce, jen jako rada, nikdy součást verdiktu;

• workflow na GitHub Actions, který scénář projde každý týden a běh nasdílí;

• výsledky přes API nebo MCP pro dashboard.

1. Najděte sitemapy

Nejrychleji to jde, když kroky 1 a 2 navrhne Kaloko podle webu samotného:

```
npx kaloko init --seo https://www.squared.one
```

Přečte robots.txt, jazyky vezme z adres sitemap (nebo z prvního úseku cest, když má web jednu sitemapu), z každého jazyka si projde vzorek stránek a zapíše qa/flows/seo-squared-one.yml: jeden rotující stack krok pro každou obsahovou rodinu s match pro každý jazyk, packy seo a aeo a produkční prostředí v novém kaloko.config.yml. Rodiny napříč jazyky páruje podle odkazů hreflang v sitemapách, potom podle stejného úseku a nakonec podle velikosti. Komentář na začátku souboru vypíše páry odhadnuté podle velikosti; ty před prvním průchodem zkontrolujte. --families 6 vezme víc rodin, --limit 100 víc stránek za týden, --packs seo,aeo,a11y přidá přístupnost. Když web není dostupný, nezapíše se nic. Zbytek stránky vysvětluje, co návrh obsahuje, abyste ho mohli doladit.

Squared uvádí v robots.txt jednu zabalenou (gzip) sitemapu na jazyk:

```
Sitemap: https://www.squared.one/sitemaps/en_US/sitemap.xml.gz
Sitemap: https://www.squared.one/sitemaps/de/sitemap.xml.gz
Sitemap: https://www.squared.one/sitemaps/es/sitemap.xml.gz
…
```

items: { robots: true } tyto řádky přečte. Když adresa sitemapy obsahuje procházený jazyk jako samostatný úsek (/sitemaps/de/…), čtou se jen sitemapy toho jazyka, takže německý průchod dostane německé stránky. en_US najde en_us i en-us; en si en-us nevezme. Web s jednou sitemapou pro všechny jazyky dostane celou a jazyk vybere match. Soubor můžete pojmenovat i sami: sitemap: '/sitemaps/{locale}/sitemap.xml.gz'. Zabalené sitemapy i indexy sitemap fungují v obou případech.

Poradí si i se stagingem za heslem. kaloko walk čte robots.txt a sitemapy s přístupem prostředí (jeho hlavičky, HTTP Basic, klientské certifikáty a cookies průchodu), a to jen na vlastních adresách prostředí. Sitemapu na jiném serveru, třeba na CDN, nebo takovou, která tam přesměruje, čte bez nich. Na jeden soubor má 60 sekund a 60 MB.

2. Rozdělte stránky na obsahové rodiny

Každá rodina je stack krok s vlastním match nebo exclude. Nejdřív se podívejte na první úsek cest v sitemapě každého jazyka: na Squared je přeložený i e-shop (/fi/tuotteet/, /pl/sklep/, /bg/foto-produkti/) a magazín je /de/zeitschrift/, /es/revista/, /fi/lehti/. Jeden regulární výraz se všemi překlady udrží jeden krok na rodinu pro všechny jazyky:

```
schema: qawalk.scenario.v1
id: SEO-weekly-squared
```

```
title: Weekly SEO and answer-engine monitor (squared.one)
kind: flow
locales: [en_US, de, es, fr, it, nl, pt]
viewports: { desktop: [1280, 800] }
environments: [production]
readonly: true
page: landing
health: { site: true }
lanes: [E-shop, Magazine, FAQ, Club and other pages]
sessions:
visitor: { title: Visitor from search }
nodes:
- id: eshop
kind: stack
title: E-shop
lane: 0
session: visitor
packs: [seo, aeo, locale]
items:
robots: true
match:
'^/([a-z]{2}/)?(eshop|tuotteet|sklep|trgovina|katastima|pood|veikals|parduotuve|webshop|prod
use|foto-produkti)/'
limit: 400
fields:
- { id: title, from: title }
- { id: http, from: http }
- { id: jsonld_types, label: Structured data }
- { id: og_title, label: OG title, from: og:title }
- { id: canonical, from: canonical }
criteria:
- { id: AC1, text: Product pages carry Product and BreadcrumbList data, check: { type:
deterministic, assert: jsonld_present, types: [Product, BreadcrumbList] } }
- id: magazine
kind: stack
title: Magazine
lane: 1
session: visitor
packs: [seo, aeo, locale]
items:
robots: true
match:
'^/([a-z]{2}/)?(magazine|zeitschrift|revista|rivista|lehti|magazyn|revija|magazin|periodiko|
ajakiri|zurnals|zurnalas|spisanie)/'
fields: [{ id: title, from: title }, { id: words, from: words }, { id: jsonld_types }]
- id: faq
kind: stack
title: FAQ
lane: 2
session: visitor
packs: [seo, aeo, locale]
items: { robots: true, match: '^/([a-z]{2}/)?faq/' }
fields: [{ id: title, from: title }, { id: jsonld_types }]
- id: pages
kind: stack
title: Club and other pages
lane: 3
session: visitor
packs: [seo, aeo, locale]
```

```
items: { robots: true, exclude:
'/(eshop|tuotteet|sklep|trgovina|katastima|pood|veikals|parduotuve|webshop|produse|foto-prod
ukti|magazine|zeitschrift|revista|rivista|lehti|magazyn|revija|magazin|periodiko|ajakiri|zur
nals|zurnalas|spisanie|faq)(/|$)' }
fields: [{ id: title, from: title }, { id: jsonld_types }]
edges: []
```

Zdroj s {locale} uvnitř { … } dejte do uvozovek, jinak YAML složené závorky přečte jako mapu.

Když je jeden výraz se všemi překlady příliš dlouhý, dejte match (nebo exclude) jeden výraz pro každý jazyk. Kaloko zkusí procházený jazyk, pak jeho základ (en pro en_US) a nakonec default:

```
items:
robots: true
match:
default: '^/([a-z]{2}/)?eshop/'
fi: '^/fi/tuotteet/'
pl: '^/pl/sklep/'
bg: '^/bg/foto-produkti/'
```

Když mapa match procházený jazyk nemá a nemá ani default, průchod skončí chybou, aby zapomenutý jazyk nevzal všechny stránky webu. Mapa exclude bez jazyka nic nevyřadí.

match a exclude se vždy použijí před limit, u každého zdroje. Když stránky dodává váš skript (items: { script: steps/seo-items.mjs }), jeden skript obslouží všechny rodiny: dostane node (id, title, lane, lane_title, session) a items.params:

```
items: { script: steps/seo-items.mjs, params: { family: magazine } }
```

```
// steps/seo-items.mjs
export default async ({ locale, node, params, url }) => {
const res = await fetch(url(`/api/sitemap/${params.family}?locale=${locale}`))
return (await res.json()).map((p) => ({ path: p.path, title: p.title }))
}
```

Před prvním celým průchodem si počty ověřte přes npx kaloko walk --limit 3 a podívejte se na čísla u kroků. Krok s 0 stránkami obvykle znamená, že v jeho výrazu chybí některý přeložený úsek.

Jedna stránka ve všech jazycích

Bez dalšího nastavení čte každý jazyk vlastní seznam, takže český produkt /eshop/obaly-na-fotky/ a jeho slovenský překlad /sk/eshop/obaly-na-fotky-nahodna-farba/ jsou ve stacku dvě různé stránky a každý jazyk vybere jiné produkty. Aby se překlady ukázaly jako jazykové verze jedné stránky, nechte stack vzít seznam v jednom jazyce a k ostatním dojít přes hreflang každé stránky:

```
items:
robots: true
match: '^/eshop/'
variants: hreflang
source_locale: cs
limit: 100
rotate: week
```

Seznam, match, limit i rotate se berou ze source_locale (bez něj z prvního jazyka scénáře). V ostatních jazycích Kaloko každou vybranou stránku načte a projde adresu, kterou pro daný jazyk uvádí její <link rel="alternate" hreflang>, případně hlavička Link nebo xhtml:link v sitemapě, když stránka sama žádný nemá. Stránka si ponechá id zdrojové stránky, takže index stacku má pro každý produkt jeden řádek se snímkem v každém jazyce a všechny jazyky v daném týdnu kontrolují stejné produkty. Adresu, titulek a pole zdrojové stránky řádek ukáže, když běh prochází i zdrojový jazyk; běh jen s ostatními jazyky (--locales sk) ukáže, co našel první procházený jazyk.

Stránka, která v některém jazyce verzi nemá, tiše nezmizí. Index stacku u ní ten jazyk označí a HC1005 ji započítá: hreflang pro jazyk chybí, stránka uvádí jen x-default (to překlad není), hreflang pro jazyk vede zpátky na stránku samotnou, stránka nejde načíst, nebo překlad odpoví chybou (error 404). Započítají se i stránky, jejichž slovenský hreflang vede na stejnou adresu (obvykle výpis nebo úvodní stránku). kaloko export findings je

vypíše jako řádky daného jazyka u zdrojové stránky, kam hreflang patří. Jestli překlad odkazuje zpátky, hlídá HC125 z packu seo.

Kaloko čte HTML tak, jak ho pošle server, bez spouštění skriptů, a s přístupem prostředí (jeho hlavičky a HTTP Basic, klientské certifikáty, cookies průchodu) jen na vlastních adresách prostředí. Když žádná alternativa neleží na doméně, ze které se stránka načetla, uvádí stránka adresy jiného webu: staging nebo náhled, jehož hreflang míří na produkci, i při jedné doméně pro každý jazyk. Kaloko pak otevře cestu každé alternativy na adrese procházeného jazyka z prostředí.

Když stránky po jazycích vypisuje váš skript, dejte každé stránce key, který je ve všech jazycích stejný (třeba cestu zdrojové stránky). Stránky se stejným key jsou ve stacku jedna stránka.

Hostované běhy (kaloko schedule add) seskupí jazykové verze stejně, se stejnými stránkami ve stejném týdnu. Přečtení zdrojové stránky kvůli hreflang se tam počítá jako stránka měsíčního limitu; víc v Hostovaných běžích.

3. Vyberte sessions

Session je jeden druh návštěvníka; každý krok nějakou uvádí. Pro hlídání veřejného SEO stačí jedna: visitor, nepřihlášený, tak jak web vidí vyhledávač. Když záleží i na stránkách po přihlášení (klubová sekce pro členy), přidejte session s account, její přihlášení jednou uložte přes npx kaloko auth save --env production --account member a těm krokům dejte session: member. Kroky po přihlášení držte mimo sitemapy: vyhledávače je nikdy neuvidí.

4. Co packy kontrolují

Pack seo kontroluje titulek, popis, canonical, hreflang, Open Graph, robots, H1 a viewport a nově i to, že stránka má JSON-LD (HC112) a že každý blok je platný JSON (HC113). Pack aeo přidává levné signály pro AI hledání a asistenty: FAQPage nebo QAPage tam, kde stránka klade otázky a odpovídá (HC901), krátký odstavec s odpovědí pod H1 (HC902), /llms.txt v kořeni webu, stažený jednou za průchod (HC903), meta popis (HC904) a stránku tak, jak ji bez JavaScriptu dostane GPTBot (HC905): jeden dotaz na stránku navíc, bez cookies a jen tam, kam GPTBota pustí robots.txt. Neprojde, když se titulek, canonical, H1, JSON-LD nebo většina textu objeví až po spuštění skriptů; když web nebo WAF GPTBota odmítne, ukáže se kontrola jako nehodnocená, ne jako chyba. Jiného crawlera zkontrolujete vlastním kritériem s assert: nojs_parity a user_agent: claudebot (nebo googlebot, perplexitybot, celý řetězec user agenta). Všechny kontroly jen radí. Pack errors k těmto krokům nedávejte: patří na chybové stránky a na běžných stránkách se jeho kontroly měkké 404 a vysvětlení chyby (HC601, HC605) nepoužijí. Když tým rozhodne, že produktové stránky musí mít strukturovaná data, převezměte kontrolu nebo napište vlastní kritérium jako AC1 výše. Podrobnosti jsou v Kritéria, kontroly a packy.

Co jedna stránka neukáže, vidí stack jako celek. Po průchodu porovná pack seo stránky téhož jazyka mezi sebou: stejné titulky, popisy a H1 (HC121–HC123), canonical, který míří jinam (HC124), hreflang, na který druhá jazyková verze neodpoví (HC125), a stránky ze sitemapy, které přesměrují, odpoví chybou nebo mají noindex (HC126). Nálezy mají počet stránek a v items.json i adresy, takže se dají předat k opravě po skupinách.

Pack locale hlídá jazykové verze: html lang, vlastní hreflang stránky, jazyk textu (nepřeložená stránka v jazykové verzi) a desetinný oddělovač cen (12,50 ? v němčině, $12.50 v angličtině), vždy proti jazyku průchodu.

health: { site: true } přidá jednou za běh kontroly celého webu: robots.txt (překlepy, zablokované úvodní stránky, zablokovaní AI roboti), sitemapy z robots.txt (dostupné, platné, úplné adresy z vlastní domény, rozumné lastmod), /llms.txt a přesměrování http:// na https:// a druhé podoby domény (s www nebo bez) jedním skokem. Když web AI roboty blokuje schválně, nastavte health: { site: { ai_crawlers: false } }. V kaloko health mají kontroly celého webu vlastní řádek pro každý web.

5. Spočítejte velikost průchodu

Každá stránka trvá zhruba 1–2 sekundy a uloží asi čtyři soubory; sdílený běh pojme nejvýš 20 000 souborů. Squared má asi 400 stránek na jazyk, takže sedm jazyků (2 800 stránek, zhruba 11 000 souborů) se do jednoho běhu vejde. Jazyky rozdělte do scénářů se stejnými kroky:

Soubor	Jazyky
qa/flows/seo-squared-west.yml	en_US, de, es, fr, it, nl, pt

qa/flows/seo-squared-north-east.yml | da, sv, fi, pl, sl, hr, el | | qa/flows/seo-squared-baltic-south.yml | et, lv, lt, hu, ro, bg | | qa/flows/seo-vyvolej.yml | cs, sk, uk (prostředí s base_url: https://www.vyvolej.to) |

walk.concurrency v kaloko.config.yml prochází několik jazyků najednou; se 3 nebo 4 se týdenní běh o pár tisících stránkách vejde do hodiny až dvou.

Velký katalog po částech

Když mají všechny jazyky dohromady víc stránek, než se vejde do jednoho běhu, projděte každý týden jen část každé rodiny. rotate: week vezme limit stránek a každý ISO týden začne jinde, takže celý seznam projde za počet stránek ÷ limit týdnů:

```
items: { robots: true, match: '^/([a-z]{2}/)?eshop/', limit: 100, rotate: week }
```

Se 400 produktovými stránkami na jazyk projde e-shop celý jednou za čtyři týdny za čtvrtinovou cenu. Průchod vypíše, kterou část bere (rotating sample, week 2026-W41 — pages 101…200 of 400, all covered in 4 weeks). Stejný týden dá stejné stránky, takže běh zopakovaný v úterý zkontroluje totéž co pondělní.

rotate: run počítá běhy místo týdnů. Počet a čas poslední kontroly každé stránky Kaloko drží v .cache/stack/ výstupní složky (tmp/kaloko), takže funguje tam, kde složka mezi běhy zůstává (váš počítač, vlastní runner). Čistý checkout na GitHub Actions začíná pokaždé od nuly: tam použijte rotate: week.

steps/<krok>/items.json v běhu pak má pro každý jazyk sekci rotation (okno, kolik stránek už prošlo kontrolou a kolik ještě nikdy) a každá stránka svůj last_checked: kdy ji Kaloko zkontroloval naposledy předtím, nebo null.

Ať se velký běh vejde

Sdílený běh pojme 300 MB v uložené podobě. HTML a JSON se posílají zabalené gzipem, takže snímky stránek zaberou zhruba desetinu své velikosti na disku; nejvíc přidávají screenshoty. Stránky stacku se fotí celé, takže typická dlouhá stránka uloží screenshot kolem 600 kB a náhled kolem 180 kB (náhled je stejně vysoký jako stránka). Běh se tak zaplní zhruba při 400 stránkách.

Rodina stránek pro SEO potřebuje stránku, ne její obrázek. capture: { screenshot: thumb, full_page: false } ponechá jen náhled první obrazovky široký 640 px, asi 25 kB, podle kterého stránku ve stacku poznáte:

```
items: { robots: true, match: '^/([a-z]{2}/)?eshop/', limit: 100, rotate: week }
capture: { screenshot: thumb, full_page: false }
```

Samotné capture: { full_page: false } uloží ostrou první obrazovku (asi 70 kB), samotné capture: { screenshot: thumb } náhled celé stránky. Výchozí nastavení nechte u rodin, kde někdo prochází celou stránku nebo kde sémantická kontrola čte, co je pod první obrazovkou.

Když se běh blíží limitu, kaloko share vypíše největší kroky, ať víte, kterou rodinu zkrátit, a upozorní, když většinu místa zabírá stack focený celý.

6. Procházejte ho každý týden na GitHub Actions

```
npx kaloko schedule export github --scenario qa/flows/seo-squared-west.yml --env production
--every week --at 5
```

Příkaz zapíše workflow, který scénář projde v pondělí v 5:00 UTC, vyhodnotí ho a běh nasdílí. Potřebuje secret KALOKO_TOKEN (role Tester, rozsah upload). Pro každý scénář vyexportujte jeden workflow, každý v jinou hodinu, a u velkých zvyšte ve workflow timeout-minutes. Když konfigurace říká, že každý scénář uvádí vlastní projekt (komentář, nebo produkt s projects:), dejte scénáři project: seo nebo jiný z nich; kaloko validate na chybějící projekt upozorní a sdílení bez něj může skončit chybou HTTP 404.

Bez GitHub Actions projde stejný scénář na straně Kaloka kaloko schedule add (plán Business). Stacky čte ze sitemap a z robots.txt, bere base_url pro každý jazyk a bez screenshotů rozhodne kontroly, kterým stačí prvky stránky; každá stránka stacku se počítá do měsíčního rozpočtu hostovaných stránek. Strukturovaná data, kontroly pro odpovědní vyhledávače a jazyka tam zůstanou "nevyhodnoceno“, takže pro plné packy SEO a AEO patří dál plné CLI na GitHub Actions. Víc v návodu Kontrolovat stránky podle plánu.

7. Výsledky do dashboardu

Každý sdílený běh má svou kondici: nálezy podle závažnosti, packu a kroku a co přibylo nebo se spravilo od

minulého týdne. Pro dashboard:

• GET /api/v1/orgs/<org>/scenarios/<id>/latest/health/aggregate?by=locale,step,check sečte nejnovější běh po jazycích, obsahových rodinách a kontrolách, s nejvíc zasaženými stránkami a s tím, kolik výsledků je od minulého týdne nových nebo opravených. …/latest/health/diff je vypíše stránku po stránce, každou s týdnem, kdy poprvé selhala. Token s rozsahem read:health čte tohle a nic dalšího, což se hodí pro dashboard, který si data stáhne jednou týdně. Tvar odpovědí je v API kondice pro dashboardy; MCP má get_health_aggregate a get_health_diff.

• Webhook health.changed dá po každém běhu vědět, když je něco nového nebo opraveného.

• GET /api/v1/orgs/<org>/health?project=<projekt> nebo MCP get_health { project } vrátí vývoj projektu; get_health { scenario } nejnovější běh jednoho scénáře.

• MCP get_stack_items { run_id, step, status?, criterion?, query? } vypíše stránky jednoho kroku s jejich poli (jsonld_types, og_title), stavem a nesplněnými kontrolami, třeba každou stránku e-shopu, které chybí HC112.

• Ve složce běhu má steps/<krok>/items.json totéž po stránkách a jeho structured_data sčítá stránky podle typu schema.org, stránky bez strukturovaných dat a stránky s neplatným JSON-LD. kaloko health --json vypíše nálezy lokálního běhu.

Obvykle stačí týdenní graf "stránky bez strukturovaných dat“ a "nové chyby kondice“ po obsahových rodinách. Když číslo poskočí, otevřete běh na plachtě a stack vyfiltrujte podle neúspěšné kontroly.

8. Předejte opravy tomu, kdo je udělá

Najít chybu je půlka práce; redaktor pak ještě musí najít stránku v CMS. Dejte každé stránce cestu zpátky:

• skript se stránkami vrátí vedle path i edit_url (stránka v CMS) a content_ref (záznam, pod kterým ji zná obsahové API);

• nebo stack dostane resolver, který je dohledá pro stránky procházené tento týden:

```
items: { robots: true, match: '^/([a-z]{2}/)?eshop/', limit: 100, rotate: week, resolve:
steps/resolve.mjs }
```

```
// steps/resolve.mjs
export default async (item, { locale }) => {
const res = await
fetch(`https://cms.example.com/api/pages?path=${encodeURIComponent(item.path)}&locale=${loca
le}`)
const page = res.ok ? await res.json() : null
return page ? { edit_url: `https://cms.example.com/pages/${page.id}/edit`, content_ref:
`page:${page.id}` } : {}
}
```

Stránka, u které resolver selže nebo do 15 sekund neodpoví, se projde i tak, jen bez odkazu, a průchod řekne, kolik jich bylo. Detail stránky na plachtě pak ukáže Upravit v CMS a CSV stacku má oba sloupce. Celý seznam:

```
npx kaloko export findings --format csv --out opravy.csv
npx kaloko export findings --format json --locale de --step eshop
```

Každý řádek je jedna nesplněná kontrola na jedné stránce v jednom jazyce: URL, kontrola, důkaz, edit_url, content_ref a příkaz, který stránku projde znovu (kaloko walk --steps eshop --items <stránka> --locales de). JSON jde rovnou agentovi s obsahovým API, u Squared nástrojům MCP, které změnu navrhnou, provedou a ověří. Po opravě stačí ten jeden příkaz a stránka se zkontroluje znovu bez procházení zbytku, i když rotující stack mezitím přešel na jiné stránky; takový průchod se do rotace nepočítá.

Pracujte s nálezy, ne se stránkami

Dvě stě produktových stránek bez strukturovaných dat v polštině je jeden problém v šabloně produktu, ne dvě stě. Kondice -> Nálezy u scénáře (odkaz u každého scénáře na stránce kondice) seskupí, co je otevřené, podle kontroly, webu, jazyka a kroku: jeden řádek na problém, se závažností, počtem zasažených stránek, několika z nich s odkazem upravit a týdnem, kdy se objevil poprvé. Nález potvrďte, když na něm někdo pracuje, nebo ho ignorujte a napište proč ("starý blog, v listopadu se stěhuje do nového CMS“). Rozhodnutí u nálezu zůstává

týden po týdnu a nález je opravený, jakmile žádná jeho stránka neselhává. Dashboard nebo agent čte totéž přes GET …/scenarios/<id>/findings nebo MCP list_findings.

Když tým pracuje v Jiře, Linearu, GitHub Issues nebo Plane, může Developer / Tester nechat scénář zakládat jedno issue na každý nový nález od zvolené závažnosti (PUT …/scenarios/<id>/findings/sync { "tracker": "jira", "min_severity": "error" }). Kaloko pak issue okomentuje, když se počet stránek hodně změní, a zavře ho, když je nález opravený. Práci rozdělí cesty (routes): technické kontroly (canonical, hreflang, strukturovaná data) do repozitáře vývojářů na GitHubu, obsahové (popisky, tenký obsah, nízká míra prokliku) do projektu marketingu v Plane. S daty o návštěvnosti (oddíl 9) issue řekne, kolik kliknutí a zobrazení stránky mají. Podrobnosti jsou v API kondice pro dashboardy.

9. Opravte nejdřív stránky, které lidé opravdu vidí

Dvě stě chybějících popisů je dlouhý seznam. Search Console ví, které z těch stránek lidé ve vyhledávání vidí, a Kaloko to umí využít. Vyexportujte ze Search Console tabulku Stránky (Výkon, Stránky, Exportovat, CSV), uložte ji vedle scénáře a uveďte ji:

```
traffic:
file: traffic/search-console-stranky.csv
```

Export stárne, takže až týdenní běh poběží, nechte ho ptát se Search Console přímo. V Google Cloud založte servisní účet, jeho e-mail přidejte do vlastnosti jako uživatele, cestu k jeho JSON klíči dejte do GOOGLE_APPLICATION_CREDENTIALS (v GitHub Actions jako secret repozitáře) a soubor nahraďte vlastností:

```
traffic:
gsc: { property: sc-domain:example.com }
```

Každý běh si pak jednou řekne o kliknutí, zobrazení, CTR a pozici všech stránek za posledních 28 dní. ga4: { property: "123456789" } stejně přidá relace a klíčové události podle vstupních stránek.

Co se změní:

• kaloko export findings dá nahoru nálezy na nejvíc zobrazovaných stránkách a ke každému řádku přidá clicks, impressions, ctr, position, sessions a impact, takže obsahový tým začne odshora;

• index stránek na plachtě má sloupce Zobrazení a Kliknutí; kliknutím podle nich seřadíte;

• pack seo přidá ke každému stacku HC127: stránky, které se ve vyhledávání ukazují často, ale klikne se na ně ani ne napůl tak často jako na stránky webu na podobných pozicích. Obvykle za to můžou titulek a popis ve výsledcích.

Vzorec a všechny klíče najdete v Konfiguračním souboru. Data z jiných SEO nástrojů (klíčová slova, viditelnost v AI) Kaloko zatím nepřipojuje.

Projít aplikace pro Android, iOS a desktop

Nativní aplikace Kaloko prochází stejně jako webové stránky: z každého kroku je screenshot s UI stromem, zkontrolovaná kritéria a místo na plachtě. Scénář říká, o jakou platformu jde, a konfigurace, kterou aplikaci otevřít.

1. Pojmenujte aplikaci pro každé prostředí

```
# kaloko.config.yml
environments:
staging:
app:
android: { package: com.acme.shop.staging, apk: build/app-staging.apk, reset: true }
ios: { bundle: com.acme.shop, app: build/Shop.app, device: iPhone 17 }
desktop:
app:
electron: { main: dist/main.js }
# desktop: { name: Notes } # aplikace pro macOS podle okna
# windows: { name: contoso, path: dist/Contoso.exe }
```

2. Nastavte platformu ve scénáři

```
platform: android # android | ios | electron | desktop | windows
devices: [pixel-8-api-34, pixel-7-api-33]
color_schemes: [light, dark]
nodes:
- id: product
kind: screen
path: shop://product/42 # deep link
packs: [app]
```

Na kroky se aplikace dostane přes deep link nebo skriptem kroku (app.tap(…), app.type(…)). V každém jazyce se aplikace spustí znovu a v daném jazyce, reset: true nejdřív smaže její data.

Co která platforma umí

| Platforma | Co ji ovládá | Poznámka | |---|---|---| | Android | adb: emulátor, telefon přes USB nebo přes Wi?Fi | neběžící emulátor se spustí; tmavý režim a omezený pohyb se nastaví pro každý snímek | | iOS Simulator | xcrun simctl | ťuknutí a UI strom potřebují idb; bez něj Kaloko přečte text z obrazovky | | iPhone a iPad | devicectl a nástroj na screenshoty | jen zachycení, text se čte z obrazovky | | Electron | Playwright | okno je webová stránka: platí všechny webové kontroly | | Aplikace pro macOS | okno aplikace a strom přístupnosti | jen zachycení; mezi obrazovkami přechází váš nástroj, AppleScript nebo člověk | | Aplikace pro Windows | UI Automation přes PowerShell | ťuknutí i psaní; procházejte ji ve Windows |

Pack app kontroluje popisky ovládacích prvků, velikost dotykových ploch, useknutý text a pády.

Obrazovky z jiných nástrojů

Screenshot může pořídit Maestro, Appium, XCUITest, Detox nebo člověk a Kaloko ho rozloží i s UI stromem:

```
npx kaloko capture --step checkout --image shot.png --tree hierarchy.xml --device "Pixel 8" --density 2.625
```

--tree přečte XML z uiautomatoru nebo Appia, idb ui describe-all --json a JSON hierarchie z Maestra. Bez stromu přečte --ocr text z obrázku (na macOS). Celou složku screenshotů nebo report testů nahrajete přes npx kaloko import.

Nejdřív zkontrolujte počítač

```
npx kaloko doctor
```

Vypíše, které nástroje pro aplikace na počítači chybějí (adb, simctl, idb, ffmpeg), a jak je nainstalovat. Na macOS si kaloko doctor --fix vyžádá oprávnění k nahrávání obrazovky a zpřístupnění, která desktopové snímky potřebují.

Napsat scénář

Scénář je flow, které Kaloko projde: soubor YAML s obrazovkami, cestou mezi nimi a tím, co má každá obrazovka splňovat. Scénáře k úkolům bývají u úkolu (docs/tasks/TASK-123/qa/scenario.yml), dlouhodobá flow, třeba pokladna, v qa/flows/.

Nechte ho napsat agenta

S nainstalovaným skillem nemusíte psát YAML, stačí zadání:

```
Přečti si úkol v docs/tasks/TASK-123, napiš k němu akceptační plán a scénář pro Kaloko a zvaliduj ho.
```

Agent napíše dva soubory: ACCEPTANCE.md pro lidi (cíl, kroky, kritéria AC1, AC2…) a scénář pro běh. Když se zadání změní, upraví nejdřív plán a potom scénář.

Z čeho se soubor skládá

```
schema: qawalk.scenario.v1
id: TASK-123-checkout
title: Pokladna
kind: task # task | flow
product: shop # volitelné, s module:
locales: [cs, en]
viewports:
desktop: [1280, 800]
mobile: [360, 800]
environments: [local, staging]
nodes:
- id: cart
kind: screen
title: Košík
path: /cart
purpose: Návštěvník vidí, co kupuje, a může pokračovat k platbě.
execution:
strategy: playwright # playwright | agent | manual
script: steps/cart.mjs
criteria:
- id: AC1
text: Stránka má právě jeden hlavní nadpis.
check: { type: deterministic, assert: count, selector: h1, min: 1, max: 1 }
edges:
- { from: cart, to: payment, label: pokračovat }
```

• Id kroků (cart) neměňte: odkazují na ně odkazy, komentáře, schválení i obrázky v dokumentaci.

• purpose je jedna věta o tom, k čemu obrazovka slouží. Čte ji evaluátor.

• execution říká, kdo se na obrazovku dostane: skript v Playwrightu, agent, který ji zachytí přes kaloko capture, nebo člověk.

• Další druhy uzlů: decision pro větvení, email pro zprávu ze schránky, stack pro mnoho stránek jedné šablony.

Všechny klíče najdete v referenci souboru scénáře. Jak přidat kontroly, popisuje návod Přidat kritéria a packy kontrol.

Zkontrolujte ho

```
npx kaloko validate qa/flows/checkout.yml
npx kaloko validate qa/flows/checkout.yml --coverage
```

validate ověří schéma a odkazy mezi kroky. --coverage porovná kritéria s akceptačním plánem a vypíše, co plán

chce a scénář nekontroluje.

Začněte od veřejných stránek

U marketingového webu nebo dokumentace začněte konceptem:

```
npx kaloko draft https://example.com/ https://example.com/pricing
```

Kaloko stránky stáhne a napíše k nim produkční scénář jen pro čtení i s plánem. Jako účel použije titulek, H1 a popis stránky a přidá packy seo, a11y, perf a console. Přidá i krok s adresou, která neexistuje, takže se zkontroluje i stránka 404. Pak účel každé stránky upřesněte a doplňte, co musí opravdu splňovat.

Kritéria, kontroly a packy

Kritérium je jedna věc, kterou má krok splnit. Má id (AC1), text pro lidi a právě jednu kontrolu check. Jak je do scénáře přidat: Přidat kritéria a packy. Jak se z výsledků stane verdikt: Jak vznikají verdikty.

Druhy kontrol

| type | Kdy ho použít | Volby | |---|---|---| | deterministic | cokoli spočitatelného nebo přesného: přítomnost, počty, text, URL, styly, hodnoty ve stránce | assert a jeho volby (níže) | | semantic | úsudek: je text srozumitelný, je na stránce jedna hlavní akce | question (anglicky, ano = splněno), true a false (co se počítá za ano a za ne), scope (CSS selektor, který se pošle evaluátorovi), about: console (čte chyby v konzoli místo stránky), vision: true (ukáže i screenshot) | | manual | nehodí se ani jedno; rozhodne kontrolující na plachtě | note |

Čísla, data a počty patří do deterministických kontrol, nikdy do sémantické otázky. V jedné otázce se ptejte na jednu věc.

Deterministické kontroly (assert)

Společné volby: selector (CSS), pattern (regulární výraz) s flags (i, m, s), value, min, max, all (vyhovět musí každá shoda, ne jen jedna).

Obsah stránky

| assert | Splněno, když | |---|---| | exists | selector najde aspoň jeden prvek | | absent | selector nenajde nic | | count | počet shod je mezi min a max | | text_matches | viditelný text prvku odpovídá pattern | | text_absent | žádný viditelný text neodpovídá pattern | | text_equals | celý text se rovná value (mezery sloučené) | | attr_equals | atribut attr se rovná value | | attr_matches | atribut attr odpovídá pattern | | no_overflow | stránka se neposouvá do strany | | no_console_errors | v konzoli nejsou chyby |

Adresa, metadata a odkazy

| assert | Splněno, když | |---|---| | title_matches | titulek stránky odpovídá pattern | | meta_matches | content meta tagu ze selector odpovídá pattern | | link_matches | href odkazu ze selector odpovídá pattern | | url_matches | URL po kroku odpovídá pattern; part vybere href, path, query, hash, host nebo origin | | url_equals | URL se rovná value (cesta nebo celá URL); ignore_query vynechá ?… a #… | | http_status | stav dokumentu je mezi min a max (výchozí 200–299), nebo se rovná value | | link_home | stránka odkazuje zpět na úvodní stránku | | links_ok | každý odkaz na stejný původ odpoví pod 400 (max odkazů na stránku, pattern přeskočí cesty, follow mimo prostředí jen pro čtení) | | hreflang_pairs | aspoň min jazyků, každá verze odpoví 200 a odkazuje zpět |

Strukturovaná data a odpovědní vyhledávače

| assert | Splněno, když | |---|---| | jsonld_present | stránka má <script type="application/ld+json">; s types: [Product, BreadcrumbList] jsou ve strukturovaných datech všechny uvedené typy (počítají se grafy, seznamy, vnořené hodnoty jako offers i mikrodata) | | jsonld_valid | každý blok JSON-LD jde přečíst jako JSON; stránka bez JSON-LD projde. Neúspěch uvede blok a chybu | | aeo_faq_schema | stránka s otázkami a odpověďmi (aspoň dvě details > summary, dl > dt nebo nadpisy h2–h4 s otazníkem na konci) má strukturovaná data FAQPage nebo QAPage. Stránka bez otázek se nehodnotí | | aeo_answer_first | první odstavec pod H1 má min až max znaků (výchozí 40–320): krátká odpověď před podrobnostmi | | llms_txt | /llms.txt webu odpoví 200 s textem, ne HTML stránkou. Jeden dotaz na web a průchod | | nojs_parity | holé HTML stránky stažené bez JavaScriptu obsahuje to, co ukazuje vykreslená stránka: titulek a canonical se shodují, meta popis, H1 a všechny typy JSON-LD v něm jsou a text těla stránky má aspoň ratio vykresleného (výchozí 0,3; posuzuje se, když má vykreslená stránka aspoň 200 znaků). user_agent: stahuje jako crawler: gptbot, oai-searchbot, claudebot, perplexitybot, googlebot, bingbot nebo celý řetězec user agenta |

Bez JavaScriptu (nojs_parity). Spousta AI crawlerů čte HTML, které pošle server, a skripty nikdy nespustí. Kontrola udělá na každou stránku jeden GET navíc, přečte ho v Node bez prohlížeče (nejvýš 500 000 znaků, 15 s) a porovná ho se stránkou, kterou průchod zachytil. Titulky a popisy porovnává tak, jak je čtou lidé (–, », pevné mezery a podobně rozluštěné). Text počítá z celého těla stránky na obou stranách, skryté části také, takže část, kterou Next.js streamuje do skrytého prvku, se počítá stejně jako po vykreslení. S user_agent:

přijde jako ten crawler: bez cookies, s user agentem bota a s Accept-Language průchodu, a jen tehdy, když to robots.txt dovolí. Stránku, kterou robots.txt botovi zavírá, nestáhne a ukáže ji jako nehodnocenou s poznámkou "blocked by robots.txt for GPTBot“; stejně tak všechny stránky, když robots.txt odpoví 5xx, protože crawlery pak nestahují nic. Když web dotaz odmítne (401, 403, 429 nebo 503, stránka s ověřením, přesměrování na jinou doménu, třeba na přihlášení), kontrola se také nehodnotí a řekne, koho web odmítl: user agenta bota (prohlížeč stránku dostane), dotaz bez cookies přihlášení nebo brány, nebo kohokoli. Nejde o problém s vykreslením. Bez user_agent: použije user agenta a přihlášení prohlížeče z průchodu a robots.txt nečte. Důkaz vypíše, co bez JavaScriptu chybí, třeba GPTBot without JavaScript: no H1 (rendered: "Photo books"); JSON-LD Product only after JavaScript; body text 0 of 1840 characters (0 %), at least 30 % expected. Titulek v jiném jazyce to řekne i s oběma hodnotami lang. Delší stránku kontrola ořízne a její JSON-LD pak neposuzuje. Každou stránku a user agenta stáhne jednou za průchod (průchod si drží jen krátké shrnutí, ne stránku), robots.txt jednou za web a bota. Porovnávejte vstupní kroky: holé HTML je stránka tak, jak se načte, ne po kliknutích kroku.

Každý snímek navíc zapíše structured_data: počet bloků JSON-LD, neplatné bloky, typy schema.org a z nich ty rozpoznané (Product, Offer, AggregateOffer, BreadcrumbList, FAQPage, QAPage, Article, BlogPosting, NewsArticle, Organization, WebSite). items.json stohového kroku sečte stránky podle typu, stránky bez strukturovaných dat a stránky s neplatným JSON-LD (structured_data na začátku indexu a v stack kroku).

Jazykové verze (používá je pack locale)

Porovnávají stránku s jazykem, ve kterém se procházela (cs, de, en_US). Když jazyk průchodu není jazykový kód (jazyk podle ISO 639-1, třeba cs; cz je země), nepoužijí se.

| assert | Splněno, když | |---|---| | locale_lang | html lang uvádí jazyk průchodu (de-DE pro de); jiný region jen zmíní, neselže kvůli němu | | locale_hreflang | vlastní hreflang stránky ji označí jejím jazykem, každý kód je platný (en-GB, ne en_UK) a tam, kde adresa nese jazyk (/de/…), kód s ní souhlasí. Stránka bez hreflang se nehodnotí | | locale_text | text stránky je v jazyce průchodu. Kaloko počítá běžná slova 26 jazyků (většina Evropy a turečtina); stránky pod min slov (výchozí 40), jiné jazyky a blízké dvojice (dánština, norština, švédština; chorvatština, slovinština) nehodnotí | | locale_prices | ceny s desetinnými místy používají oddělovač daného jazyka (12,50 ? v de, $12.50 v en). Stránky bez takových cen se nehodnotí |

Styly a hodnoty ve stránce

| assert | Splněno, když | |---|---| | style_equals | spočítaná CSS vlastnost property se rovná value nebo design tokenu token; op porovnává čísla (>=), tolerance dovolí zaokrouhlení v px | | style_matches | spočítaná property odpovídá pattern | | js_equals | jeden JavaScriptový výraz expression, který jen čte, vrátí ve stránce value (nebo odpovídá pattern, nebo se porovná přes op) |

Přístupnost (používá je pack a11y)

| assert | Splněno, když | |---|---| | a11y_images_alt | obrázky mají alternativní text | | a11y_names | odkazy a tlačítka mají přístupný název | | a11y_labels | formulářová pole mají popisky | | a11y_headings | úrovně nadpisů se nepřeskakují | | a11y_contrast | text splňuje kontrast WCAG AA | | a11y_main | stránka má landmark main | | a11y_unique_ids | id prvků jsou jedinečná | | a11y_focus_order | pořadí tabulátoru odpovídá stránce a dojde ke každému ovládacímu prvku | | a11y_focus_visible | fokus je vidět s kontrastem aspoň 3 : 1 | | a11y_keyboard_trap | tabulátor nikde neuvízne | | a11y_skip_link | odkaz pro přeskočení nebo hlavní obsah do tří zastavení tabulátoru | | a11y_ax_names | ovládací prvky mají název ve stromu přístupnosti prohlížeče | | a11y_landmarks | landmarky jsou úplné a opakované mají popisek | | a11y_aria | role a atributy ARIA jsou platné |

Výkon, aplikace a design

| assert | Splněno, když | |---|---| | perf_lcp, perf_cls, perf_ttfb, perf_weight | vykreslení největšího obsahu, posun rozvržení, čas do prvního bajtu nebo váha stránky nepřekročí max | | app_labels | každý ovládací prvek aplikace má přístupný popisek | | app_touch_targets | dotykové plochy mají aspoň 48 dp (Android) nebo 44 pt (iOS) | | app_no_truncation | žádný text není useknutý trojtečkou | | app_no_crash | aplikace nespadla ani nepřestala odpovídat | | design_pixels | obrazovka se od reference designu liší nejvýš v podílu ratio pixelů (výchozí 0,03) | | design_structure | nadpisy, landmarky, akce a pole odpovídají referenci designu | | uses_tokens | barvy, písma, velikosti, mezery a zaoblení pocházejí ze sady tokenů |

Packy

packs: [seo, a11y] u kroku přidá pevnou sadu kontrol. Jsou to kontroly kondice: jen radí, ukazují se zvlášť od přijetí a do verdiktu kroku se nepočítají. Pack design se do přijetí počítá vždy, ostatní jen tehdy, když je scénář uvede v health: { acceptance: [seo] }. Všechny kontroly všech packů, proč na nich záleží a jak jsou závažné, najdete v Kontrolách kondice.

Na designovém prostředí (kind: design) jsou kroky vaše vlastní HTML ze složky, a tak jsou tam packy, které splní jen kód (seo, aeo, perf, console, errors), vypnuté. Pokud některý chcete i tam, uveďte ho ve scénáři v health: { design_packs: [console] }. a11y, usability a locale běží i na designech.

Kontroly kondice mají id HC, akceptační kritéria id AC. Číslo patří kontrole, takže HC401 (LCP jako kontrola kondice) a AC401 jsou táž kontrola ve dvou rolích: pack v přijetí přidá AC401 a převzetím HC401 dostane krok kritérium AC401 s from: HC401. Vlastní kritérium, které zaujme místo kontroly, ji nahradí a počítá se do přijetí; tak se mění limit (AC401 s max: 4000) nebo převezme jedna kontrola (kaloko health adopt <krok>:<kontrola>). Běhy a scénáře z doby před 7. 10. 2026 označují kontroly kondice AC101…; Kaloko je čte jako HC101… a stará id přijímá i v příkazech.

| Pack | Id | Co přidá | |---|---|---| | seo | HC101–HC113, HC121–HC127 | jeden H1, titulek 10–70 a popis 50–160 znaků, canonical, hreflang pro každý jazyk, Open Graph, žádný noindex, meta viewport, html lang, žádné přetečení, žádné chyby v konzoli, JSON-LD na stránce (HC112) a platný (HC113); u stack kroku navíc kontroly napříč jeho stránkami (níže) | | usability | HC201–HC204 | účel je jasný z první obrazovky, jedna hlavní akce, pole s popisky, texty odkazů, které říkají, co udělají | | a11y | HC301–HC314 | všechny kontroly přístupnosti výše; průchod navíc projde stránku tabulátorem, nic nekliká ani nepíše | | perf | HC401–HC404 | LCP do 2,5 s, CLS pod 0,1, TTFB do 800 ms, váha stránky do 3 MB (měřeno v prohlížeči průchodu) | | console | HC501 | evaluátor si přečte chyby v konzoli a řekne, jestli se týkají toho, co krok přijímá | | errors | HC601–HC606 | pro chybové stránky: skutečný stav 4xx/5xx, odkaz na úvod, navigace nebo hledání, žádný výpis chyby ze serveru, srozumitelná zpráva s cestou dál, žádné chyby v konzoli. HC601 a HC605 platí jen pro chybové stránky (krok s expect_status: 404 nebo stránka s odpovědí 4xx/5xx); jinde se nepoužijí | | app | HC701–HC705 | popisky, dotykové plochy, žádný useknutý text, žádný pád, jasný účel a další krok | | design | AC801–AC803 | implementace proti referenci designu: pixely, struktura, tokeny | | aeo | HC901–HC905 | pro odpovědní vyhledávače (AI hledání, asistenti): FAQPage nebo QAPage tam, kde stránka klade otázky a odpovídá, krátký odstavec s odpovědí (40–320 znaků) pod H1, /llms.txt v kořeni webu, meta popis a stránka bez JavaScriptu tak, jak ji dostane GPTBot (HC905, nojs_parity) | | locale | HC1001–HC1004 (HC1005 u stacků s items.variants, níže) | pro vícejazyčné weby: html lang, vlastní hreflang stránky, jazyk textu a desetinný oddělovač cen odpovídají jazyku průchodu. Jen ve scénářích, které uvádějí locales |

Pack a11y pokrývá část WCAG 2.2 AA. Audit se čtečkou obrazovky nenahradí.

Pack aeo čte levné signály ze stránky, neměří pořadí v žádném AI asistentovi. Hostované běhy zkontrolují jeho meta popis (HC904), ostatní kontroly aeo a také HC112 a HC113 ukážou jako "nehodnoceno“; kaloko walk a plánovaný workflow na GitHubu vyhodnotí všechny. HC905 potřebuje živou stránku: kaloko recheck ji nechá tak, jak ji našel průchod. Id HC9xx patří packu aeo, HC10xx packu locale; i kontroly locale hostované běhy ukážou jako "nehodnoceno“.

Napříč stránkami stacku (pack seo)

Stack krok s packem seo po průchodu porovná své stránky i mezi sebou. Nic se znovu nestahuje: kontroly čtou titulek, popis, H1, canonical, meta robots, hreflang a stav HTTP z každého snímku.

| Id | Selže, když | |---|---| | HC121 | dvě nebo více stránek jednoho jazyka mají stejný titulek. Stránky s chybovou odpovědí, stránky s kanonickou adresou jinam a stejná adresa podruhé se nepočítají | | HC122 | … stejný meta popis | | HC123 | … stejný H1 | | HC124 | stránka uvádí jako kanonickou jinou adresu (stránky mířící na jednu adresu tvoří shluk), nebo canonical míří na stránku ze stacku, která odpoví chybou nebo má noindex. Na stagingu nebo náhledu, kde canonical míří na produkční doménu, platí za stránku samotnou, když se liší jen doménou | | HC125 | stránka s hreflang neoznačí sama sebe, nebo jazyková verze, kterou stack prošel, odpoví chybou či neodkáže zpátky. Alternativy mimo stack se jen spočítají, nestahují se (stahuje je hreflang_pairs); chybějící x-default se zmíní, neselže kvůli němu | | HC126 | stack načtený ze sitemapy (items.sitemap,

items.robots): stránka odpoví jinak než 200, přesměruje nebo má noindex | | HC127 | jen s daty o návštěvnosti (traffic v konfiguraci nebo scénáři): stránka s aspoň 100 zobrazeními má CTR pod polovinou mediánu webu na podobných pozicích (1–3, 4–10, 11–20). Přepište její titulek a popis (Konfigurační soubor) | | HC1005 | stack s items.variants: hreflang, s packem i bez něj: stránka zdrojového jazyka nemá v procházeném jazyce vlastní verzi (hreflang pro jazyk chybí, je tam jen x-default, hreflang vede zpátky na stránku samotnou, stránka nejde načíst, nebo překlad odpoví 4xx/5xx: error 404), nebo víc stránek vede na stejnou verzi. Hreflang čte průchod, když hledá jazykové verze |

Jsou to nálezy kondice stack kroku, počítané ve stránkách. steps/<krok>/items.json je drží v cross_page se skupinami a jejich adresami (nejvýš 20 skupin po 10) a každá dotčená stránka má id v cross_fails. Kontrolu napříč stránkami nejde převzít jako kritérium jedné stránky. Osiřelé stránky a hloubku prokliku Kaloko zatím nekontroluje.

Kontroly celého webu (health.site)

Některé věci patří celému webu, ne stránce. S health: { site: true } ve scénáři zkontroluje průchod každý web jednou za běh (jednou pro každý origin ze základních adres prostředí) a zapíše do běhu site.json:

| Id | Selže, když | |---|---| | HC1101 | /robots.txt neodpoví 200, je to HTML stránka, nebo má řádky, které roboti přeskočí (překlep jako Disalow, pravidlo před prvním User-agent) | | HC1102 | robots.txt blokuje / nebo úvodní cestu některého procházeného jazyka všem robotům, Googlebotu nebo Bingbotu. Na prostředí production je to chyba, jinde varování | | HC1103 | robots.txt blokuje GPTBot, OAI-SearchBot, ChatGPT-User, PerplexityBot, ClaudeBot, Google-Extended nebo CCBot na celém webu nebo na procházené úvodní cestě. Kontroluje se jen na production | | HC1104 | sitemapa uvedená v robots.txt (nebo /sitemap.xml, když robots.txt žádnou neuvádí) neodpoví, není urlset ani index sitemap, je useknutá, má relativní adresy nebo stránky z jiné domény než sitemapa a web (robots.txt smí ukázat na sitemapu jinde), víc než 50 000 adres nebo hodnoty lastmod, které nejsou datum nebo leží v budoucnosti. Indexy sitemap se procházejí do dvou úrovní, nejvýš 50 souborů | | HC1105 | /llms.txt neodpoví 200 s textem | | HC1106 | http:// nepřesměruje rovnou na https:// téže domény (dočasné 302 nebo 307 zmíní). Na jiném než výchozím portu se nehodnotí | | HC1107 | druhá podoba domény (example.com pro www.example.com a naopak) odpovídá sama, nebo její první přesměrování vede jinam. Subdomény druhou podobu nemají a nehodnotí se; jiný než výchozí port zůstává |

Kontroly posílají vlastní požadavky, proto jsou vypnuté, dokud o ně scénář nepožádá. Ochranu prostředí (hlavičky, HTTP Basic) posílají jen na origin webu, přesměrování procházejí samy, z odpovědi přečtou nejvýš 10 MB a na jeden web mají dvě minuty. Odpověď 401 (přihlašovací brána) kontrolu neudělá nálezem, jen se nepoužije. Prostředí s klientským certifikátem (mTLS) se nekontrolují. Místo true jde nastavit:

```
health:
site:
paths: [/, /cs/, /eshop/] # cesty, které robots.txt musí nechat otevřené (výchozí: / a
úvodní cesta každého jazyka)
ai_crawlers: false # blokovat AI roboty je rozhodnutí webu: HC1103 vynechat
sitemaps: 20 # nejvíc souborů sitemap na web (výchozí a nejvýš 50)
environments: [production] # na jiných prostředích se kontroly celého webu přeskočí
(výchozí: všude)
```

kaloko health ukáže řádek za každý web a jeho nálezy pod packem site jako site:<host>:HC1104; manifest běhu je drží jako site. Kontroly celého webu nejde převzít jako kritéria.

Závažnost nálezu kondice

Neúspěšná kontrola je chyba, když lidé nebo vyhledávače zůstanou venku: HC107 (noindex), HC108 (viewport), HC110 (přetékání), HC111, HC501 a HC606 (chyby v konzoli), HC301–HC303, HC305, HC310 a HC312 (názvy, popisky, kontrast, past pro klávesnici), HC601 a HC604 (falešná 404, výpis chyby), HC701 a HC704 (popisky v aplikaci, pád), HC1102 (robots.txt blokuje procházené úvodní stránky, na production). Každá jiná neúspěšná kontrola je varování a odpověď, kterou si evaluátor nebyl jistý, je nejistá.

Příkazy CLI

Všechny příkazy CLI kaloko a co dělají. Spouštějí se jako npx kaloko <příkaz> v projektu, který má balíček nainstalovaný. npx kaloko help vypíše stejný seznam se všemi volbami.

Volby, které fungují všude: --run <id|složka> (nebo KALOKO_RUN) vybere běh, --config <soubor> konfiguraci, --json vypíše data pro skripty (Výstup pro skripty níže) a KALOKO_AGENT=<jméno> označí, co zapsal agent. kaloko help <příkaz> vysvětlí jeden příkaz, kaloko help export run jeden podpříkaz.

Nastavení projektu

| Příkaz | Co dělá | |---|---| | kaloko init | Vytvoří kaloko.config.yml, nainstaluje skill pro Claude Code nebo Codex (--agent), pozná framework, dev server, jazyky a hlavní cesty a připraví první scénář; --docs=off\|changelog\|full založí changelog a složky dokumentace. --project, --project-name a --product pojmenují projekt v Kaloku. V repozitáři s více weby je vypíše a cesty nehádá: --site jeden vybere, --base-url načte stránky z běžícího webu (sitemapa, potom odkazy z úvodní stránky), --skeleton zapíše scénář s TODO | | kaloko init --seo <web> | Týdenní hlídání SEO a odpovědních enginů na veřejném webu: přečte robots.txt a sitemapy, pozná jazyky a největší obsahové rodiny (první úsek cesty, v každém jazyce přeložený) a zapíše qa/flows/seo-<web>.yml s jedním rotujícím stack krokem pro každou skupinu. --families, --limit (stránek z rodiny a jazyka za týden), --packs seo,aeo,a11y,perf, --env, --out, --force | | kaloko login | Přihlásí tento počítač přes prohlížeč a uloží token (nejvýš na 30 dní) do ~/.config/kaloko/.env; s --print ho jen vypíše | | kaloko doctor | Zkontroluje Node, Chrome, konfiguraci, prostředí, porty prohlížeče, token, klíč evaluátoru a nástroje pro aplikace; --fix si na macOS řekne o oprávnění pro snímání | | kaloko draft <url…> | Připraví první plán a scénář jen pro čtení pro veřejné stránky, s packy seo, a11y, perf a console | | kaloko validate <soubor…> | Zkontroluje scénáře proti schématu a jejich odkazy; --coverage porovná kritéria s akceptačním plánem; --json vypíše problémy každého souboru | | kaloko scenarios | Vypíše scénáře, které konfigurace najde | | kaloko ci github\|gitlab | Napíše workflow, které u každého pull requestu projde dotčené kroky na preview nasazení a napíše k němu komentář | | kaloko ci github --release | Napíše workflow, které zapíše každý release vydaný vaším nástrojem na releasy | | kaloko ci preview-url | V CI najde adresu preview, pull request a jeho základ | | kaloko ci plan | V CI nebo v sandboxu agenta: řekne kontrole Kaloko v pull requestu, kterých scénářů se commit týká |

Průchod a snímání

| Příkaz | Co dělá | |---|---| | kaloko start | Založí běh scénáře --scenario v prostředí --env a nastaví ho jako aktuální; --locales, --viewports, --browsers, --color-schemes, --devices a další určí, co se nasnímá; --states omezí stavy kroků, --token-modes režimy tokenů designu; --pii redact nebo pseudonymize skryje osobní údaje ve všech snímcích; --ref CC-2026-001 (lze opakovat, v CI KALOKO_REFS) uvede záznamy, ke kterým běh patří, --purpose dry_run, formal nebo smoke (KALOKO_PURPOSE) řekne, proč se šlo | | kaloko walk | Spustí kroky v Playwrightu, nasnímá je a vyhodnotí; --affected, --record, --heal, --stability N, --concurrency N, --ephemeral | | kaloko capture --step <id> | Nasnímá aktuální stránku pro krok; --state empty jako jeden z jeho stavů, --crop přiloží prvek, --image převezme obrazovku z jiného nástroje | | kaloko open --step <id> | Otevře adresu kroku ve sdíleném prohlížeči | | kaloko mail --step <id> | Zachytí e?mail ze schránky prostředí, nebo uloženou zprávu přes --file | | kaloko browser start\|stop\|status | Jeden sdílený prohlížeč na prostředí, který ovládají agenti, skripty i lidé | | kaloko browser install | Stáhne prohlížečová jádra (--browsers chromium,webkit,firefox) | | kaloko devices | Vypíše předvolby zařízení, které lze použít jako viewporty | | kaloko session reset | Začne jako nový návštěvník: smaže cookies a úložiště ve sdíleném prohlížeči | | kaloko auth save | Člověk se jednou přihlásí (SSO, MFA) a průchod relaci použije znovu; kaloko auth list ukáže uložené relace | | kaloko affected | Řekne, kterých kroků se změněné soubory týkají a proč; s --fail-if-none skončí kódem 1, když žádného (--check je starší název) | | kaloko import <cesta> | Rozloží screenshoty, reporty a trace z Playwrightu nebo průchod jiného agenta jako běh | | kaloko status | Ukáže, co aktuální běh nasnímal |

Výsledky a kontrola

| Příkaz | Co dělá | |---|---| | kaloko evaluate | Spustí deterministické kontroly a sémantický evaluátor pro všechny

snímky; --recheck po změně scénáře, --budget N nejvýš N placených dotazů na evaluátor | | kaloko build | Zapíše run.json a místní náhled | | kaloko preview | Sestaví náhled a vypíše cestu k němu | | kaloko share | Nahraje běh do kaloko.app; --pr napíše komentář do pull requestu nebo merge requestu. V běhu, který skrývá osobní údaje, nahrávání zastaví nezamaskovaný e?mail, telefon, rodné číslo nebo IBAN a vypíše krok a prvek (--allow-pii běh nahraje i tak); ostatní běhy se nahrají s upozorněním | | kaloko review | Co na sdíleném běhu ještě musí rozhodnout člověk, a odkaz do režimu kontroly | | kaloko health | Kondice běhu (kontroly z packů, které jen radí) a co se změnilo od minulého běhu; --remote pro sdílený běh, --fail-on error ukončí příkaz kódem 1, když jsou v kondici chyby | | kaloko health adopt <krok>:<kontrola> | Udělá z kontroly kondice kritérium přijetí kroku v souboru scénáře, beze změny nebo s --text | | kaloko health pull | Zapíše do souboru scénáře kontroly, které lidé převzali na kaloko.app | | kaloko feedback | Otevřené připomínky recenzentů, vrácené kroky jako první | | kaloko comment | Komentuje, odpovídá, uzavírá a znovu otevírá připomínky z terminálu | | kaloko compare | Textový rozdíl dvou běhů: stavy, verdikty, skóre a důkazy | | kaloko stability | Najde oblasti, které se mění při každém běhu; --apply je přidá do scénáře, --pull převezme oblasti nakreslené na kaloko.app | | kaloko calibrate | Porovná sémantický evaluátor s hodnocením lidí a navrhne prahy | | kaloko signoff <run-id> | Stáhne podepsaný záznam o přijetí přijatého běhu; --package stáhne balíček dokladů kteréhokoli nahraného běhu (záznam, matice dohledatelnosti, auditní stopa, snímky, podepsaný manifest); --verify <soubor> ověří záznam nebo balíček i bez tokenu | | kaloko evidence <run-id> | Doklady běhu, nebo releasu s --release <verze> [--product p] (Business; --ref a --purpose ponechají jen jeho běhy jednoho záznamu nebo účelu): stav přijetí a podpisu, kritéria tak, jak se procházela, s odkazy na požadavky, výsledky, otisky snímků, rozhodnutí, podpisy, auditní stopa; --csv vypíše matici dohledatelnosti, --json celý dokument | | kaloko video export | Udělá z videí běhu jedno video; kaloko video list, trace a tools ukážou videa, průběh kroků a nástroje | | kaloko screens | Hledá v nejnovějších screenshotech všech kroků napříč vašimi flow | | kaloko inbox | Vaše upozornění na kaloko.app |

Sdílené běhy

| Příkaz | Co dělá | |---|---| | kaloko projects | Projekty organizace | | kaloko projects create <slug> | Založí projekt s názvem, který lidé uvidí (--name "ECO Podcasty"), případně omezený nebo jako součást produktu; existující projekt odmítne a nepřejmenuje | | kaloko projects rename <slug> | Změní název, který lidé vidí; dělá to správce projektu | | kaloko projects sync | Srovná Kaloko s project:, project_name: a product: v konfiguraci: chybějící projekt založí, jinak pojmenovaný přejmenuje a přidá ho do produktu | | kaloko runs | Sdílené běhy, u každého scénáře ten nejnovější; --all vypíše všechny, stránku po stránce až do konce; --limit N skončí po N a --cursor pokračuje odtamtud; --ref CC-2026-001 vypíše všechny běhy daného záznamu, --purpose formal jen formální běhy | | kaloko trash <run-id…> | Přesune běhy do koše; kaloko restore <run-id> jeden vrátí | | kaloko prune | Ponechá nejnovější běhy každého scénáře a prostředí a ostatní vypíše ke smazání; --yes je přesune do koše | | kaloko keep <run-id> | Ponechaný běh nikdy nevyprší ani se neuklidí; --off ho uvolní | | kaloko archive <scenario-id> | Skryje vyřazený scénář ze seznamu; jeho běhy zůstanou | | kaloko scenario move <scenario-id> --project <slug> | Přesune scénář do jiného existujícího projektu i se všemi běhy, koncepty designu a obrazovkami, stejně jako Nastavení scénáře na jeho stránce v aplikaci. Přesun má přednost před project: v konfiguraci i ve scénáři: další sdílené běhy scénáře půjdou do nového projektu, dokud ho nepřesunete zpátky | | kaloko public | Zpřístupní sdílený běh bez přihlášení | | kaloko export run\|steps | run stáhne sdílený běh jako zip; steps --scenario <id> nejnovější snímek každého kroku (volby --run a --scenario fungují dál) | | kaloko export findings | Seznam oprav z lokálního běhu: každé nesplněné kritérium a health kontrola po stránkách a jazycích s důkazem, s edit_url a content_ref stránky a s příkazem, který ji projde znovu; s traffic v konfiguraci i kliknutí, zobrazení a relace stránky a impact, od největšího; --format csv\|json, --locale, --step, --out |

Jira, Linear a hostované běhy

| Příkaz | Co dělá | |---|---| | kaloko issue create | Úkol v Jiře nebo Linearu z vráceného kroku nebo nesplněného kritéria; kaloko issue list je vypíše | | kaloko schedule add | Kaloko každý den nebo týden otevře stránky scénáře jen pro čtení a běh nasdílí; --health-alert error\|warning\|off určí, kdy zhoršená kondice upozorní tým | | kaloko schedule list\|run\|pause\|resume\|remove | Správa naplánovaných běhů | | kaloko schedule hook <id> [--remove] | Adresa a tajemství, kterými nasazení nebo CI spustí běh na serveru Kaloka; tajemství se ukáže jednou | | kaloko schedule export github | Workflow pro GitHub Actions, které scénář pravidelně projde celým CLI |

Produkty, changelog a dokumentace

| Příkaz | Co dělá | |---|---| | kaloko products sync | Pošle produkty z konfigurace do Kaloka, aniž by zapsal release | | kaloko products list | Produkty, které v Kaloku čtete, a produkty z konfigurace, které tam ještě nejsou | | kaloko products remove <product> | Archivuje produkt: zmizí z Produktů, jeho novinky a dokumentace vrátí 404, vydání a dokumentace zůstanou uložené | | kaloko products restore <product> | Vrátí archivovaný produkt i s vydáními a dokumentací | | kaloko changelog draft | Napíše sekci Unreleased z pull requestů a změněných obrazovek; --apply ji zapíše do CHANGELOG.md | | kaloko release [<verze>] | Zapíše release se záznamem z changelogu, obrázky a dokumentací | | kaloko docs publish | Pošle složku s dokumentací do Kaloka jako dokumentaci jedné verze | | kaloko docs check | Rozbité odkazy, zastaralé sekce, chybějící překlady a položky changelogu bez obrázku; při chybách skončí kódem 1 | | kaloko export docs | Dokumentace a changelog jako Markdown (výchozí), nebo s --format jako Docusaurus, VitePress, MkDocs, HTML, PDF či JSON; kaloko export --format <f> funguje dál |

Design

| Příkaz | Co dělá | |---|---| | kaloko sync | Vymění revize návrhu se společným konceptem (--pull, --push) | | kaloko design lint | Zkontroluje složku s návrhem podle pravidel přenositelného HTML | | kaloko design import figma | Udělá z framů ve Figmě designové kroky a z proměnných tokeny | | kaloko history | Verze flow a revize kroku | | kaloko spec [krok] | Co postavit podle schváleného designového kroku: prvky, styly napojené na tokeny, stavy; --state empty pro jeden stav kroku | | kaloko drift [krok] | Čím se implementace liší od schváleného návrhu; dokud se liší, končí kódem 1 | | kaloko reference | Ukáže nebo nastaví referenci designu a srovnávací běh | | kaloko tokens pull\|push\|check\|validate | Design tokeny projektu (DTCG) | | kaloko branch list\|create\|merge | Větve konceptu návrhu | | kaloko export live\|design | live stáhne živé soubory verze do složky (předání vývoji), design verzi v otevřených formátech; --live a --design fungují dál |

Výstup pro skripty

S --json zapíše příkaz, který vypisuje data, na stdout jediný JSON dokument a nic jiného; průběh, poznámky a varování jdou na stderr. Funguje u start, status, evaluate, build, share, validate, scenarios, runs, projects, review, feedback, inbox, auth list, browser status, ci preview-url, affected, prune, devices, doctor a u dalších příkazů, jejichž nápověda ho uvádí. kaloko share --json vypíše id běhu a jeho adresu:

```
url=$(npx kaloko share --json | jq -r .url)
```

Když příkaz s --json selže, poslední řádek na stderr vypadá takto: {"error":{"message":"Cannot reach https://kaloko.app (ENOTFOUND).","code":"unreachable","next":"…"}}. code je usage, unreachable, no_token, unauthorized, failed, nebo vlastní důvod služby (třeba plan).

Příkazy z novější verze

kaloko help --json vypíše všechny příkazy s podpříkazy a volbami jako JSON ({"version":"…","commands":{"<název>":{"flags":[…],"subcommands":{…}}}}); agent ho může číst místo textu nápovědy.

Když napíšete příkaz, podpříkaz nebo volbu, kterou vaše verze kaloko nezná, podívá se na /cli/latest.json služby, se kterou pracuje (kaloko.app, nebo KALOKO_URL): tam jsou příkazy nejnovější vydané verze a kaloko si je na den uloží do ~/.cache/kaloko. Jen když je ta verze novější než vaše a má přesně to, co jste napsali, řekne vám to místo hádání překlepu:

```
There is no command "projects" in this kaloko.
kaloko projects is in kaloko 0.16.0 (you have 0.15.0). Update: npm i -D kaloko@latest
```

Návratový kód zůstává 2 a s --json je ten řádek v next chyby. Bez připojení čeká nejvýš 1,5 s a vypíše totéž co dřív. KALOKO_NO_UPDATE_CHECK=1 nebo --offline kontrolu vypne. kaloko doctor také ukáže, že je k dispozici novější kaloko.

Návratové kódy

| Kód | Význam | |---|---| | 0 | Hotovo | | 1 | Kontroly našly problémy, nebo příkaz selhal (docs check, drift, design lint, affected --fail-if-none a validate hlásí svá zjištění kódem 1) | | 2 | Chyba použití: neznámý příkaz nebo volba, chybějící argument | | 3 | Služba Kaloko není dostupná: počítač je offline, selhalo DNS, spojení bylo odmítnuto nebo přerušeno. Chybová odpověď služby je 1 | | 4 | Chybí token, nebo je neplatný, prošlý či zrušený: spusťte kaloko login |

Doby trvání

--timeout (mail), --wait (ci preview-url), --expires (share) a --older-than (prune) berou číslo ve své jednotce jako dřív (u prvních dvou sekundy, u ostatních dny), nebo číslo s s, m, h či d: --wait 10m, --expires 72h, --older-than 14d.

kaloko help vypíše všechno výše se všemi volbami.

Konfigurační soubor

kaloko.config.yml leží v kořeni repozitáře a říká CLI, kde vaše flow běží. kaloko init ho vytvoří i s komentáři, kaloko doctor ho zkontroluje. Jeho formát je qawalk.config.v1 a povinné jsou schema, org a environments.

```
schema: qawalk.config.v1
org: acme
project: shop
project_name: E-shop
scenarios: [qa/flows/*.yml]
environments:
local:
base_url: http://localhost:3000
staging:
base_url: https://staging.acme.test
access:
basic: { user_env: STAGING_USER, password_env: STAGING_PASSWORD }
accounts:
member: { user: qa@acme.test, password_env: MEMBER_PASSWORD }
production:
base_url: https://acme.com
readonly: true
blocked_paths: ["/account/**"]
walk:
browsers: [chromium]
share:
expires_days: 30
```

Klíče nejvyšší úrovně

| Klíč | Co obsahuje | |---|---| | schema | vždy qawalk.config.v1 | | org | zkratka (slug) organizace na kaloko.app | | project | projekt v Kaloku pro běhy (jeho zkratka, třeba eco-podcasty); výchozí je název repozitáře. Scénář, který někdo přesunul do jiného projektu (v aplikaci nebo přes kaloko scenario move), tam zůstane: přesun má před tímto klíčem přednost | | project_name | název, který lidé v Kaloku vidí (ECO Podcasty). První kaloko share s ním chybějící projekt založí, kaloko projects sync existující přejmenuje. Sdílení běhu nikdy nepřejmenovává, jen upozorní, když se názvy liší | | product | zkratka produktu v Kaloku, jehož je projekt součástí; kaloko projects sync do něj projekt přidá | | scenarios | masky souborů se scénáři | | output | složka běhů (výchozí tmp/kaloko) | | environments | kde flow běží (níže) | | browser | channel (kanál Playwrightu, třeba chrome; prázdný = přibalené Chromium) a port sdíleného prohlížeče (výchozí 9333) | | walk | výchozí hodnoty pro každý běh: concurrency (jazyky najednou, 1–16), browsers, color_schemes, reduced_motion, forced_colors, display_modes | | capture | výchozí hodnoty pro každý scénář: record, trace | | evaluate | budget.calls: kolik nejvýš placených dotazů na evaluátor smí jedno vyhodnocení stát (níže) | | evaluators | sémantický evaluátor jev, váš vlastní model a evaluátor obrázků vision (níže) | | pii | allow: údaje, které jako osobní jen vypadají (firemní kontakt), * znamená jakýkoli text; nemaskují se a kaloko share kvůli nim nezastaví | | share | url (výchozí https://kaloko.app), token_env (výchozí KALOKO_TOKEN), expires_days (výchozí 30) | | docs | nastavení changelogu a dokumentace (níže) | | products | co vydáváte a verzujete jako celek (níže) | | traffic | čísla ze Search Console a GA4 ke stránkám, aby šly nálezy seřadit podle návštěvnosti (níže); traffic ve scénáři ho nahradí |

Prostředí

Každý klíč pod environments je název, který použijete v kaloko start --env.

| Klíč | Co obsahuje | |---|---| | kind | app (výchozí), nebo design (složka živých HTML kroků) | | base_url | adresa, nebo jedna pro každý jazyk se záložní default; ${VAR} se doplní z proměnných prostředí (review aplikace) | | readonly | true pro produkci: CLI jen čte, nikdy se nepřihlašuje do administrací a nevytváří data | | pii | redact nebo pseudonymize: každý běh skryje osobní údaje; required: běh nevznikne, dokud je neskryje scénář nebo kaloko start --pii | | blocked_paths | masky cest, které běh jen pro čtení nikdy neotevře | | allowed_post_paths |

masky cest, kam běh jen pro čtení smí poslat POST (košík) | | may_create_data | false zakáže vytváření dat (výchozí true) | | access | ochrana před celým prostředím: basic (user_env, password_env), headers (servisní tokeny), client_certificate a origins, které je dostanou | | accounts | lidé, kteří se přihlašují, pod stejnými klíči jako account ve scénáři: user nebo user_env, password_env, totp_env, storage_state (přihlášení uložené přes kaloko auth save) | | mail | schránka: provider (mailpit, script, none), url, user_env, password_env, address, script, options, auth | | app | testovaná aplikace pro každou platformu: android, ios, electron, desktop, windows | | serve, fixtures, default_token_mode | designová prostředí: složka kroků, JSON fixtures, výchozí režim tokenů (starší název tokens_mode funguje dál) | | token_modes | designová prostředí: režimy tokenů, ve kterých se vykreslí každý krok (bez něj všechny režimy sady; false jen default_token_mode) | | origins | designová prostředí: adresy, ze kterých design načítá písma, styly a obrázky (https://fonts.googleapis.com); organizace je povolí v Nastavení -> Zabezpečení | | html_view | experimentální: ke každému snímku statický HTML pohled |

Evaluátory

| Klíč | Co obsahuje | |---|---| | evaluators.jev.api_key_env | proměnná s vaším vlastním klíčem k evaluátoru (výchozí TYPESAFE_API_KEY) | | evaluators.jev.hosted | bez místního klíče vyhodnocuje z měsíčního limitu vašeho plánu (výchozí true); false použije jen váš klíč | | evaluators.jev.thresholds | skóre pro pass a fail | | evaluators.jev.model, base_url, max_state_chars, signals, concurrency | jemnější nastavení, potřeba jen výjimečně | | evaluators.vision | druhý evaluátor, který se dívá na screenshot (poskytovatel anthropic, klíč v ANTHROPIC_API_KEY); false ho vypne | | evaluators.<název> | váš vlastní textový evaluátor: kind: openai_compatible, base_url, model, případně api_key_env, concurrency (výchozí 2), max_state_chars (výchozí 24000; delší text stránky se zkrátí), timeout_ms, response_format (json_schema, json_object nebo none) | | evaluators.text | který evaluátor odpovídá na sémantické otázky jako první: jev (výchozí) nebo název vašeho vlastního | | evaluators.escalate_to | jev: odpovědi vašeho modelu, které na verdikt nestačí, se znovu zeptají JEV. Jen tímto směrem | | evaluate.budget.calls | kolik nových dotazů na JEV (vaším klíčem nebo přes Kaloko) smí jedno vyhodnocení položit, včetně předaných odpovědí; dotazy na váš vlastní model se nepočítají. Nejdřív přijdou na řadu kroky, pak vzorek každého stacku, vybraný podle priority (nejdřív stránky, které neprošly kontrolou, pak ty kontrolované nejdávněji). Co se nevejde, zůstane not_evaluated s poznámkou; kaloko evaluate --budget N limit přepíše | | evaluators.triage | experiment: Clef pojmenuje druh problému na každé obrazovce a ve snímcích z nahrávky, jen jako rada. model (clef nebo rychlejší clef-flash), keep (od jaké pravděpodobnosti se nález ukáže, výchozí 0,7), frames_per_step (0 nahrávky vynechá), cascade (Clef rozhodne kritéria, u kterých si textový evaluátor není jistý); false třídění vypne |

Místní model v Ollamě odpoví jako první a nejasné odpovědi předá JEV:

```
evaluate:
budget: { calls: 300 }
evaluators:
ollama:
kind: openai_compatible
base_url: http://localhost:11434/v1 # LM Studio: http://localhost:1234/v1
model: qwen3:8b
text: ollama
escalate_to: jev
```

Vlastní evaluátor se použije, jen když ho text jmenuje, a text stránky odchází jen na jeho base_url. Jeho verdikty jsou na plachtě i v komentáři k pull requestu označené "vyhodnotil váš vlastní model“.

Dokumentace a produkty

| Klíč | Co obsahuje | |---|---| | docs.dir | zdroje dokumentace (výchozí docs) | | docs.changelog | soubor changelogu (výchozí CHANGELOG.md) | | docs.style | keep-a-changelog nebo release-notes | | docs.version_source | auto, git-tag, package.json, release-please, changesets, semantic-release, pyproject, cargo, gemspec, manual | | docs.languages | jazyky dokumentace; první je jazyk originálů | | docs.framework | auto, docusaurus, vitepress, mkdocs, nextra, plain | | docs.publish | kdo čte: members, domain, nebo public (poslední dvě potřebují admina) | | docs.images | viewport a locale (reader nebo jazyk) obrázků | | products.<slug> | name (text nebo pro každý jazyk), version (source, path, pattern), docs, project nebo projects,

modules, reference_env, default_scenario, publish |

Bez products je produktem projekt.

Data o návštěvnosti

traffic připojí ke stránkám, které běh prošel, čísla z vyhledávání a analytiky. Chybějící popis u nejprodávanějšího produktu pak přijde na řadu dřív než stejná chyba na stránce, kterou nikdo nenajde. Dokud ho nepřidáte, nic se neděje, a dokud nenastavíte gsc nebo ga4, Kaloko se Googlu na nic neptá.

```
traffic:
file: qa/traffic/search-console-stranky.csv # export ze Search Console nebo GA4
gsc:
property: sc-domain:acme.cz # nebo https://www.acme.cz/
credentials_env: GOOGLE_APPLICATION_CREDENTIALS
ga4:
property: "123456789"
low_ctr: { min_impressions: 100, ratio: 0.5 }
```

| Klíč | Co dělá | |---|---| | file | jeden nebo víc exportů CSV, cesta vůči souboru, který je uvádí: tabulka Stránky ze Search Console nebo přehled GA4 podle vstupních stránek. Kaloko čte české i anglické hlavičky (Stránka, Vstupní stránka, Page, URL, Landing page; Kliknutí/Clicks, Zobrazení/Impressions, CTR, Pozice/Position, Relace/Sessions, Konverze nebo Klíčové události) a čísla tak, jak je který jazyk píše (1 234, 1,234, 3,5 %) | | gsc.property | vlastnost Search Console. Kaloko se přihlásí jako servisní účet, přes searchAnalytics.query si řekne o kliknutí, zobrazení, CTR a pozici po stránkách za 28 dní končících předevčírem (délku změní days) a odpověď si pro běh uloží | | ga4.property | id vlastnosti GA4. Relace a klíčové události (conversions) podle vstupní stránky za 28 dní končících včera, přes Data API | | credentials_env | proměnná s cestou k JSON klíči servisního účtu, nebo přímo s JSONem (secret v CI). Výchozí GOOGLE_APPLICATION_CREDENTIALS. E-mail servisního účtu přidejte do vlastnosti Search Console jako uživatele a do GA4 jako čtenáře | | low_ctr | prahy kontroly HC127 (níže): stránky s aspoň min_impressions zobrazeními (výchozí 100), jejichž CTR je pod ratio (výchozí 0,5) násobkem mediánu webu |

Stránky se párují podle adresy: host bez www., cesta bez koncového lomítka a dotaz bez sledovacích parametrů (utm_*, gclid …); na schématu a kotvě nezáleží. Cesta z GA4 (/cs/eshop/zidle) se připojí, když stack prochází jeden host; stack přes víc hostů takové řádky vynechá a řekne to v stack.traffic.note. Průchod hostu, pro který data nemají vůbec žádný řádek (staging, náhled), najde čísla z produkce podle cesty, pokud ji data uvádějí jen u jednoho hostu. Sloupec CTR bez znaku % se čte jako procenta, když je v něm některá hodnota větší než 1, jinak jako podíl. Když je stránka v exportu i v odpovědi API, platí čísla z API, nesčítají se. Okno pro API se počítá od začátku běhu, takže každé sestavení jednoho běhu pracuje se stejnými čísly.

Co z toho máte:

• každá stránka stacku má v items.json traffic: { clicks, impressions, ctr, position, sessions, conversions } a index stránek na plachtě dostane sloupce Zobrazení a Kliknutí, podle kterých jde řadit;

• kaloko export findings přidá tytéž sloupce a impact a nálezy s největším dopadem dá nahoru. impact = váha × (zobrazení + 10 × (kliknutí + relace) + 100 × konverze), kde váha je 3 u chyby nebo nesplněného kritéria přijetí a 1 u varování. Stránka bez čísel má dopad 0;

• stack s packem seo dostane HC127: stránky s hodně zobrazeními, jejichž CTR je pod polovinou mediánu webu na podobných pozicích (1–3, 4–10, 11–20, každá skupina aspoň s pěti stránkami k porovnání). Obvykle to znamená, že titulek a popis ve výsledcích chtějí přepsat. Bez dat o návštěvnosti se kontrola vůbec neobjeví.

Zdroj, který nejde přečíst, Kaloko ohlásí na stderr a vynechá; běh se sestaví i tak. Klíč servisního účtu nikdy neopustí váš počítač a do běhu se nezapíše; viz Bezpečnost a vaše data.

Tajné údaje

Tajné údaje do konfigurace nikdy nepatří. Kde je nějaký potřeba, konfigurace jen pojmenuje proměnnou prostředí (password_env: STAGING_PASSWORD, { env: CF_ACCESS_CLIENT_SECRET }). Kaloko hodnotu hledá:

1. v prostředí procesu,

2. v .env vedle kaloko.config.yml,

3. v ~/.config/kaloko/.env.

Proměnná, která už je nastavená, má přednost před soubory. Hodnoty se odstraní ze zachyceného HTML, z textu stránky i z výstupu konzole. kaloko doctor řekne, které pojmenované proměnné chybí. Token pro sdílení je KALOKO_TOKEN, pokud share.token_env neříká jinak; viz Role a tokeny.

API kondice pro dashboardy

Týdenní SEO průchod několika tisíci stránek skončí jako jeden běh se stovkami nálezů. Dashboard ho nechce stahovat a počítat znovu. Tyto adresy odpovídají už sečtenými čísly: po jazycích, krocích (krok stack je jedna šablona stránek), kontrolách nebo packech, se stránkami, kterých se problém týká nejvíc, a s tím, co se změnilo od běhu předtím.

Token pro dashboard

V Nastavení -> API tokeny vytvořte token s rozsahem read:health (na Enterprise i jako servisní účet). Čte adresy z této stránky a zbytek kondice (GET …/health běhu, scénáře nebo projektu, MCP get_health); na všechno ostatní odpoví 403. Poslouží i token s rozsahem read.

```
curl -H "Authorization: Bearer $KALOKO_TOKEN" \
"https://kaloko.app/api/v1/orgs/acme/scenarios/seo-weekly/latest/health/aggregate?by=locale,
step"
```

Adresy

| Požadavek | Co vrátí | |---|---| | GET /api/v1/orgs/<org>/scenarios/<id>/latest | Nejnovější sdílený běh scénáře, který vidíte: id, datum, verdikt, počty kondice a odkazy níže. ?environment=production vybere nejnovější běh v jednom prostředí | | GET …/scenarios/<id>/latest/health/aggregate | Souhrn tohoto běhu jedním voláním | | GET …/scenarios/<id>/latest/health/diff | Rozdíl tohoto běhu jedním voláním | | GET /api/v1/orgs/<org>/runs/<run>/health/aggregate | Počty po skupinách s nejvíc zasaženými stránkami a počty z rozdílu | | GET /api/v1/orgs/<org>/runs/<run>/health/diff | Nové, opravené a pořád otevřené výsledky, každý s během, který ho viděl poprvé | | GET /api/v1/orgs/<org>/scenarios/<id>/findings | Otevřené výsledky seskupené do nálezů podle kontroly, webu, jazyka a kroku, se stavem (níže) |

MCP má totéž jako get_health_aggregate, get_health_diff (s run_id, nebo se scenario a environment) a list_findings.

Co se počítá

Jeden výsledek je jedna stránka v jednom jazyce pro jednu kontrolu. Stránka prošlá ve více viewportech se počítá jednou, s nejhorším verdiktem. Krok stack počítá každou svou stránku, obyčejný krok je stránkou sám. Výsledek je pass, fail, needs_review (hodnotitel si nebyl jistý) nebo not_evaluated. Kontroly, které se na stránku nevztahují, se nepočítají.

Kondice jen radí: tato čísla nikdy nezmění verdikt kroku ani běhu.

Souhrn

GET …/health/aggregate bere:

| Parametr | Význam | |---|---| | by | Čárkami oddělené: locale, step, check, pack. Výchozí locale,step,check. Prázdné sečte vše do jedné skupiny | | locale, step, check, pack | Jen výsledky s touto hodnotou | | top | Zasažené stránky na skupinu, 0 až 20 (výchozí 5); selhávající stránky před nejistými | | limit | Skupiny v odpovědi, 1 až 5000 (výchozí 500), nejvíc selhávající první |

```
{
"schema": "kaloko.health-aggregate.v1",
"run": { "id": "2026-10-12-seo-weekly-a1b2c3", "scenario": "seo-weekly", "title": "Týdenní
SEO", "project": "web", "environment": "production", "started_at":
"2026-10-12T05:00:00.000Z", "canvas":
"https://kaloko.app/acme/runs/2026-10-12-seo-weekly-a1b2c3/#health", "aggregate_url": "…",
"diff_url": "…" },
"by": ["locale", "step", "check"],
"filter": {},
```

```
"totals": { "results": 16800, "pass": 15910, "fail": 812, "needs_review": 64,
"not_evaluated": 14, "pages": 2800 },
"checks": { "HC112": { "text": "The page carries structured data (JSON-LD)", "pack": "seo",
"level": "warning" } },
"diff": { "previous": { "id": "2026-10-05-seo-weekly-d4e5f6", "started_at":
"2026-10-05T05:00:00.000Z" }, "new": 14, "fixed": 3, "still_open": 859, "unchecked": 0,
"worse": 1, "expired": 0, "open": 876 },
"group_count": 168,
"truncated": false,
"groups": [
{
"id": "locale=pl;step=eshop;check=HC112",
"locale": "pl", "step": "eshop", "check": "HC112",
"results": 400,
"counts": { "pass": 210, "fail": 190, "needs_review": 0, "not_evaluated": 0 },
"affected": 190,
"top": [
{ "step": "eshop", "locale": "pl", "check": "HC112", "page": "sklep-kubek", "path":
"/pl/sklep/kubek", "url": "https://www.example.com/pl/sklep/kubek", "verdict": "fail",
"evidence": "no JSON-LD block on the page" }
]
}
]
}
```


• id skupiny se skládá z jejích hodnot, takže stejná skupina má každý týden stejné id: použijte ho jako klíč v dashboardu.

• affected je fail plus needs_review.

• checks vyjmenuje každou kontrolu běhu s textem, packem a úrovní (error nebo warning, když selže).

• diff je shrnutí rozdílu níže; previous je null u prvního běhu scénáře v daném prostředí.

Rozdíl

GET …/health/diff porovná výsledky, které selhávají nebo jsou nejisté, s tím, co měl scénář otevřené po svém předchozím sdíleném běhu ve stejném prostředí a projektu. Běhy stejného scénáře v jiném projektu se nikdy neporovnávají.

Výsledek je opravený (fixed), jen když ho tento běh zkontroloval a prošel. Výsledek, který tento běh nezkontroloval, zůstává otevřený a je v seznamu nezkontrolované (unchecked): stránka mimo tento týdenní výběr (rotate), vynechaný jazyk, kontrola, o které hodnotitel nedokázal rozhodnout. Drží si first_seen a rozhodne další běh, který ho zkontroluje. Výsledek, který 8 běhů po sobě nebo 60 dní nikdo nezkontroloval, vypadne: naposledy se objeví s "expired": true a započítá se do expired.

| Parametr | Význam | |---|---| | state | new, fixed, still_open nebo unchecked; bez něj všechny čtyři seznamy | | locale, step, check, pack | Jen výsledky s touto hodnotou | | limit, offset | Stránkování v každém seznamu: nejvýš 2000 (výchozí 200) |


```
{
"schema": "kaloko.health-diff.v1",
"run": { "id": "2026-10-12-seo-weekly-a1b2c3", "…": "…" },
"previous": { "id": "2026-10-05-seo-weekly-d4e5f6", "started_at": "2026-10-05T05:00:00.000Z"
},
"counts": { "new": 14, "fixed": 3, "still_open": 859, "unchecked": 0, "worse": 1, "expired":
0 },
"open": 876,
"truncated": false,
"new": {
"total": 14,
"items": [
```

```
{ "id": "eshop|pl|HC112|sklep-kubek", "step": "eshop", "locale": "pl", "check": "HC112",
"page": "sklep-kubek", "url": "https://www.example.com/pl/sklep/kubek", "severity":
"warning", "text": "The page carries structured data (JSON-LD)", "first_seen": { "run":
"2026-10-12-seo-weekly-a1b2c3", "at": "2026-10-12T05:00:00.000Z" } }
]
},
"fixed": { "total": 3, "items": [] },
"still_open": { "total": 859, "items": [] },
"unchecked": { "total": 0, "items": [] }
}
```


• id výsledku je step|locale|check|page. Zůstává stejné, dokud krok, jazyk, kontrola a stránka drží svá id, takže jednu stránku sledujete týden po týdnu. U obyčejného kroku je page null.

• severity je error nebo warning u selhávající kontroly a info, když si hodnotitel nebyl jistý. Pořád otevřený výsledek, kterému se závažnost změnila, má ve was tu předchozí; worse počítá ty, které se zhoršily.

• first_seen je běh, ve kterém výsledek začal selhávat a od té doby v žádném běhu neprošel. Výsledek, který byl opravený a vrátil se, začíná znovu.

• open počítá, co je po tomto běhu otevřené: co teď selhává nebo je nejisté, plus nezkontrolované.

• Web s víc než 50 000 otevřenými výsledky v jednom běhu si nechá 50 000 nejhorších a řekne truncated: true; zbytek se počítá jako nezkontrolovaný, nikdy jako opravený.

Nálezy

Osm set selhávajících stránek bývá tucet problémů: jedné šabloně v jednom jazyce chybí strukturovaná data. GET /api/v1/orgs/<org>/scenarios/<id>/findings seskupí, co má scénář otevřené, do nálezů: jedna kontrola na jednom webu v jednom jazyce na jednom kroku. Otisk (fingerprint) je check|site|locale|step, kde web je origin URL stránky (https://www.example.com). Týden po týdnu zůstává stejný, takže nález se počítá, rozhoduje a sleduje jednou, ať zasáhne kolik chce stránek. Nálezy čte i token read:health (MCP list_findings).

| Parametr | Význam | |---|---| | environment | Prostředí; bez něj nejnovější se zaznamenanou kondicí | | locale, step, check | Jen nálezy s touto hodnotou | | status | open, acknowledged, ignored nebo fixed | | limit, offset | Stránkování: nejvýš 1000 (výchozí 200) |


```
{
"schema": "kaloko.health-findings.v1",
"scenario": "seo-weekly", "project": "web", "environment": "production",
"run": { "id": "2026-10-12-seo-weekly-a1b2c3", "started_at": "2026-10-12T05:00:00.000Z",
"canvas": "…" },
"page": "https://kaloko.app/acme/health/seo-weekly?environment=production",
"counts": { "open": 9, "acknowledged": 2, "ignored": 1, "fixed": 3 },
"filter": {}, "total": 15, "offset": 0, "limit": 200,
"findings": [
{
"id": "5f0c2a9e1b7d4c33",
"fingerprint": "HC112|https://www.example.com|pl|eshop",
"check": "HC112", "text": "The page carries structured data (JSON-LD)", "pack": "seo",
"site": "https://www.example.com", "locale": "pl", "step": "eshop",
"severity": "warning", "status": "acknowledged", "reason": null,
"status_by": "jana@example.com", "status_at": "2026-10-12T09:14:00.000Z",
"count": 190, "unchecked": 0, "fixed": 4,
"samples": [
{ "page": "sklep-kubek", "url": "https://www.example.com/pl/sklep/kubek", "edit_url":
"https://cms.example.com/pages/812/edit" }
],
"first_seen": { "run": "2026-09-28-seo-weekly-77aa01", "at": "2026-09-28T05:00:00.000Z" },
"last_seen": { "run": "2026-10-12-seo-weekly-a1b2c3", "at": "2026-10-12T05:00:00.000Z" },
"issue": { "tracker": "jira", "key": "WEB-412", "url":
"https://acme.atlassian.net/browse/WEB-412", "state": "open" }
```

```
}
]
}
```

• id je krátký hash otisku; používejte ho v odkazech a v PATCH.

• severity je nejhorší z výsledků nálezu; count je počet stránek, kterých se teď týká. unchecked počítá stránky, které nejnovější běh nezkontroloval (zůstávají otevřené), fixed stránky, které nejnovější běh opravil.

• samples ukáže až 10 stránek s edit_url, které uvádí jejich položka stacku, takže obsahový tým jde rovnou do CMS (Najděte, co na webu selhává).

• first_seen je nejstarší první výskyt z jeho výsledků. last_seen je nejnovější běh, který ho zkontroloval a pořád selhával; u opraveného nálezu je to běh před opravou.

• status je fixed, jakmile žádná jeho stránka neselhává; jinak to, co nastavil člověk (acknowledged, ignored s důvodem v reason), nebo open. Rozhodnutí patří k otisku: ignorovaný nález, který se vrátí, zůstává ignorovaný.

• Když má stack traffic: (Opravte nejdřív stránky, které lidé opravdu vidí), má nález traffic (clicks a impressions sečtené přes jeho stránky, které čísla mají, a kolik těch stránek je, pages) a impact a každá ukázková stránka svoje traffic. Bez dat o návštěvnosti tam žádný z klíčů není.

• issue.key je WEB-412 v Jiře nebo Linearu, acme/website#87 na GitHubu a MKT-12 v Plane.

• Nálezy jdou od nejhoršího, opravené na konci.

Nález potvrdíte nebo ignorujete přes PATCH …/scenarios/<id>/findings/<nález>[?environment=] s { "status": "acknowledged" }, { "status": "ignored", "reason": "Starý blog, v listopadu se stěhuje do nového CMS" } nebo { "status": "open" }. Smí to každý, kdo smí komentovat běhy scénáře; token potřebuje rozsah comment (token read:health nic nemění). Stejnou tabulku ukazuje aplikace v Kondice -> Nálezy u každého scénáře, s tlačítky pro tytéž tři možnosti.

Issues k nálezům

Scénář může pro každý nový nález založit issue v Jiře, Linearu, GitHub Issues nebo Plane. Zapíná to Developer / Tester přes PUT …/scenarios/<id>/findings/sync:

```
{ "tracker": "jira", "min_severity": "error", "environment": "production" }
```

V Jiře a Linearu jde issue do projektu nebo týmu, na který je projekt scénáře namapovaný v Nastavení -> Integrace (Propojte trackery a kanály). U GitHubu a Plane je cíl přímo v nastavení a routes pošlou část nálezů do jiného trackeru než zbytek scénáře:

```
{
"tracker": "github", "min_severity": "warning",
"github": { "repository": "acme/website", "labels": ["seo"], "assignee": "jana" },
"plane": { "project": "MKT", "labels": ["content"] },
"routes": [{ "tracker": "plane", "checks": ["HC127", "HC9*"], "packs": ["aeo"] }]
}
```

• github: repozitář (owner/repo), štítky a volitelně komu issue přiřadit. Kaloko issue založí jako aplikace Kaloko na GitHubu, když je na repozitáři nainstalovaná a má oprávnění Issues; jinak fine-grained tokenem, který připojil admin (Issues: read and write na ten repozitář).

• plane: identifikátor projektu (MKT) a štítky, které v projektu už existují.

• Repozitář i projekt v Plane musí být na seznamu, který pro projekt scénáře povolil admin organizace v Nastavení -> Integrace (projekt bez vlastního řádku použije výchozí). Cokoli jiného Kaloko odmítne s 403 dřív, než se zeptá GitHubu nebo Plane, takže odpověď neprozradí, jestli to existuje. Když admin cíl ze seznamu odebere, nová issues tam už nepůjdou; ta existující Kaloko sleduje dál.

• routes: nejvýš 20, v pořadí. Rozhodne první cesta, která jmenuje kontrolu nálezu (HC127, nebo začátek jako HC1*) nebo jeho balíček (seo, aeo…); všechny ostatní nálezy jdou do tracker. Každý tracker z cest musí být připojený. Když se cesty změní, existující issue zůstane, kde je, a Kaloko ho sleduje tam.

Při uložení Kaloko u trackeru ověří repozitář, přiřazeného člověka, projekt i štítky a řekne, co nesedí.

Po každém sdíleném běhu scénáře Kaloko založí jedno issue na každý nový nález se závažností aspoň

min_severity (error, warning nebo info), nikdy dvě pro jeden otisk a žádné pro ignorovaný. Issue říká, která kontrola kde selhává, na kolika stránkách, ukáže až deset z nich s odkazy na úpravu a s traffic: i kliknutí a zobrazení každé stránky a celého nálezu s jeho dopadem; odkazuje zpět na nález. Když se počet zasažených stránek změní aspoň o tři a o čtvrtinu, issue dostane komentář; když je nález opravený, komentář a issue se zavře (Jira: první přechod do Done; Linear: první dokončený stav týmu; GitHub: zavřené jako dokončené; Plane: první dokončený stav projektu). Když se vrátí, totéž issue dostane komentář a znovu se otevře, pokud to tracker dovolí (Plane: první stav k zahájení).

Jeden běh založí nejvýš 20 issues, nejhorší první, a do trackerů zavolá nejvýš 60krát během 20 sekund, z toho nejvýš 25 požadavků do Plane, počítá se každý (jeho API bere 60 za minutu; zavření issue navíc jednou za běh načte stavy projektu); zbytek přijde s dalším během. Nález, který tracker odmítne (třeba kvůli poli, které nebere), si chybu nechá v seznamu a ostatní pokračují. Když tracker odmítne spojení, chce zpomalit nebo neodpovídá, běh pro ten tracker skončí a zkusí to další běh; nálezy pro jiný tracker pokračují. Založení, které se do trackeru vůbec nedostalo (spojení nešlo obnovit, tým v Linearu neexistuje, adresa Plane je odmítnutá), zkusí nález znovu s dalším během. Když tracker jasně neřekne, jestli issue založil (vypršel čas nebo chyba serveru), nález ukazuje issue jako zakládané s důvodem v issue.error a Kaloko druhé nezaloží: podívejte se do trackeru, a když tam nic není, zvolte u nálezu na stránce nálezů Zkusit znovu (jde to, až issue čeká dvě minuty), nebo pošlete POST …/scenarios/<id>/findings/<finding>/retry[?environment=] (Developer / Tester); další běh ho založí. Zakládání visící déle než hodinu uvolní i uložení nastavení.

Nastavení patří ke scénáři v jeho projektu: když scénář přesunete do jiného projektu, issues přestane zakládat, dokud je tam někdo nezapne. { "tracker": null } synchronizaci vypne; GET na stejné adrese ukáže nastavení a trackery, do kterých organizace může nálezy posílat.

Běhy sdílené dřív

Počty běhu vzniknou jednou, po jeho sdílení. Běh sdílený dřív, než toto API existovalo, se spočítá při každém dotazu, což u velkého běhu může pár sekund trvat; jeho rozdíl ho porovná s během před ním a řekne "recorded": false. První běh sdílený potom začne historii svého scénáře.

Když se něco změní

Webhook health.changed pošle počty z rozdílu a první nové a opravené výsledky po každém sdíleném běhu, ve kterém se něco pohnulo: Webhooky.

Kontroly kondice

Všechny packy a všechny kontroly, které spouštějí, vypsané ze stejného kódu, jaký spouští CLI, takže seznam odpovídá verzi, kterou si nainstalujete. Kontroly kondice jen radí: plachta je ukazuje na kartě Kondice a verdikt kroku ani běhu nikdy nemění. Když na kontrole vašemu týmu záleží, převezměte ji jako kritérium (kaloko health adopt <krok>:<kontrola>); od té chvíle se počítá do přijetí.

Krok si packy zapne přes packs: [seo, a11y] v souboru scénáře (Přidat kritéria a packy). Druhy kontrol, jejich volby a id najdete v Kritéria, kontroly a packy.

Závažnost neúspěšné kontroly: chyba, když stránka zavře dveře lidem nebo vyhledávačům, jinak varování. Odpověď, kterou si evaluátor nebyl jistý, se ukáže jako nejisté.

Základy SEO (seo)

Co si ze stránky přečtou vyhledávače: titulek, popis, kanonickou adresu, hreflang, Open Graph, robots, viewport a strukturovaná data. Na kroku typu stack navíc porovná stránky mezi sebou.

Na designových prostředích, kde jsou kroky vaše vlastní HTML ze složky, je vypnutý. Pokud ho tam chcete, uveďte ho ve scénáři v health: { design_packs: [...] }.

| Id | Kontrola | Proč na ní záleží | Závažnost | Kde platí | |---|---|---|---|---| | HC101 | Právě jeden nadpis H1 | Vyhledávače i čtečky obrazovky berou H1 jako název stránky; žádný nebo několik ho znejasní. | varování | každý krok s packem | | HC102 | Titulek má 10 až 70 znaků | Titulek je modrý odkaz ve výsledcích hledání; příliš krátký nic neřekne, příliš dlouhý se ořízne. | varování | každý krok s packem | | HC103 | Popis stránky má 50 až 160 znaků | Meta popis je úryvek pod odkazem; bez něj vyhledávač vybere náhodný text. | varování | každý krok s packem | | HC104 | Kanonický odkaz je uvedený | Bez kanonické adresy si kopie stránky (parametry, filtry) konkurují ve vyhledávání. | varování | každý krok s packem | | HC105 | hreflang pro každý jazyk | hreflang říká vyhledávačům, kterou jazykovou verzi komu ukázat. | varování | scénáře s více jazyky | | HC106 | Open Graph titulek a popis | Sdílené odkazy na sítích a v chatech berou náhled z Open Graph. | varování | každý krok s packem | | HC107 | Stránka není vyřazená z indexace | Značka robots s noindex stránku úplně vyřadí z výsledků hledání. | chyba | každý krok s packem | | HC108 | Viewport pro telefony | Bez značky viewport ukazují telefony zmenšenou desktopovou stránku. | chyba | každý krok s packem | | HC109 | Atribut lang u html | Čtečky obrazovky podle <html lang> volí hlas a vyhledávače jazyk. | varování | každý krok s packem | | HC110 | Nic nepřetéká do strany | Obsah širší než obrazovka nutí telefony posouvat stránku do strany a rozbíjí rozvržení. | chyba | každý krok s packem | | HC111 | V konzoli nejsou chyby | Chyby v konzoli často znamenají, že něco tiše nefunguje. | chyba | každý krok s packem | | HC112 | Strukturovaná data (JSON-LD) jsou na stránce | Strukturovaná data říkají vyhledávačům a AI asistentům, co stránka je (produkt s cenou, článek, FAQ), a otevírají rozšířené výsledky hledání. | varování | každý krok s packem | | HC113 | Každý blok JSON-LD je platný JSON | Blok, který nejde přečíst, se ignoruje celý: cena, drobečková navigace ani FAQ z něj se do výsledků hledání nedostanou. | varování | každý krok s packem | | HC121 | Titulky se v rámci jazyka neopakují | Stránky se stejným titulkem vypadají pro vyhledávač jako duplicity; vybere jednu a ostatní schová. | varování | kroky typu stack, napříč stránkami | | HC122 | Meta popisy se v rámci jazyka neopakují | Popis zkopírovaný na více stránek hledajícímu nic neřekne o té, kterou se chystá otevřít. | varování | kroky typu stack, napříč stránkami | | HC123 | Nadpisy H1 se v rámci jazyka neopakují | Dvě stránky se stejným H1 si konkurují ve stejném hledání a matou lidi, kteří přijdou na tu druhou. | varování | kroky typu stack, napříč stránkami | | HC124 | Každá stránka je svou kanonickou adresou | Kanonická adresa jinam žádá vyhledávač, aby stránku vynechal; když míří na chybovou stránku nebo noindex, vypadnou obě. | varování | kroky typu stack, napříč stránkami | | HC125 | Jazykové verze na sebe přes hreflang odkazují navzájem | Vyhledávač hreflang bez potvrzení z druhé strany ignoruje a ve výsledcích se ukáže špatný jazyk. | varování | kroky typu stack, napříč stránkami | | HC126 | Stránky ze sitemapy odpovídají 200, nepřesměrují a jdou indexovat | Sitemapa je seznam stránek, které chcete nechat najít; přesměrování, chyby a noindex v ní plýtvají procházením a podkopávají důvěru v celý seznam. | varování | stacky načtené ze sitemapy | | HC127 | Často zobrazované stránky nemají výrazně nižší míru prokliku | Lidé stránku ve výsledcích vidí, ale kliknou jinam: titulek a popis ve výsledcích neslibují to, co hledají. | varování | stacky s daty o návštěvnosti |

Odpovědní vyhledávače (aeo)

Co převezmou AI vyhledávače a asistenti: strukturovaná data FAQ, krátkou odpověď pod H1, llms.txt, meta popis a stránku tak, jak ji bez JavaScriptu dostane GPTBot. Levné signály ze stránky, ne pozice v žádném asistentovi.

Na designových prostředích, kde jsou kroky vaše vlastní HTML ze složky, je vypnutý. Pokud ho tam chcete, uveďte ho ve scénáři v health: { design_packs: [...] }.

| Id | Kontrola | Proč na ní záleží | Závažnost | Kde platí | |---|---|---|---|---| | HC901 | Otázky a odpovědi na stránce mají strukturovaná data FAQPage nebo QAPage | Odpovědní vyhledávače i rozšířené výsledky převezmou otázky a odpovědi mnohem spolehlivěji, když je stránka označí jako FAQPage nebo QAPage. | varování | každý krok s packem | | HC902 | Pod H1 je krátký odstavec s odpovědí (40 až 320 znaků) | AI asistenti citují první jasnou odpověď, kterou najdou; krátký odstavec hned pod nadpisem je ten, který si vezmou. | varování | každý krok s packem | | HC903 | Soubor llms.txt je v kořeni webu | llms.txt dává jazykovým modelům krátkou mapu webu: co nabízí a které stránky číst. | varování | každý krok s packem | | HC904 | Stránka má meta popis | Bez meta popisu si vyhledávače i asistenti napíšou vlastní shrnutí z náhodného textu. | varování | každý krok s packem | | HC905 | I bez JavaScriptu dostane GPTBot titulek, kanonickou adresu, H1, strukturovaná data a většinu textu | Většina AI crawlerů JavaScript nespouští. Co se objeví až po skriptech, pro ně neexistuje, i když stránka v prohlížeči vypadá hotová. | varování | každý krok s packem |

Přístupnost (a11y)

Část WCAG 2.2 AA, kterou jde na stránce změřit, průchod tabulátorem (bez klikání a psaní) a strom přístupnosti. Audit se čtečkou obrazovky nenahradí.

Běží i na designových prostředích.

| Id | Kontrola | Proč na ní záleží | Závažnost | Kde platí | |---|---|---|---|---| | HC301 | Obrázky mají alternativní text | Bez alt textu nevidomí ani vyhledávače nevědí, co obrázek ukazuje. | chyba | každý krok s packem | | HC302 | Odkazy a tlačítka mají přístupný název | Tlačítko s ikonou bez názvu čtečka přečte jen jako "tlačítko“. | chyba | každý krok s packem | | HC303 | Formulářová pole mají popisky | Uživatel čtečky uslyší u pole bez popisku jen "editační pole“. | chyba | každý krok s packem | | HC304 | Úrovně nadpisů se nepřeskakují | Lidé se pohybují po nadpisech; skok z H2 na H4 zamlží strukturu. | varování | každý krok s packem | | HC305 | Text splňuje kontrast WCAG AA | Text s nízkým kontrastem se špatně čte na slunci i lidem se slabším zrakem. | chyba | každý krok s packem | | HC306 | Stránka má hlavní oblast (main) | Uživatelé čteček skáčou rovnou na <main>, aby přeskočili navigaci. | varování | každý krok s packem | | HC307 | Id prvků jsou jedinečná | Duplicitní id rozbijí popisky, kotvy i skripty, které prvky hledají podle id. | varování | každý krok s packem | | HC308 | Pořadí tabulátoru odpovídá stránce a dojde ke každému prvku | Kdo ovládá web klávesnicí, posouvá se tabulátorem; když fokus skáče nebo tlačítko vynechá, ztratí se nebo nemůže dál. | varování | každý krok s packem | | HC309 | Fokus klávesnice je dobře vidět (kontrast aspoň 3 : 1) | Bez viditelného rámečku fokusu uživatel klávesnice neví, kde na stránce je. | varování | každý krok s packem | | HC310 | Klávesnice nikde neuvízne | Prvek, který tabulátor nepustí ven (vložený přehrávač, vlastní menu), uživatele klávesnice uvězní. | chyba | každý krok s packem | | HC311 | Odkaz pro přeskočení na obsah nebo rychlá cesta k němu | Bez odkazu "Přeskočit na obsah“ musí uživatelé klávesnice a čteček na každé stránce protabovat celé menu. | varování | každý krok s packem | | HC312 | Ovládací prvky mají název ve stromu přístupnosti | Strom přístupnosti je to, co čtečky skutečně čtou; prvek bez názvu ohlásí jen jako "tlačítko“. | chyba | každý krok s packem | | HC313 | Orientační oblasti jsou úplné a pojmenované | Uživatelé čteček skáčou mezi hlavičkou, navigací, obsahem a patičkou; neoznačené duplicity znějí všechny stejně. | varování | každý krok s packem | | HC314 | Role a atributy ARIA jsou platné | Chybná ARIA je horší než žádná: čtečky jí věří a ohlašují, co na stránce není. | varování | každý krok s packem |

Použitelnost (usability)

Otázky pro evaluátor vybrané podle typu stránky kroku (page: landing, work, list, settings, form, doc) a popsaná formulářová pole.

Běží i na designových prostředích.

| Id | Kontrola | Proč na ní záleží | Závažnost | Kde platí | |---|---|---|---|---| | HC201 | Účel stránky je jasný hned na první obrazovce | Návštěvník se během pár vteřin rozhodne, jestli je na správném místě. | varování | typy stránek landing, doc | | HC202 | Jedna jasná hlavní akce | Několik stejně výrazných tlačítek lidi zdrží; žádné je nechá bez dalšího kroku. | varování | typ stránky landing | | HC203 | Formulářová pole mají viditelný popisek | Placeholder při psaní zmizí a lidé zapomenou, co do pole patří. | varování | každý krok s packem | | HC204 | Odkazy a tlačítka říkají, co udělají | "Klikněte zde“ nebo "Více“ neřekne, kam vede, zvlášť mimo kontext nebo ve čtečce. | varování | každý krok s packem | | HC205 | Z nadpisu je jasné, k čemu obrazovka je | Lidé mezi obrazovkami přepínají celý den; s mlhavým nadpisem musí hledat, kde vlastně jsou. | varování | typy stránek work, list, settings, form | | HC207 | Každá položka seznamu říká, co je zač a jak ji otevřít | Řádek jen s identifikátorem nebo datem nutí lidi otevírat položky jednu po druhé. | varování | typ stránky list | | HC208 | Každé nastavení říká, co změní a jak se uloží | Lidé odcházejí z nastavení a nevědí, jestli se změna projevila. | varování | typ stránky settings | | HC206 | Je jasné, jak formulář odeslat a co bude dál | Když není jasné odesílací tlačítko nebo další krok, lidé odešlou dvakrát nebo to vzdají. | varování | typ stránky form | | HC209 | Text jde rychle projít: nadpisy a krátké odstavce | Čtenář dokumentace hledá svou otázku očima; souvislý text odpověď schová. | varování | typ stránky doc |

Výkon (perf)

Laboratorní hodnoty změřené v prohlížeči průchodu: LCP, CLS, TTFB a velikost stránky. Skuteční návštěvníci na pomalejších telefonech mohou vidět jiná čísla.

Na designových prostředích, kde jsou kroky vaše vlastní HTML ze složky, je vypnutý. Pokud ho tam chcete, uveďte ho ve scénáři v health: { design_packs: [...] }.

| Id | Kontrola | Proč na ní záleží | Závažnost | Kde platí | |---|---|---|---|---| | HC401 | Největší obsah se vykreslí do 2,5 s | LCP je doba, než stránka působí hotově; Google ho používá při řazení. | varování | každý krok s packem | | HC402 | Rozvržení se při načítání nehýbe (CLS pod 0,1) | Obsah, který při načítání poskakuje, svádí k ťuknutí vedle. | varování | každý krok s packem | | HC403 | Server odpoví do 800 ms | Pomalý první bajt zdrží všechno ostatní na stránce. | varování | každý krok s packem | | HC404 | Stránka má do 3 MB | Těžké stránky jsou pomalé a drahé na mobilních datech. | varování | každý krok s packem |

Konzole (console)

Evaluátor přečte chyby v konzoli kroku a řekne, jestli se týkají toho, co krok přijímá, nebo jde o šum cizích skriptů.

Na designových prostředích, kde jsou kroky vaše vlastní HTML ze složky, je vypnutý. Pokud ho tam chcete, uveďte ho ve scénáři v health: { design_packs: [...] }.

| Id | Kontrola | Proč na ní záleží | Závažnost | Kde platí | |---|---|---|---|---| | HC501 | Chyby v konzoli se netýkají toho, co krok přijímá | Odděluje skutečné chyby od šumu cizích skriptů (analytika, rozšíření). | chyba | každý krok s packem |

Chybové stránky (errors)

Pro stránky 404, 410, 500 a údržbu: skutečný chybový stav, cesta zpět a žádné technické podrobnosti navenek. Patří na kroky, které jsou chybovými stránkami.

Na designových prostředích, kde jsou kroky vaše vlastní HTML ze složky, je vypnutý. Pokud ho tam chcete, uveďte ho ve scénáři v health: { design_packs: [...] }.

| Id | Kontrola | Proč na ní záleží | Závažnost | Kde platí | |---|---|---|---|---| | HC601 | Odpovídá chybovým stavem (4xx/5xx), ne "měkkou“ 200 | Chybějící stránka s odpovědí 200 se zaindexuje jako obsah a skryje rozbité odkazy před monitoringem. | chyba | jen chybové stránky (expect_status nebo stránka s odpovědí 4xx/5xx) | | HC602 | Odkazuje zpět na úvod | Slepá ulička návštěvníka ztratí. | varování | každý krok s packem | | HC603 | Navigace nebo hledání zůstává dostupné | Lidé na chybové stránce chtějí najít, pro co přišli. | varování | každý krok s packem | | HC604 | Žádný výpis chyby, SQL ani ladicí stránka | Výpisy chyb prozradí cesty, verze a dotazy, které pomáhají útočníkům. | chyba | každý krok s packem | | HC605 | Chybu vysvětlí lidsky a nabídne cestu dál | Klidné vysvětlení udrží lidi na webu, místo aby odešli. | varování | jen chybové stránky (expect_status nebo stránka s

odpovědí 4xx/5xx) | | HC606 | V konzoli nejsou chyby | Skripty, které na chybové stránce padají, často rozbijí i její navigaci. | chyba | každý krok s packem |

Jazykové verze (locale)

Pro vícejazyčné weby: html lang, vlastní hreflang stránky, jazyk textu a desetinný oddělovač cen odpovídají procházenému jazyku. Jen ve scénářích, které uvádějí locales.

Běží i na designových prostředích.

| Id | Kontrola | Proč na ní záleží | Závažnost | Kde platí | |---|---|---|---|---| | HC1001 | html lang odpovídá procházenému jazyku | Čtečky podle html lang volí hlas a vyhledávače jazyk; německou stránku označenou jako anglickou čtečka přečte anglickým přízvukem. | varování | každý krok s packem | | HC1002 | Vlastní hreflang stránky odpovídá jazyku, kódy jsou platné a sedí k adresám | Špatný vlastní hreflang nebo kód typu en_UK způsobí, že vyhledávač celou sadu jazykových verzí přeskočí. | varování | každý krok s packem | | HC1003 | Text je v jazyce procházené verze | Nepřeložená stránka v jazykové verzi ztratí návštěvníky a pro vyhledávač vypadá jako duplicita zdrojového jazyka. | varování | každý krok s packem | | HC1004 | Ceny používají desetinný oddělovač daného jazyka | Zápis 12.50 ? působí na německého nebo českého zákazníka divně a 12,50 $ na amerického; vypadá to, že obchod nebyl dělaný pro ně. | varování | každý krok s packem | | HC1005 | Každá stránka má v každém jazyce vlastní verzi (hreflang) | Stránka bez překladu pošle návštěvníky toho jazyka jinam a vyhledávač jim ukáže, co zbude; když víc stránek míří na jednu verzi, jde obvykle o náhradu výpisem nebo úvodní stránkou. | varování | stacky s items.variants: hreflang |

Obrazovky aplikací (app)

Nativní aplikace na Androidu, iOS a desktopu: popsané ovládací prvky, dotykové plochy, oříznutý text, pády a jasný další krok.

| Id | Kontrola | Proč na ní záleží | Závažnost | Kde platí | |---|---|---|---|---| | HC701 | Každý ovládací prvek má přístupný popisek | VoiceOver a TalkBack přečtou prvek bez popisku jen jako "tlačítko“. | chyba | každý krok s packem | | HC702 | Dotykové plochy aspoň 48 dp / 44 pt | Malé plochy se minou, hlavně palcem a u starších lidí. | varování | každý krok s packem | | HC703 | Žádné oříznuté texty (…) | Oříznutý text skryje ceny, jména nebo pokyny, často jen v delších jazycích. | varování | každý krok s packem | | HC704 | Aplikace nespadla ani nezamrzla | Pád nebo hlášení "aplikace nereaguje“ ukončí práci. | chyba | každý krok s packem | | HC705 | Účel obrazovky a další krok jsou jasné | Na malé obrazovce potřebují lidé hned vidět, co dál. | varování | každý krok s packem |

Reference designu (design)

Implementace proti schválené verzi designu: pixely, struktura a tokeny. Jediný pack, který se vždy počítá do přijetí, proto má id AC.

| Id | Kontrola | Proč na ní záleží | Závažnost | Kde platí | |---|---|---|---|---| | AC801 | Vypadá jako reference designu (rozdíl pod 3 % pixelů) | Viditelné rozdíly od schváleného designu znamenají, že se postavilo jinak, než se domluvilo. | akceptační kritérium | každý krok s packem | | AC802 | Stejná struktura jako reference designu | Chybějící nadpisy, akce nebo pole mění, co lidé mohou udělat, i když to vypadá podobně. | akceptační kritérium | každý krok s packem | | AC803 | Barvy, písma, velikosti a mezery pocházejí z tokenů | Ruční hodnoty se odchylují od design systému a zdražují pozdější změny. | akceptační kritérium | každý krok s packem |

Kontroly webu (site)

Jednou za web a běh, ne za krok: robots.txt, sitemapy, které uvádí, llms.txt a přesměrování na https a na jednu podobu domény. Vypnuté, dokud scénář nenastaví health: { site: true }, protože posílají vlastní požadavky. Nejde je převzít jako kritéria.

| Id | Kontrola | Proč na ní záleží | Závažnost | Kde platí | |---|---|---|---|---| | HC1101 | robots.txt odpovídá 200 a každý řádek dává smysl | Roboti přeskočí řádky, kterým nerozumí, takže překlep jako "Disalow“ potichu otevře nebo zavře část webu; chyba serveru procházení úplně zastaví. | varování | jednou za web | | HC1102 |

robots.txt neblokuje procházené úvodní stránky | Disallow, který zakryje úvodní stránku nebo kořen jazykové verze, vyřadí web z vyhledávání; často jde o nastavení ze stagingu, které se dostalo na produkci. | chyba na production, jinde varování | jednou za web | | HC1103 | robots.txt neblokuje AI roboty | GPTBot, ClaudeBot, PerplexityBot a další se řídí robots.txt; když jsou zablokovaní, web se v odpovědích AI neobjeví a asistenti jeho stránky uživatelům neotevřou. | varování | jednou za web, jen na production | | HC1104 | Sitemapy odpovídají, jsou platné a uvádějí úplné adresy z vlastní domény | Rozbitou nebo useknutou sitemapu, relativní adresy nebo cizí doménu vyhledávač ignoruje; vymyšlená data lastmod ho naučí ignorovat i je. | varování | jednou za web | | HC1105 | Soubor llms.txt je v kořeni webu | llms.txt dává jazykovým modelům krátkou mapu webu: co nabízí a které stránky číst. | varování | jednou za web | | HC1106 | http:// přesměruje na https:// jedním skokem | Každé přesměrování navíc stojí čas při každé první návštěvě a stránka dostupná přes http:// dělí odkazy i pozice mezi dvě adresy. | varování | jednou za web | | HC1107 | Druhá podoba domény (s www nebo bez) přesměruje jedním skokem | Když odpovídá example.com i www.example.com, vidí vyhledávač dvě kopie webu; řetěz přesměrování zdrží každou návštěvu, která přijde druhou cestou. | varování | jednou za web |

Nástroje MCP

MCP server Kaloka dovolí agentovi číst běhy, připomínky, návrhy, changelog i dokumentaci a odpovídat kontrolujícím. Je součástí plánu Team a vyšších. Zachytávání, vyhodnocení a nahrávání zůstávají v CLI.

Připojení

Server je na https://kaloko.app/mcp (MCP přes Streamable HTTP). Ověříte se API tokenem organizace v hlavičce Authorization; jak ho vytvořit: Role a tokeny.

```
claude mcp add --transport http kaloko https://kaloko.app/mcp --header "Authorization: Bearer $KALOKO_TOKEN"
```

Ostatní agenti (Codex, Cursor) dostanou stejnou adresu a hlavičku jako HTTP MCP server. Na čtení stačí token jen pro čtení; nástroje, které zapisují, potřebují token s příslušným rozsahem. Token read:health (pro dashboard) vidí jen get_health, get_health_aggregate, get_health_diff a list_findings. Screenshoty a soubory přes MCP necestují: odpovědi nesou jejich adresy, které stáhnete stejným tokenem.

list_runs, list_notifications a list_comments odpovídají po stránkách. Když odpovídá víc položek, než je limit, odpověď má next; nástroj zavolejte znovu s cursor nastaveným na tuto hodnotu. REST API stránkuje stejné seznamy přes ?cursor= a další stránku uvádí v hlavičkách X-Next-Cursor a Link.

Běhy a kontrola

| Nástroj | Co dělá | |---|---| | get_started | kde organizace stojí, její ukázkový běh a další kroky | | add_sample_run | přidá ukázkový běh, nebo vrátí ten, který už existuje | | list_projects | projekty organizace s jejich zkratkami | | upsert_project | založí projekt s názvem nebo ho přejmenuje (správce projektu) a přidá ho do produktu; jako kaloko projects create, rename a sync | | list_runs | sdílené běhy od nejnovějšího, s filtrem podle scénáře, projektu, verdiktu, autora, prostředí, větve, ref (záznam, ke kterému běh patří, třeba CC-2026-001) nebo purpose (dry_run, formal, smoke); cursor pro další stránku | | get_run_summary | jeden běh: stav každého kroku a jazyka, kritéria, která chtějí pozornost, s důkazy a zdůvodněním, adresy screenshotů, odkaz na plachtu, řádky (lanes) a role (sessions) toku a u každého kroku jeho řádek a roli | | get_stack_items | stránky stack kroku s jejich stavem a nesplněnými kritérii | | get_step_assets | nejnovější snímek každého kroku scénáře na stálých adresách (Business); state vybere stav kroku | | get_step_recording | video a záznam průběhu kroku; state vybere stav kroku | | get_criteria_verdicts | verdikty kontrolujících u ručních nebo přepsaných kritérií | | get_review_queue | o čem v běhu ještě musí rozhodnout člověk, a odkaz na kontrolu, který mu předáte | | decide_step | vlastní schválení nebo vrácení kroku agentem; označí se jako agentovo a nikdy se nepočítá jako lidské | | revoke_carried_approval | zruší schválení převzaté u kroku, na který se má někdo podívat znovu | | get_signoff_record | podepsaný záznam o přijetí běhu (Business) | | get_evidence | doklady běhu nebo releasu pro vaše vlastní dokumenty: stav přijetí a podpisu, kritéria tak, jak se procházela, s odkazy na požadavky, výsledky, otisky snímků, rozhodnutí, podpisy, auditní stopa a matice dohledatelnosti (Business) | | get_health | kondice běhu, nejnovějšího běhu scénáře nebo vývoj projektu: nálezy podle závažnosti, packu a kroku a co přibylo nebo se spravilo od minulého běhu; jen radí, verdikt nemění | | get_health_aggregate | kondice běhu (nebo nejnovějšího běhu scénáře) sečtená pro dashboard: po jazycích, krocích, kontrolách nebo packech, kolik výsledků prošlo, selhalo nebo je nejistých, nejvíc zasažené stránky a kolik je nových či opravených od minulého běhu; smí ho volat i token read:health | | get_health_diff | po stránkách, jazycích a kontrolách co je nového, opraveného nebo pořád otevřeného od minulého běhu, s během, který to viděl poprvé; smí ho volat i token read:health | | list_findings | otevřené výsledky scénáře seskupené do nálezů podle kontroly, webu, jazyka a kroku: závažnost, počet zasažených stránek, ukázkové stránky s odkazem do CMS, první a poslední výskyt, stav (otevřený, potvrzený, ignorovaný s důvodem, opravený) a issue v trackeru; jen čte, smí ho volat i token read:health | | adopt_health_check | udělá z kontroly kondice u kroku (HC103) kritérium přijetí AC103 scénáře, beze změny nebo s novým textem (Developer / Tester); do souboru scénáře ho zapíše kaloko health pull | | get_affected_steps | ke kterým krokům scénáře změněné soubory dosáhnou, a proč | | list_notifications | schránka majitele tokenu |

Připomínky a úkoly

| Nástroj | Co dělá | |---|---| | get_feedback | verdikt běhu, schválené kroky, otevřená vrácení a komentáře | | list_comments | otevřená vlákna v celé organizaci s odpověďmi, špendlíky a nakreslenými oblastmi | | add_comment | komentář ke kroku, běhu nebo celému flow, nebo odpověď | | resolve_comment | uzavře vlákno s poznámkou, co se změnilo | | reopen_comment, edit_comment, delete_comment | znovu otevře vlákno; upraví nebo smaže vlastní komentář | | get_findings | nálezy experimentálního třídění v běhu (druh problému na každé obrazovce) a odpovědi lidí na ně | | answer_finding | označí nález jako správný, jako ne-problém, nebo jako převedený na komentář | | get_triage_report | jak se třídění daří ve vaší organizaci: užitečné nálezy podle druhu, jak sedí jeho pravděpodobnosti, nálezy převedené na komentář, čas do prvního posouzení s nálezy a bez nich | | list_ignore_regions | oblasti, které lidé nakreslili, aby se vynechaly z porovnání pixelů | | add_ignore_region | navrhne takovou oblast | | create_issue | úkol v Jiře nebo Linearu z kroku, se screenshotem, kritériem a poznámkami | | list_issues | úkoly vytvořené z kroků a jejich stav | | list_schedules, run_schedule | hostované běhy a jejich limit stránek; jeden spustí hned |

Design

| Nástroj | Co dělá | |---|---| | get_draft | kterou revizi ukazuje každý krok společného konceptu a kdo ji změnil | | get_step | předání kroku: živé soubory, tokeny, struktura stránky, otevřené komentáře; u každého snímku stav kroku a režim tokenů | | get_step_spec | co se má u kroku postavit podle schváleného návrhu: prvky, styly převedené na tokeny, stavy, kontrast; state pro prázdný, chybový nebo jiný stav kroku | | get_step_drift | čím se běh implementace liší od reference designu, v jednom stavu kroku (state) | | get_step_history | verze a revize kroku s autory | | list_step_changes | kroky, které se změnily mezi dvěma referencemi nebo od posledního přijatého běhu | | get_references, get_tokens | reference designu a baseline; sada tokenů projektu | | get_draft_feedback | komentáře ke konceptu | | get_version_notes, set_version_notes | přečte nebo přepíše, co se ve verzi změnilo | | publish_version, set_reference, set_step_status, add_draft_comment | publikuje koncept jako verzi, nastaví referenci, nastaví stav kroku, okomentuje koncept |

Obrazovky, changelog, dokumentace a produkty

| Nástroj | Co dělá | |---|---| | search_screens | knihovna obrazovek: nejnovější screenshot každého kroku, hledání podle slov, projektu, produktu, jazyka nebo stavu | | list_products | produkty, které čtete, s nejnovějším vydáním, dokumentací a čtenáři | | get_changelog | vizuální changelog produktu, volitelně i s obrázky | | draft_changelog_entry | co přijaté běhy od posledního vydání ukazují jinak, jako koncept záznamu s odkazy na kroky před a po | | resolve_step_image | jeden odkaz na krok (kaloko:<scénář>/<krok>) jako adresa obrázku | | get_docs | publikovaná dokumentace produktu: verze, jazyky, postranní menu, nebo jedna stránka | | docs_check | co v publikované dokumentaci chce pozornost: rozbité odkazy, zastaralé části, chybějící překlady |

Chyby se vracejí jako výsledek nástroje označený jako chyba (neznámý běh, chybějící argument). Neplatný token dostane HTTP 401; organizace v plánu nižším než Team se dozví, že MCP server její plán nemá.

Soubor scénáře

Scénář je soubor YAML (obvykle v qa/flows/), který popisuje jedno flow: jeho kroky, šipky mezi nimi a to, co má každý krok splnit. Jeho formát je qawalk.scenario.v1; kaloko validate <soubor> soubor podle něj zkontroluje. Jak scénář napsat krok za krokem: Napsat scénář.

```
schema: qawalk.scenario.v1
id: WEB-12-checkout
title: Checkout
locales: [en, cs]
viewports: { desktop: [1280, 800], mobile: iphone-15 }
environments: [local, staging]
nodes:
- id: cart
kind: screen
title: { en: Cart, cs: Košík }
path: /cart
purpose: The shopper sees what they are buying and can continue.
execution: { strategy: playwright, script: steps/cart.mjs, instructions: Add one product and
open the cart. }
criteria:
- { id: AC1, text: Shows the total, check: { type: deterministic, assert: exists, selector:
.cart-total } }
- id: payment
kind: screen
title: Payment
path: /checkout/payment
packs: [a11y]
edges:
- { from: cart, to: payment }
```

Klíče nejvyšší úrovně

Povinné jsou schema, id, title, nodes a edges.

| Klíč | Co obsahuje | |---|---| | schema | vždy qawalk.scenario.v1 | | id | id scénáře (malá písmena, číslice, pomlčky); úkol v něm (WEB-12-…) propojí issue | | title | název pro lidi | | kind | task (výchozí) nebo flow | | archived | true skryje vyřazený scénář ze seznamu; jeho běhy zůstanou dostupné | | source | cesta k akceptačnímu plánu (ACCEPTANCE.md), používá ji kaloko validate --coverage | | version | číslo, které zvýšíte, když se flow změní | | locales | jazyky, ve kterých se prochází, např. [en, cs] | | viewports | názvy s velikostí ([1280, 800]), předvolba zařízení (iphone-15, ipad-landscape) nebo seznam předvoleb | | browsers | chromium (výchozí), webkit, firefox | | color_schemes | light, dark | | reduced_motion | no-preference, reduce | | forced_colors | none, active (vysoký kontrast ve Windows) | | display_modes | browser, standalone (nainstalovaná webová aplikace) | | devices | scénáře aplikací: emulátory, simulátory nebo telefony, na kterých se prochází | | token_modes | designová prostředí: které režimy sady tokenů vykreslit ([brand-a, brand-b]), nebo false jen pro default_token_mode prostředí; bez něj všechny režimy | | environments | prostředí z konfigurace, na kterých scénář smí běžet | | readonly | true u scénáře, který jen čte; na prostředí jen pro čtení je povinné | | platform | web (výchozí), android, ios, electron, desktop, windows | | health | acceptance: [seo]: packy, jejichž kontroly se v tomto scénáři počítají do přijetí; bez toho jsou kontroly z packů jen kondice (Kontrola kondice aplikace). site: true (nebo { paths, ai_crawlers, sitemaps, environments }) přidá jednou za běh kontroly celého webu: robots.txt, sitemapy, llms.txt, přesměrování (Kritéria, kontroly a packy). design_packs: [console]: packy jen pro kód (seo, aeo, perf, console, errors), které mají běžet i na designových prostředích, kde jsou jinak vypnuté | | traffic | čísla ze Search Console a GA4 pro stránky tohoto scénáře; nahradí traffic z konfigurace (Konfigurační soubor) | | tags | volné štítky | | refs | odkazy na požadavky, které scénář ověřuje, tak, jak je píše váš tracker nebo QMS ([URS-12, FS-3.4]); v matici dohledatelnosti platí pro každé kritérium (doklady o kvalitě) | | nodes | kroky (níže) | | edges | šipky mezi kroky (níže) | | lanes | popisky řádků na plachtě, index = číslo řádku (odkaz končící #lane=<číslo> otevře daný řádek) | | issue | úkol: number, repo (owner/name), key (Jira nebo Linear,

ENG-42), url, title | | sessions | lidé ve flow, každý s title, volitelně mailbox, account, blocked_on | | project | projekt v Kaloku pro jeho běhy; přebije konfiguraci. Když konfigurace chce po každém scénáři vlastní projekt (komentář, nebo produkt s projects:), kaloko validate a kaloko start na chybějící projekt upozorní | | product, module | který produkt a modul z products: flow ukazuje, pro changelog a dokumentaci | | capture | kvalita snímků pro všechny kroky (níže) | | pii | redact nebo pseudonymize: osobní údaje skryté ve všech snímcích, vyhledané automaticky (návod) |

Klíče kroku (nodes)

Každý krok potřebuje id, kind a title.

| Klíč | Co obsahuje | |---|---| | id | id kroku, používá se v odkazech, v --steps a v odkazech na kroky (kaloko:<scénář>/<krok>) | | kind | screen, email, external (obrazovka třetí strany), decision, stack (mnoho stránek jedné šablony) | | title | text, nebo jeden pro každý jazyk ({ en: Cart, cs: Košík }) | | path | adresa vůči base_url, nebo jedna pro každý jazyk; ve scénářích aplikací deep link; {random} vytvoří adresu, která nemůže existovat | | purpose | jedna věta o tom, k čemu krok slouží; čte ji evaluátor | | refs | odkazy na požadavky u kroku; v matici dohledatelnosti platí pro jeho kritéria | | lane | řádek na plachtě | | readonly | krok jen čte | | execution | strategy (playwright, script, agent, manual), script (cesta ke skriptu kroku), instructions (povinné), requires (např. browser) | | mail | e-mailové kroky: subject a to | | criteria | co má krok splnit (níže) | | session | který člověk ze sessions krok dělá | | affects | masky souborů, které krok ovlivňují, pro kaloko walk --affected, nebo always | | packs | hotové sady kontrol kondice: seo, usability, a11y, perf, console, errors, app, aeo, locale a design, který se počítá do přijetí | | expect_status | stav HTTP, kterým má krok odpovědět, např. 404 (nebo seznam). Stav 400 a vyšší označí chybovou stránku: pack errors kontroluje HC601 (měkká 404) a HC605 (vysvětlení) jen na chybových stránkách, tedy s expect_status nebo při odpovědi 4xx/5xx | | capture | kvalita snímků jen pro tento krok | | record | true nahraje video kroku, i když průchod nenahrává; false nikdy | | ignore | oblasti vynechané z porovnání pixelů: { selector } nebo { rect: [x, y, w, h], viewport }, volitelně omezené přes viewports, locales, s poznámkou note | | items | stack kroky: odkud se berou stránky (urls, file, sitemap (i gzip; sitemap: robots nebo robots: true čte sitemapy z /robots.txt pro procházený jazyk) s match/exclude/limit, crawl, script). match a exclude jsou regulární výraz nebo mapa podle jazyka ({ default: '^/eshop/', fi: '^/fi/tuotteet/' }) a u každého zdroje platí před limit; rotate: week nebo rotate: run vezme limit stránek každý týden nebo běh z jiného místa, takže velký seznam projde po částech; variants: hreflang (se source_locale) vezme seznam v jednom jazyce a v ostatních projde hreflang alternativu každé stránky, takže stránka a její překlady jsou ve stacku jedna stránka a chybějící překlad se nahlásí (HC1005), v kaloko walk i v hostovaných běžích; params dostane script, který dostane i node (id, title, lane, lane_title, session); skript nebo seznam v JSON může každé stránce dát edit_url (stránka v CMS) a content_ref (její záznam) a resolve: steps/resolve.mjs (výchozí export (item, ctx) => ({ edit_url, content_ref })) je doplní procházeným stránkám pro index stacku a kaloko export findings; v designovém prostředí jsou varianty složky kroku, vyjmenované v urls, nebo každá podsložka s index.html | | fields | stack kroky: až 12 sloupců indexu stacku (id, label, from: title, h1, description, canonical, lang, words, http, jsonld_types, meta tag jako og:title nebo meta:robots, CSS selektor, selektor@atribut, count:selektor) | | details | pojmenované části obrazovky pro návody: { id, name, selector }, volitelně context (section, card, page nebo CSS selektor), padding a states; v dokumentaci je ukáže kaloko:<scénář>/<krok>#<id> (Publikovat živou dokumentaci) | | states | další stavy, ve kterých se krok snímá, např. [empty, error, loading]; název může být jakýkoli (níže) |

Kritéria

Každé kritérium má id (AC1, AC12b), text pro lidi, volitelně highlight (CSS selektor, který se na screenshotu orámuje), volitelně refs (identifikátory požadavků, které ověřuje, volný text) a právě jednu kontrolu check: deterministic, semantic nebo manual. Všechny typy kontrol a jejich volby najdete v Kritéria, kontroly a packy.

environments: [production] omezí kritérium na některá prostředí, states: [empty] na některé stavy kroku. Jinde se ukáže jako nepoužitelné a krok kvůli němu nebude částečný.

Stavy kroku

Košík není jedna obrazovka: může být prázdný, nemusí se načíst nebo se ještě načítá. states: u kroku nasnímá každý z těchto stavů vedle výchozího:

```
- id: cart
kind: screen
title: Košík
path: /cart
states: [empty, error, loading]
execution: { strategy: playwright, script: steps/cart.mjs, instructions: Otevřete košík. }
criteria:
- { id: AC1, text: Ukazuje celkovou cenu, check: { type: deterministic, assert: exists,
selector: .cart-total } }
- { id: AC2, text: Říká, že je košík prázdný, states: [empty], check: { type: semantic,
question: Říká stránka, že je košík prázdný? } }
```

• Průchod spustí skript kroku jednou pro každý stav a předá mu ho jako state (nejdřív 'default'). Skript stránku do stavu dostane sám: vyprázdní košík, podvrhne chybu API, zpomalí odpověď.

• Designový krok pro stav empty vykreslí cart/empty.html (pro jazyk cart/cs/empty.html). Když stránka chybí, Kaloko otevře cart/index.html?state=empty, takže jedna stránka může stavy přepínat sama.

• Každý stav má vlastní screenshoty a verdikty. Na plachtě přibude vedle jazyka přepínač Stav kroku, režim revize i mřížka ukazují vybraný stav a krok, který takový stav nemá, ukáže svou výchozí obrazovku.

Šipky (edges)

| Klíč | Co obsahuje | |---|---| | from, to | id kroků | | label | text u šipky, třeba volba v rozhodnutí | | kind | main (výchozí), nebo alt pro alternativní cestu |

Snímky (capture)

| Klíč | Co obsahuje | |---|---| | format | webp (výchozí), nebo png (bezeztrátový, pro tisk) | | scale | 1 nebo 2 (pixely zařízení na snímku) | | quality | kvalita WebP od 0,3 do 1 | | mask | osobní údaje skryté ve všem, co krok uloží: CSS selektory, auto (vyhledávání), nebo obojí; masky scénáře a kroku se sčítají (návod) | | annotate | očíslované značky u prvků kritérií s highlight | | full_page | celá stránka (výchozí, i u stránek stacku), nebo s false jen první obrazovka: u dlouhých stránek několikrát menší | | screenshot | thumb uloží jako obrázek kroku jen malý náhled (640 px na šířku, na výšku jako zachycená stránka): pro stacky se stovkami stránek, kde kontroly čtou stránku, ne obrázek, nejmenší spolu s full_page: false; výchozí full | | record | true, step, flow, failed nebo off; viz kaloko walk --record | | trace | ke každému kroku záznam akcí, požadavků a konzole (výchozí true) |

capture kroku má přednost před scénářem a scénář před konfigurací; výjimkou jsou masky, ty se sčítají.

Shrnutí pro posouzení dodavatele

Shrnutí na jednu stránku, které může vaše QA přiložit k posouzení dodavatele. Popisuje Kaloko tak, jak je dnes; podrobnosti jsou ve stránce Doklady o kvalitě pro váš systém kvality a na stránce o důvěře.

| | | |---|---| | Produkt | Kaloko: CLI (kaloko v npm), které prochází obrazovky aplikace podle napsaných kritérií, a služba kaloko.app, která běhy ukládá, sbírá rozhodnutí lidí a podepisuje záznamy | | Dodavatel | Sinfin, kontakt na stránce o důvěře | | Zamýšlené použití v práci se systémem kvality a v regulovaných oborech | Provedení a záznam kontrol uživatelského rozhraní, sběr rozhodnutí z kontroly, ověřitelné záznamy a doklady pro vlastní systém kvality zákazníka | | Není určeno pro | Řízení dokumentů, odchylky, CAPA, záznamy o školení ani samostatné uvolnění produktu | | Stav validace | Nikdo ho nevalidoval ani necertifikoval; netvrdíme shodu s ISO 9001, ISO 13485, IEC 62304, ISO 27001, 21 CFR Part 11 ani EU GMP Annex 11. Své použití validuje zákazník | | Hosting a data | Cloudová služba; kde jsou data uložená a kteří zpracovatelé s nimi pracují, uvádí stránka o důvěře | | Ujištění o bezpečnosti | SOC 2 se připravuje, zpráva zatím není | | Řízení přístupu | Role v organizaci a v projektech; povinné dvoufázové ověření a limity relací (Business); firemní přihlášení přes SAML nebo OIDC, SCIM a povolené IP adresy (Enterprise); API tokeny s omezeným rozsahem a platností | | Auditní stopa | Log, do kterého se jen přidává, s tím, kdo, kdy, odkud a jakým klientem; auditní stopa běhu je součástí jeho balíčku dokladů | | Záznamy | Podepsané Ed25519 nad kanonickým JSON; SHA-256 každého snímku a manifestu běhu; ověřitelné i bez připojení přes kaloko signoff --verify proti zveřejněnému veřejnému klíči | | Elektronické podpisy | Volitelné, ve výchozím stavu vypnuté: význam ze seznamu zákazníka, čerstvý druhý krok, jméno, e-mail, čas s časovým pásmem, vazba na běh; změna chce důvod a zůstane zaznamenaná. Tokeny a agenti nepodepisují nikdy | | AI | Hodnotitelé s AI mohou výsledek navrhnout; jejich verdikty jsou doporučení i s názvem modelu; rozhodují lidé | | Doklady | Pro každý běh i release, jako JSON (qawalk.evidence.v1) a CSV, a jako podepsaný balíček ZIP | | Uchování | Přijaté běhy zůstávají, dokud trvá plán Business nebo Enterprise; ostatní běhy vyprší po době uchování podle plánu. Databázi lze obnovit k libovolné minutě posledních 30 dnů | | Řízení změn | Sémantické verze a veřejný changelog CLI; datové formáty s verzemi, které v rámci verze jen přibírají pole; služba se aktualizuje průběžně a každý záznam uvádí verzi služby | | Známé mezery | Žádná certifikace; zatím bez zprávy SOC 2; bez hesel (podpis stojí na přihlášení a druhém kroku); službu nejde podržet na určité verzi; zdroj času není validovaný; nejde právně zablokovat smazání |

Webhooky

Kaloko umí zavolat vaši adresu, když se s během něco stane: je sdílen, přijat nebo vrácen, krok je schválen nebo vrácen, někdo přidá komentář nebo vyřeší vlákno, doklady jsou připravené nebo se změní kondice. Dokumentace se tak přegeneruje z nejnovějších snímků a portál vašeho QMS ukáže u testovacích případů verdikt a otevřené připomínky, aniž by se musel Kaloka každou minutu ptát. Webhooky jsou součástí plánu Business a vyšších.

Nastavení

Administrátor organizace přidá adresu v Nastavení -> Integrace -> Webhooky a zaškrtne události a projekty. Každý webhook má vlastní podpisový klíč, který najdete hned u něj. Organizace může mít až 10 webhooků.

Webhook bere všechny otevřené projekty (i běhy bez projektu), nebo jen projekty, které zaškrtnete. Události omezeného projektu dostanou jen webhooky, které ho mají zaškrtnutý, takže portál klienta se dozví o jeho projektu a o ničem jiném. Komentář hosta patří k projektu běhu, na který ho napsal.

Totéž jde přes API s tokenem administrátora:

| Požadavek | Co dělá | |---|---| | GET /api/v1/orgs/<org>/webhooks | Vaše webhooky s událostmi, projekty a klíči | | POST /api/v1/orgs/<org>/webhooks | Přidá webhook: { "url", "events": [...], "projects": [...], "all_projects": true } | | PATCH /api/v1/orgs/<org>/webhooks/<id> | Změní události nebo projekty; co nepošlete, zůstane | | DELETE /api/v1/orgs/<org>/webhooks/<id> | Odebere ho | | GET /api/v1/orgs/<org>/webhooks/<id>/deliveries | Poslední doručení: událost, stav, pokusy, poslední odpověď |

Prázdný řetězec v projects znamená běhy bez projektu. Bez projects a all_projects bere nový webhook všechny otevřené projekty.

Události

| Událost | Kdy | |---|---| | run.shared | Běh byl sdílen (nahrán a dokončen) | | run.accepted | Běh byl přijat | | run.returned | Někdo vrátil celý běh; s úryvkem jeho poznámky | | review.step_changed | Krok byl schválen nebo vrácen, nebo rozhodnutí přestalo platit: smazané, vyřešené, nebo odvolané převzaté schválení | | comment.added | Nový komentář u běhu: vlákno, odpověď nebo vrácený krok s poznámkou | | comment.resolved | Vlákno je vyřešené, v Kaloku nebo tím, že jeho úkol v Jiře či Linearu je hotový | | comment.changed | Komentář byl upraven nebo smazán, nebo se vlákno znovu otevřelo (change řekne co) | | evidence.ready | Doklady běhu nebo releasu jsou připravené ke stažení | | health.worse | Kondice plánovaného běhu nebo běhu z CI je horší než při předchozím průchodu | | health.changed | Sdílený běh má v kondici problémy, které jsou oproti běhu před ním nové nebo opravené |

Schválení kroku není komentář, takže přijde jen review.step_changed. Vrácení kroku přinese obojí: rozhodnutí i vlákno, které ho vysvětluje.

Jak doručení vypadá

Každé doručení je JSON POST s těmito hlavičkami:

| Hlavička | Hodnota | |---|---| | Kaloko-Event | Událost, např. comment.added | | Kaloko-Signature | t=<unixový čas>,v1=<hex HMAC-SHA256 z "<t>.<tělo>" podpisovým klíčem webhooku> | | User-Agent | Kaloko-Webhooks/1 |

Stejné hodnoty posíláme i jako QAwalk-Event a QAwalk-Signature pro příjemce napsané před přejmenováním.

Tělo je obálka kolem data události:

```
{
"id": "evt_Xb3kQ9vT2mLp",
"type": "comment.added",
"created_at": "2026-10-09T10:14:03.512Z",
"org": "acme",
```

```
"data": {
"run": {
"id": "2026-10-09-checkout-a1b2c3",
"scenario": "checkout",
"project": "shop",
"canvas": "https://kaloko.app/acme/runs/2026-10-09-checkout-a1b2c3/",
"api": "https://kaloko.app/api/v1/orgs/acme/runs/2026-10-09-checkout-a1b2c3"
},
"step": "payment",
"locale": "cs",
"viewport": "desktop",
"thread_id": "k3Jd9QpZr",
"comment_id": "k3Jd9QpZr",
"kind": "comment",
"author": { "email": "pavla@acme.example", "via": "session" },
"excerpt": "Po slevovém kódu je celkem 0 Kč. Zeptejte se [email] na pravidlo zaokrouhlení.",
"resolved": false,
"at": "2026-10-09T10:14:03.401Z",
"url":
"https://kaloko.app/acme/runs/2026-10-09-checkout-a1b2c3/#step=payment&comments=step&thread=
k3Jd9QpZr",
"comments_api":
"https://kaloko.app/api/v1/orgs/acme/runs/2026-10-09-checkout-a1b2c3/comments"
}
}
```

Odpovězte libovolným 2xx do 5 sekund. Kaloko posílá až poté, co odpovědělo člověku, který změnu udělal, takže pomalý příjemce nikoho nezdrží.

Ověření podpisu

Spočítejte HMAC z časového razítka, tečky a těla přesně tak, jak přišlo, porovnejte v konstantním čase a stará razítka odmítněte:

```
import { createHmac, timingSafeEqual } from 'node:crypto'

export function verify (rawBody, header, secret, toleranceSeconds = 300) {
const { t, v1 } = Object.fromEntries(header.split(',').map((p) => p.split('=')))
if (!t || !v1 || Math.abs(Date.now() / 1000 - Number(t)) > toleranceSeconds) return false
const expected = createHmac('sha256', secret).update(`${t}.${rawBody}`).digest('hex')
return expected.length === v1.length && timingSafeEqual(Buffer.from(expected),
Buffer.from(v1))
}
```

```
import hashlib, hmac, time

def verify(raw_body: bytes, header: str, secret: str, tolerance: int = 300) -> bool:
parts = dict(p.split("=", 1) for p in header.split(","))
t, v1 = parts.get("t"), parts.get("v1")
if not t or not v1 or abs(time.time() - int(t)) > tolerance:
return False
expected = hmac.new(secret.encode(), f"{t}.".encode() + raw_body,
hashlib.sha256).hexdigest()
return hmac.compare_digest(expected, v1)
```

Opakování

Když vaše adresa neodpoví 2xx, Kaloko to během pár sekund zkusí ještě dvakrát a pak zhruba den v prodlužujících se odstupech (od několika minut po 12 hodin); potom to vzdá. Každý pokus je podepsaný znovu, ale id události zůstává stejné, takže si zpracovaná id pamatujte a opakování přeskočte. Jedna změna nikdy

nepřijde jako dvě události: jeden komentář je jeden comment.added.

V Nastavení vidíte poslední odpověď každého webhooku i jeho poslední doručení (doručeno, zkusí se znovu, vzdáno) a API je vypíše pod …/deliveries. Doručení zůstávají v přehledu 30 dní.

Obsah událostí

Události nesou id, stavy, krátké úryvky a odkazy. Snímky, přílohy a celé komentáře zůstávají za API: stáhnete je vlastním tokenem z adres api a comments_api. Úryvek je jeden řádek o nejvýš 140 znacích a e-mailové adresy v něm nahrazuje [email].

Každá událost o běhu má run: id, scenario, project (nebo null), canvas (běh v Kaloku) a api. Časy jsou ve formátu ISO 8601 v UTC. via říká, jak člověk jednal: session (v aplikaci), token (CLI, CI, MCP), agent nebo integration (Jira, Linear).

run.returned

```
{
"run": { "id": "…", "scenario": "checkout", "project": "shop", "canvas": "…", "api": "…",
"environment": "staging", "verdict": "returned" },
"by": { "email": "ada@acme.example", "via": "session" },
"at": "2026-10-09T11:02:41.118Z",
"note": "Celková cena u kroku se slevovým kódem nesedí."
}
```

review.step_changed

```
{
"run": { "id": "…", "scenario": "checkout", "project": "shop", "canvas": "…", "api": "…" },
"step": "payment",
"locale": "cs",
"viewport": "desktop",
"state": "returned",
"by": { "email": "ada@acme.example", "via": "session" },
"at": "2026-10-09T10:20:12.006Z",
"comment_id": "Qm2xYt7Lc",
"note": "Celkem ukazuje 0 Kč",
"url": "https://kaloko.app/acme/runs/…/#step=payment",
"review_queue_api": "https://kaloko.app/api/v1/orgs/acme/runs/…/review-queue"
}
```

state je approved, returned nebo revoked. comment_id určuje rozhodnutí; odvolané převzaté schválení má místo něj "carried": true. locale a viewport přijdou, když rozhodnutí padlo u jednoho snímku. note mají vrácené kroky.

comment.added, comment.resolved, comment.changed

Pole	Význam
step	Id kroku, nebo null u komentáře k celému běhu
locale, viewport	Snímek, u kterého komentář vznikl, pokud nějaký je
thread_id	První komentář vlákna; u nového vlákna stejné jako comment_id
comment_id	Komentář, o kterém událost je
kind	comment, reply nebo returned (poznámka k vrácenému kroku)
change	Jen u comment.changed: edited, deleted nebo reopened
author	Kdo komentář napsal: email, via a u agentů agent
by	Kdo ho vyřešil, upravil, smazal nebo znovu otevřel
excerpt	Začátek komentáře; u smazaného chybí
resolved	Jestli je vlákno teď vyřešené
url	Vlákno na plachtě
comments_api	Všechny komentáře běhu pro váš token

url otevře krok s komentáři a vláknem na očích: https://kaloko.app/<org>/runs/<běh>/#step=<krok>&comments=step&thread=<vlákno>. Komentář k celému běhu vede na …/runs/<běh>/#comments=run.

run.shared, run.accepted, evidence.ready, health.worse

run.shared a run.accepted nesou běh s jeho kroky a adresy, kde stáhnete nejnovější snímky a změny od předchozího přijatého běhu (steps_url, changes_url). evidence.ready jmenuje běh nebo release a adresy jeho dokladů, balíčku a matice dohledatelnosti: Doklady o kvalitě pro váš systém kvality. health.worse nese běh, ten předchozí a nové nálezy: Kontrola kondice aplikace.

health.changed

Přijde po každém sdíleném běhu scénáře, jehož kontroly kondice se oproti předchozímu běhu ve stejném prostředí pohnuly: něco selhává (nebo si jím hodnotitel není jistý), co předtím ne, nebo je v pořádku něco, co předtím selhávalo. Počítá se po stránkách, jazycích a kontrolách, takže se hodí na týdenní SEO průchod tisíci stránek. První běh a běh, ve kterém se nic nepohnulo, nepošlou nic.


```
{
"run": { "id": "2026-10-12-seo-weekly-a1b2c3", "scenario": "seo-weekly", "title": "Týdenní
SEO", "environment": "production", "started_at": "2026-10-12T05:00:00.000Z", "canvas":
"https://kaloko.app/acme/runs/2026-10-12-seo-weekly-a1b2c3/#health" },
"previous": { "id": "2026-10-05-seo-weekly-d4e5f6", "started_at": "2026-10-05T05:00:00.000Z"
},
"counts": { "new": 14, "fixed": 3, "still_open": 120, "unchecked": 0, "worse": 1, "expired":
0 },
"open": 134,
"new": [
{ "id": "eshop|de|HC112|sklep-kubek", "step": "eshop", "locale": "de", "check": "HC112",
"text": "The page carries structured data (JSON-LD)", "page": "sklep-kubek", "url":
"https://shop.example/de/eshop/kubek", "severity": "warning", "first_seen": { "run":
"2026-10-12-seo-weekly-a1b2c3", "at": "2026-10-12T05:00:00.000Z" } }
],
"fixed": [],
"aggregate_url":
"https://kaloko.app/api/v1/orgs/acme/runs/2026-10-12-seo-weekly-a1b2c3/health/aggregate",
"diff_url":
"https://kaloko.app/api/v1/orgs/acme/runs/2026-10-12-seo-weekly-a1b2c3/health/diff"
}
```


new a fixed vyjmenují prvních 20, zbytek je za diff_url. Pole vysvětluje API kondice pro dashboardy.

Navrhněte flow se svým agentem

V tomto návodu navrhnete obrazovky jednoho flow jako živé HTML se svým agentem, necháte tým, ať je okomentuje, publikujete verzi a uděláte z ní návrh, se kterým se porovná každá implementace. Potřebujete projekt, ve kterém už proběhl kaloko init, a scénář pro dané flow, třeba qa/flows/checkout.yml.

Přidejte designové prostředí

Designové prostředí je složka, kterou CLI při průchodu servíruje. Přidejte ho do kaloko.config.yml:

```
environments:
design: { kind: design, serve: design }
```

Scénář zůstává stejný, jaký používá vývoj a QA: stejné kroky, stejná kritéria. Návrh se tak podle akceptačních kritérií zkontroluje dřív, než ho někdo začne stavět.

Nechte agenta nakreslit kroky

Požádejte agenta, s jakýmkoli design skillem, který váš tým používá:

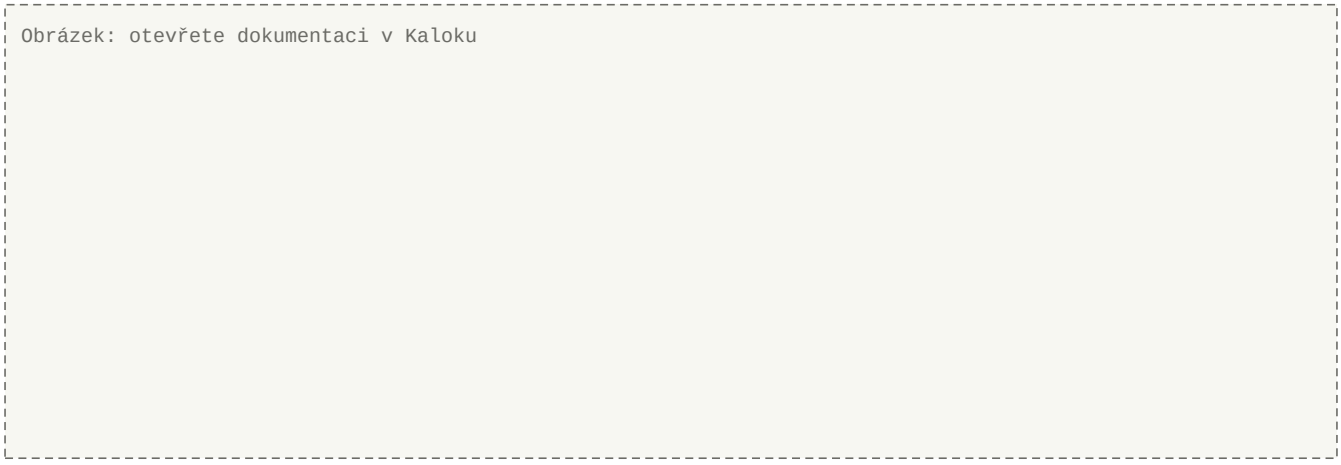
> Navrhni checkout flow do design/ podle našeho brand skillu: jedna složka na každý krok z qa/flows/checkout.yml, > sdílené CSS v design/_shared, tokeny v design/tokens.json. Pak spusť kaloko walk a kaloko sync.

Z každého kroku vznikne design/<krok>/index.html: obyčejné HTML, CSS a JS, které se otevře i z disku. Odkazy mezi kroky jsou relativní, třeba ../payment/index.html.

Projděte, zkontrolujte a synchronizujte

```
npx kaloko start --scenario qa/flows/checkout.yml --env design
npx kaloko design lint
npx kaloko walk
npx kaloko sync
```

design lint zkontroluje složku podle pravidel přenosného HTML: relativní adresy, žádné CDN, žádné úložiště v prohlížeči, předvídatelný obsah. walk vykreslí jen kroky, které se změnily, a sync je uloží do společného konceptu a převezme kroky, které změnili ostatní.



Koncept designového flow · cs · desktop

Jedna šablona v několika variantách

Některé kroky mají varianty: e-mail pro každý důvod, který může podpora vybrat, landing page pro každé město, kartu pro každý tarif. Ve scénáři z takového kroku udělejte stack krok a každé variantě dejte vlastní složku:

```
- id: credit
```

```
kind: stack
title: Kredit od podpory
items: { urls: [refund, late, damaged] }
```

```
design/credit/index.html přehled, nepovinný
design/credit/refund/index.html
design/credit/late/index.html
design/credit/damaged/index.html
```

Když items.urls složky nevyjmenuje, variantou je každá podsložka s index.html. walk vyfotí každou variantu v každém viewportu a na plachtě krok otevře seznam svých variant i s výsledky. Kliknutím variantu otevřete živě. design lint ohlásí složku varianty, které chybí index.html.

Webová písma a prototyp, který si pamatuje

Živé kroky načítají soubory jen ze složky s návrhem. Pokud vaše designy berou písma z Google Fonts, Adobe Fonts nebo z firemního CDN, admin organizace tyto adresy přidá v Nastavení -> Zabezpečení -> Písma a obrázky v designech. Zapište je i do prostředí:

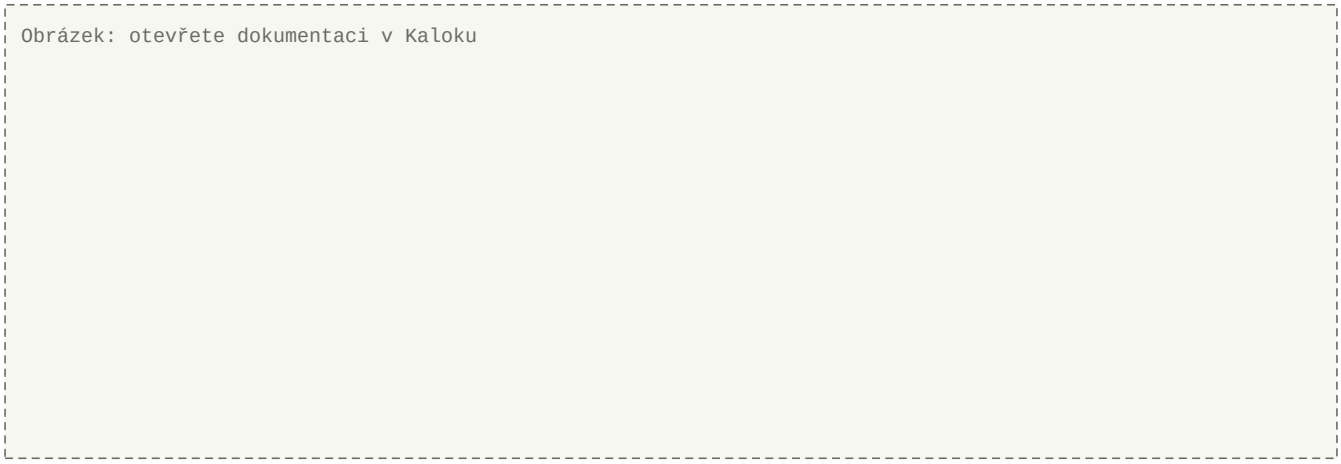
```
environments:
design: { kind: design, serve: design, origins: [https://fonts.googleapis.com,
https://fonts.gstatic.com] }
```

design lint pak řekne, které z nich organizace ještě nepovolila. Z těchto adres se načtou styly, písma a obrázky, skripty nikdy.

Košík naplněný v jednom kroku se může ukázat v dalším: požádejte agenta, ať ho drží ve stavu prototypu (window.kaloko.state). Každý krok otevřený samostatně dál ukazuje svá ukázková data a Začít znovu v prezentaci košík zase vyprázdní.

Komentujte společně

Tým otevře koncept na plachtě, proklikne si živé kroky a připíná komentáře k prvkům. Každý vidí, kdo se právě dívá. Agent si komentáře přečte přes kaloko feedback, upraví, o co žádají, a synchronizuje znovu.



Živý designový krok · cs · desktop

Prázdné košíky a témata značek

Obrazovka, kterou lidé vidí jen v ideálním stavu, se bez těch ostatních i postaví. Vypište je u kroku a agent nakreslí pro každý stav stránku vedle stránky kroku:

```
- id: cart
states: [empty, error, loading]
```

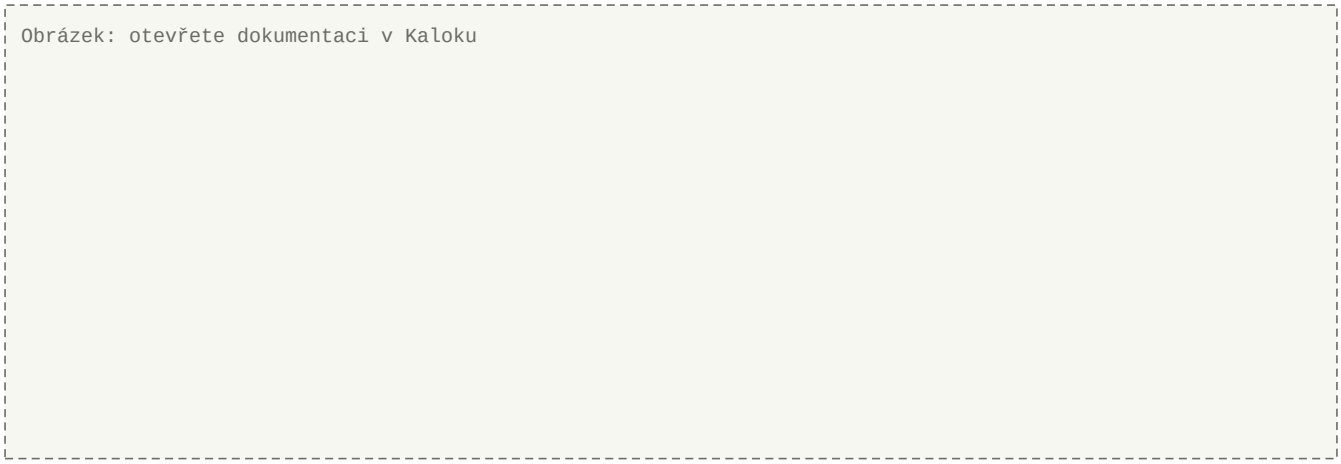
design/cart/empty.html je prázdný košík, design/cart/error.html košík, který se nenačetl. Každý stav má vlastní snímek, komentáře i verdikty a na plachtě přibude vedle jazyka přepínač Stav kroku.

Sada tokenů s více režimy, třeba brand-a a brand-b, vykreslí každý krok v každém režimu s tokeny režimu jako CSS proměnnými a na plachtě přibude přepínač Režim tokenů. tokens_mode v prostředí zůstává výchozím režimem; token_modes: [brand-a, brand-b] vybere, které režimy se vykreslí. Samotný světlý a tmavý režim se dál řídí color_schemes: [light, dark].

Publikujte verzi

```
npx kaloko share --name "Checkout v2"
```

Tím publikujete verzi celého flow, třeba "v2 · 3 změněné, 17 beze změny“. Nezměněné kroky se ukládají jen jednou. share --notes-preview vám před publikováním ukáže poznámky o tom, co se změnilo.



Publikovaná verze · cs · desktop

Udělejte z ní referenci

Až verzi někdo přijme, nastavte ji jako referenci designu (od plánu Team):

```
npx kaloko reference --design --version 2
```

U kroků implementace ve scénáři zapněte packs: [design]. Od plánu Business se pak každý běh implementace porovná s referencí: pixely, struktura, tokeny a kontrast, a kaloko drift napíše, který token použít tam, kde se kód liší.

Když máte návrh ve Figmě

Pokud návrh už žije ve Figmě, kaloko design import figma <adresa-souboru> --scenario qa/flows/checkout.yml udělá z framů designové kroky a z proměnných tokeny. Složka framů vyexportovaných z Figmy funguje i bez tokenu.

První běh za deset minut

Na konci tohoto návodu budete mít jedno flow své aplikace na plachtě v kaloko.app, nasdílené, se všemi obrazovkami ve všech jazycích a velikostech obrazovky, které scénář uvádí. Potřebujete Node 20 nebo novější, Chrome a aplikaci spuštěnou lokálně.

Nainstalujte CLI

V kořeni projektu:

```
npm install --save-dev kaloko
npx kaloko init --agent claude --org <vaše-organizace>
```

Pro Codex použijte --agent codex, pro oba --agent all, a když budete CLI ovládat sami, volbu vynechte. init zapíše kaloko.config.yml a zkopíruje skill k vašemu agentovi. Zároveň si projekt přečte: framework, příkaz a adresu dev serveru, jazyky a hlavní cesty. Podle nich připraví první scénář qa/flows/main-pages.yml, takže první průchod funguje bez úprav.

Aby měl projekt v Kaloku hned správný název, přidejte --project eco-podcasty --project-name "ECO Podcasty". Zkratka se objeví v adresách, název uvidí lidé v aplikaci.

Když je v repozitáři víc webů nebo aplikací (monorepo, aplikace v Rails s několika weby ve Foliu), init je vypíše a scénář nepřipraví, protože cesty z celého repozitáře by weby smíchaly. Jeden vyberte přes --site a pokud běží, zadejte jeho adresu přes --base-url. Stránky se pak vezmou z jeho sitemapy nebo navigace:

```
npx kaloko init --site eco-podcasty --base-url http://localhost:3000
```

Pokud organizaci ještě nemáte, nejdřív ji založte; stanete se jejím adminem. Přes Google, Microsoft nebo GitHub to jde naráz: přihlásíte se, pojmenujete organizaci a můžete začít, bez čekání na odkaz v e?mailu.

Přihlaste tento počítač

```
npx kaloko login
npx kaloko doctor
```

login otevře prohlížeč, kde kód schválíte a vyberete, co smí token dělat. Uloží se do ~/.config/kaloko/.env. doctor zkontroluje Node, Chrome, konfiguraci, prostředí a token a u každého x napíše, co opravit.

Po přihlášení npx kaloko projects sync založí projekt s názvem z konfigurace. Jinak ho založí váš první kaloko share.

Projděte flow

Spusťte dev server a pak:

```
npx kaloko start --scenario qa/flows/main-pages.yml --env local
npx kaloko walk
npx kaloko preview
```

start založí běh, walk otevře v prohlížeči každý krok, zachytí ho a průběžně spouští kontroly. Na konci vypíše verdikty: co neprošlo a na co se má podívat člověk. preview vypíše cestu k místní plachtě, která se otevře přímo z disku.

S nainstalovaným skillem nemusíte nic psát a stačí požádat agenta: "Projdi s Kalokem hlavní stránky na localu a ukaž mi, co neprochází.“ Spustí stejné příkazy.

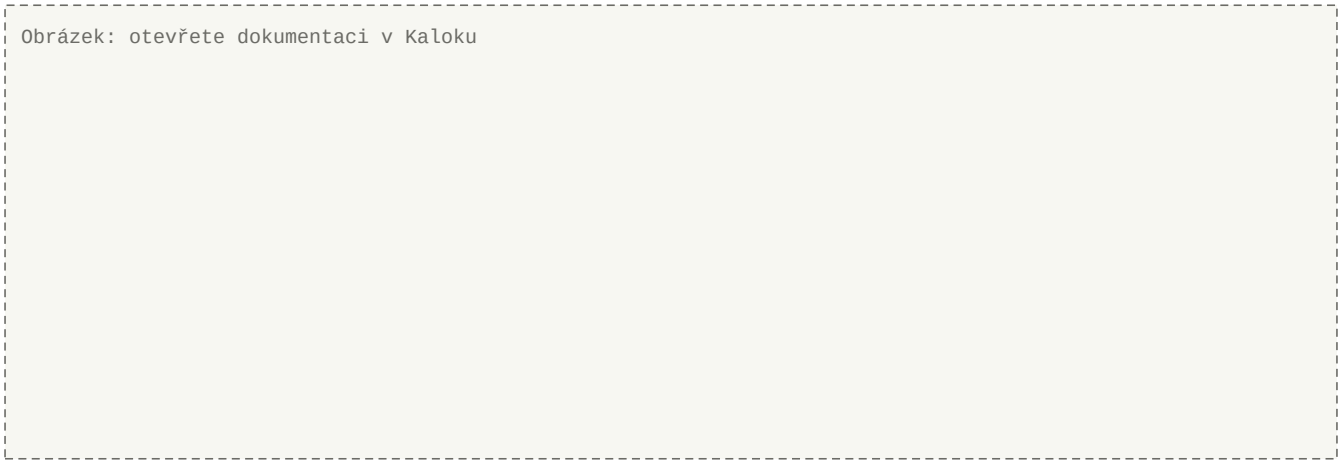
Nasdílejte ho

```
npx kaloko share
```

Běh se nahraje do kaloko.app a příkaz vypíše odkaz. Na větvi s pull requestem přidá share --pr odkaz i do komentáře v pull requestu.

Podívejte se a rozhodněte

Otevřete odkaz. Běh se ukáže jako mapa kroků; kliknutím na krok uvidíte snímek, kritéria a co se zjistilo.



Mapa běhu · cs · desktop

Kroky, které sedí, schvalte, ostatní vraťte s poznámkou, a až budete spokojeni, běh přijměte. Váš agent si poznámky přečte přes kaloko feedback.

Kam dál

- Napište scénář pro flow, na kterém vám záleží nejvíc.
- Projít běh ukáže plachtu podrobněji.
- Spouštějte ho v CI u každého pull requestu.
- Přehled CLI uvádí všechny příkazy.

Co je nového: Kaloko

0.22.0

Vydáno 2026-10-20

Added

• A Health checks page in the help lists every pack and each of its checks: what it looks at, why it matters, its severity and where it applies, in English and Czech. It is written from the same code the CLI runs, and the Health tab on the canvas links to it.

• The scenario page in the app shows which health packs the latest run had on and off, and how to switch one with packs: in the scenario file.

• Light or dark, your pick: the account menu in the app and on the canvas has a theme switch (System, Light, Dark). The choice stays on this device and applies before the page paints, so nothing flashes white at night.

• The account menu holds who you are, the language of Kaloko, the theme, a link to the Kaloko website and Sign out. On a phone the same items sit at the bottom of the menu.

• Signed-in people reach the website again: "Kaloko website" in the account menu and the footer leads to the home page instead of back to your organization.

• A product you no longer ship can be archived: kaloko products remove <product>, or Archive product at the end of its changelog settings. It disappears from Products, its changelog and docs answer "not found", and its releases and docs stay stored. kaloko products sync reports it as archived instead of creating it again, and kaloko products restore <product> brings it back.

• The help page on browsers and devices describes the default capture setup and how to change it.

• kaloko scenario move <scenario-id> --project <slug> moves a scenario to another project with all its runs, design drafts and screens. Admins do the same in the app, in Scenario settings on the scenario's page, after a confirmation. The move wins over project: in the config, so later shares land in the new project too.

Changed

• On a design environment the code-only packs (seo, aeo, perf, console, errors) are off, because a designer's HTML has no real server, meta tags or error statuses. health: { design_packs: [console] } in the scenario keeps one on. The run says which packs it turned off (packs_off in run.json).

• Design versions are listed under Design drafts on the home page and no longer among the runs.

• The canvas explains Live, Inspect and Dev mode in a line each: Live runs your own uploaded HTML, Inspect styles shows an element's styles and tokens, and Dev mode is the handoff to development. Kaloko does not write code from a design.

• The canvas shows the same main links as the app, in the same order: Scenarios, Runs, Screens, Projects, Products.

• The language of Kaloko moved from the canvas bar and the app header into the account menu, so it no longer sits next to the run's own locale switch.

• The menu button on a phone is round with three lines in the app too, like on the canvas.

• Someone who is a guest or a viewer in another organization can now create their own. Only a full member elsewhere (Designer, Developer / Tester or Admin) is asked to sign in instead. Sign-in goes to the organization where you are a full member first; the others are in the organization switcher.

• New scenarios from kaloko init, kaloko draft and the template capture mobile at 360 × 800 instead of 390 × 844, next to desktop at 1280 × 800. Existing scenarios keep their sizes and baselines. A tablet is added only when you ask for one (tablet: [744, 1133]).

• The scenario and runs lists no longer look the same. A scenario row starts with its trend, the number of runs and the last result; the runs list is split by day, and each run starts with its time and who shared it.

• Branch and commit moved from their own column to a grey line under the scenario name, with a long branch cut short (hover shows it whole). Both lists now fit a 1280 px screen without a sideways scrollbar.

• The scenario list shows the project as a plain label. The project picker in every row is gone, so a stray click no longer moves a scenario with all its runs.

• Return and Approve on a step have their own bar under the comment box, so a comment no longer sits one click away from a decision. In review mode you decide in the bottom bar only; the side panel is for comments.

• In Live mode the comment box says that comments go on an element of the page and starts picking one; pinning and drawing stay on the screenshot.

• "Mark a changing region" no longer disappears on drafts, for viewers and reviewers, or on a step without a screenshot. The button stays, disabled, and says why.

Fixed

• The Screens page follows dark mode: tiles, picture frames and the status labels used light colours on a dark page.

• On the Screens page, the "not checked" label no longer hides off screen.

• The canvas viewport switch offers only the viewports the run has: no "Both" for a run captured on one, and Tablet when the run has a tablet (with three or more viewports the first choice is "All").

• The gate of an environment (HTTP Basic, service-token headers) no longer follows a redirect to another host. Podcast audio that a gated page sends on to S3 now plays instead of failing with HTTP 400, and an identity provider or a link checked on another host never sees the gate. Redirects within the gate keep it, as before.

0.21.3

Vydáno 2026-10-15

Fixed

• In a step's detail, opening an AI-evaluated or failed criterion no longer squeezes its explanation into a narrow column with the technical detail spilling over it; everything sits under the criterion's title again.

0.21.2

Vydáno 2026-10-13

Changed

• Stack pages now report console errors. Before, the check passed on every page, so stack steps with no_console_errors or the console pack may fail after this update where they passed.

• kaloko doctor --json now lists the token's scopes in the token check, so a CI script can check that the token may make a run public (full), not only share it (upload), before it spends twenty minutes walking.

Fixed

• Console errors on the shared browser stay with the capture they belong to. A capture used to list every error the browser had seen on the same page, so one failed request from yesterday's test showed up in every run after it. Now kaloko capture counts only errors from the tab it captures, logged after the run started and after the last capture on that tab, and a walk counts what a page logged since the previous step on it was captured.

• The shared browser's console log starts empty with every kaloko browser start and keeps only its last few megabytes, so a browser left running for a week no longer piles up a log nobody reads.

0.21.1

Vydáno 2026-10-11

Added

• Findings can open issues in GitHub Issues and Plane, not only in Jira and Linear. Connect them in Settings -> Integrations (the Kaloko GitHub app with the Issues permission, or a fine-grained token; a Plane API key), then

name the repository or the Plane project in PUT …/scenarios/<id>/findings/sync. One issue per finding, a comment when the number of pages changes a lot, closed when it is fixed and reopened when it comes back. Admins list which repositories and Plane projects each Kaloko project may send findings to.

• Routes per check or pack in the findings sync: send canonical, hreflang and structured data to the developers' repository and thin content or low click-through to the marketers' Plane project, from one scenario.

• With traffic: on the stack, findings carry the clicks, impressions and impact of their pages, and so do the issues the sync opens.

Fixed

• A tracker that timed out while creating an issue for a finding could get a second one on the next run. The finding now waits and says the issue may exist; after a look at the tracker, Try again on the findings page lets Kaloko create it with the next run.

• Evaluating the same run again no longer sends a step's recording to triage a second time. On a busy machine the first sampled frame sometimes came out empty, so the second evaluation picked other frames and missed the cache. Kaloko now waits until a frame is really drawn and remembers the frames it chose for each recording, so they are the same every time.

0.21.0

Vydáno 2026-10-11

Changed

• Stack pages are now captured whole by default, like every other step; until now a stack kept the first screen only. This is a changed default, and stack runs grow several times: a typical long page stores a screenshot of about 600 KB instead of about 70 KB, and a thumbnail of about 180 KB instead of about 25 KB, because the thumbnail is as tall as the page. Vision checks now see the whole page too, not just what fits on the first screen. To keep the old size, set capture: { full_page: false } on the stack; capture: { screenshot: thumb, full_page: false } is the smallest and suits big monitors that need the page facts rather than the picture.

• When a stack captured whole takes most of a run close to the per-run limit, kaloko share names it and suggests full_page: false; a run over the limit lists full_page: false among the ways to trim it.

• Captures finish finite animations, and most captures show endless ones (a spinner, a pulsing badge) at their first frame. On pages with animations the first run after the update can differ from the runs before it, even though the page did not change.

Fixed

• Before a capture Kaloko waits until what scrolling changed on fixed and sticky parts is back as it was at the top (a header that floats while scrolling, a search panel fading in), or has stopped changing for a second, at most 3 s. Full-page captures no longer show a sticky header caught in its floating state with a half-visible panel.

• A full-page capture no longer catches CSS transitions halfway: resizing the page for the capture restarted them on some sites (a search panel fading out again over the breadcrumbs). Transitions end at once and finite animations finish for the capture.

• A page of a stack has its own Health tab, as a step has: its health checks with the evidence for that page, findings first, and Adopt for the stack step. Before, health checks of stack pages were only in the run's Health panel and the page listed them among its acceptance criteria.

0.20.1

Vydáno 2026-10-11

Added

• Hosted runs group the language versions of a stack page by hreflang (items.variants: hreflang), as kaloko walk does: the same pages in the same week in every language, one row per page in the stack index, and a missing translation marked on the page and counted in HC1005. Reading a source page for its hreflang counts as one

hosted page. kaloko validate no longer warns about such stacks in hosted runs.

Fixed

• kaloko walk reads the sitemaps and robots.txt of a stack with the environment's access (its headers, HTTP Basic, client certificates and the walk's cookies), so a staging behind a password no longer ends with an empty stack and HTTP 401. A sitemap on another host, or one that redirects there, is still read without them. Each file has 60 seconds and 60 MB.

0.20.0

Vydáno 2026-10-10

Added

• Findings: what a scenario has open in health is grouped by check, site, language and step, so two hundred product pages without structured data in Polish are one row, not two hundred. Each finding has its severity, how many pages it hits, up to ten of them with the CMS edit link, and when it first and last appeared. Health -> Findings shows the table per scenario; GET …/scenarios/<id>/findings and MCP list_findings give the same to dashboards and agents (a read:health token reads them too).

• Acknowledge a finding when someone is on it, or ignore it with a reason. The decision stays with the finding from week to week; a finding is fixed once none of its pages fails. Anyone who may comment on the scenario may decide, in the app or with PATCH …/scenarios/<id>/findings/<id>.

• A scenario can open a Jira or Linear issue for each new finding at or above a severity, comment when the number of pages changes a lot, close it when the finding is fixed and open it again when it comes back (PUT …/scenarios/<id>/findings/sync, Developer / Tester). A run opens at most 20 issues; the rest follow with the next run.

• Hosted runs read stack steps: pages listed by items.urls, a sitemap (gzipped and sitemap indexes too) or the sitemaps in robots.txt, with match, exclude, limit and rotate: week picking the same pages as kaloko walk. Each page is read without a screenshot: checks of selectors, texts, titles and meta tags are decided, index fields such as title, H1, description and JSON-LD types are filled in, and the canvas shows the stack with its index. Big stacks are read over several quarter-hour rounds, at most 1,000 pages per run; Run now, deploy triggers and edits wait for a run under way.

• kaloko schedule add takes an environment with a base_url per language; the hosted run opens each language at its own address, otherwise at default, as the walk does.

• Settings -> Integrations and kaloko schedule list show how far a stack run got and how many health results the last run found new and fixed.

• items: { variants: hreflang, source_locale: cs } on a stack step shows the translations of a page as its language versions instead of separate pages. The list, limit and rotate come from the source language, and every other language walks each page's own hreflang alternate, so the stack index has one row per page with a screenshot per language and all languages check the same pages in a given week.

• A page without a translation is reported instead of left out: the stack index marks the missing language, HC1005 counts the pages (no hreflang for the language, only x-default, a hreflang back to the page itself, a translation that answers an error, or several pages sharing one translation), and kaloko export findings lists one row per page and language.

• Traffic data for the fix list: traffic: in kaloko.config.yml or the scenario joins Search Console and GA4 numbers onto the walked pages, from a CSV export (English or Czech headers) or, with a service account key, straight from the Search Console and GA4 APIs, once per run. kaloko export findings then adds clicks, impressions, CTR, position, sessions and an impact score and puts the most seen pages first, the page index on the canvas gets Impressions and Clicks columns, and the seo pack adds HC127 for pages that are seen often in search but clicked less than half as often as the site's pages at similar positions. Nothing is requested unless you configure it.

Changed

• Every page of a stack in a hosted run counts against the monthly budget of hosted pages. When the budget runs out halfway, the run waits up to a day for more, then shares the pages read so far and names the rest as not walked.

• A hosted run with stack steps takes up to 30 languages (10 without).

Security

• Sitemaps and robots.txt read by hosted runs go through the same public-address check as pages, on every redirect and for sitemaps on other hosts too, without the environment's credentials; pages are opened only on the environment's own addresses. match and exclude are checked when a schedule is saved and run in a linear-time matcher.

0.19.0

Vydáno 2026-10-10

Added

• Check nojs_parity: one more request per page fetches the raw HTML without JavaScript and compares it with the rendered page. It fails when the title or canonical differ, when the meta description, H1 or a JSON-LD type appear only after scripts run, or when the raw body text is below 30 % of the rendered one (ratio:). With user_agent: gptbot (or oai-searchbot, claudebot, perplexitybot, googlebot, bingbot, a whole user agent string) it comes like that crawler, without cookies, and a page robots.txt closes to it is reported as blocked, not fetched. A bot the site, a WAF or a gate turns away (403, 429, a challenge page, a redirect to a sign-in) is not evaluated, with a note saying who was turned away. The aeo pack adds it as HC905 for GPTBot. Hosted runs show it as not evaluated; kaloko recheck keeps the walk's result.

• An evaluator budget for large sites: evaluate.budget.calls in kaloko.config.yml, or kaloko evaluate --budget N and kaloko walk --budget N, caps the paid JEV calls of one evaluation. Ordinary steps are asked first, then the pages of each stack, starting with pages that fail a deterministic check and pages checked longest ago. What does not fit stays "not evaluated" with a note, and kaloko evaluate says how many captures it left out. Cached answers cost nothing, so the next evaluation picks up the rest. The same captures always spend the budget on the same pages.

• items.semantic_order: priority on a stack step picks its semantic sample the same way without a budget; by default the sample stays the first pages in walk order.

• Your own text evaluator behind an OpenAI-compatible API, such as Ollama (http://localhost:11434/v1) or LM Studio: an evaluator of kind: openai_compatible with base_url and model, chosen with evaluators.text. It answers the same questions from the same page text. With evaluators.escalate_to: jev, answers it is not sure about go on to JEV, counted in the budget. Its verdicts are marked as your own model's on the canvas, in acceptance records and in the pull request comment. Nothing changes until you configure it, and the page text goes only to your base_url.

• A dashboard can read the health of a run already counted: GET …/runs/<id>/health/aggregate?by=locale,step,check gives pass, fail, unsure and not evaluated per language, step, check or pack, with the pages hit most. Group ids stay the same from week to week. GET …/scenarios/<id>/latest finds the newest shared run of a scenario (?environment=), and …/latest/health/aggregate answers for it in one call. MCP has get_health_aggregate.

• Health over time: GET …/runs/<id>/health/diff (MCP get_health_diff) lists, page by page, what fails or is unsure now and did not in the previous run of the scenario on the same environment, what was fixed and what is still open, each with the run where it first appeared. A page the run did not check (a rotating sample, a language left out) stays open as unchecked instead of counting as fixed. The health page shows new and fixed per scenario, and the scenario's health history has them per run.

• The health.changed webhook tells you after a shared run when something in its health is new or fixed, with the counts and the first results.

• Tokens with the read:health scope read health and the newest run of each scenario and nothing else, for a dashboard that pulls once a week. Create one in Settings -> API tokens (on Enterprise also as a service

account). The fields are described in the new help page "Health API for dashboards".

• kaloko init --seo <site> drafts a weekly SEO and answer-engine monitor from the live site: robots.txt and the sitemaps give the languages, a sample of each language's pages the biggest content families (first path segment), paired across languages by hreflang, the same segment or size. It writes qa/flows/seo-<site>.yml with one rotating stack step per family (match per language, rotate: week, packs seo and aeo; a11y and perf with --packs), the production environment in a new config, and prints the next steps. A site that cannot be reached writes nothing; a site whose pages live on its www (or https) name is followed there.

• Stack pages can link back to their source: an item script or a JSON list gives edit_url and content_ref, or items.resolve: steps/resolve.mjs looks them up for the walked pages (15 s per page). The page detail shows "Edit in CMS", and the stack CSV and MCP get_stack_items carry both.

• kaloko export findings [--format csv|json] [--locale l] [--step s] [--out file]: the fix list of a local run, one row per failing criterion or health check, page and language, with the evidence, edit_url, content_ref and the command that walks that page again. It reads the run and writes nothing into it.

• Pack aeo (answer engines: AI search and assistants), advisory like the other packs: HC901 questions and answers on the page (details/summary, dl/dt, two or more headings ending with "?") come with FAQPage or QAPage structured data, HC902 a short answer-first paragraph follows the H1, HC903 /llms.txt answers at the site root (one request per site and walk), HC904 a meta description is present. Pages without questions show HC901 as not evaluated. Hosted runs check HC904 and mark the others "not evaluated" with one note.

• Stack steps read gzipped sitemaps (sitemap: /sitemaps/{locale}/sitemap.xml.gz, or gzip bytes behind any URL) and follow sitemap indexes two levels deep.

• items: { sitemap: robots } (or robots: true) takes the pages from the sitemaps listed in /robots.txt. When their URLs name the walked locale (/sitemaps/de/…), only that locale's sitemaps are read. match, exclude and limit work as before.

• Stack index fields jsonld_types (the structured data types of each page, for example Product, Offer, BreadcrumbList) and meta tags by name, such as from: og:title, from: twitter:card or from: meta:robots, so items.json carries them for every URL.

• items: { limit: 100, rotate: week } walks a big catalogue in turns: each ISO week (or each run with rotate: run) takes the next limit pages of the list, so every page is checked within a few weeks. The stack index shows per language which part was walked and how many pages were never checked, and each page its last_checked.

• match and exclude of a stack step take a map by language ({ default: '^/eshop/', fi: '^/fi/tuotteet/' }) for sites that translate their URL segments.

• An item script gets the step (node: id, title, lane, session) and items.params, so one script serves several stacks.

• expect_status: 404 on a step says it is an error page. kaloko draft sets it on the drafted 404 step.

• The seo pack checks structured data: HC112 JSON-LD is on the page, HC113 every JSON-LD block is valid JSON (a warning names the block and the parse error). New assert jsonld_present takes types: [Product, BreadcrumbList] for pages that must carry certain types; jsonld_valid is the parse check.

• Every capture records structured_data (blocks, invalid blocks, the schema.org types and the recognized ones: Product, Offer, BreadcrumbList, FAQPage, Article, BlogPosting, Organization, WebSite and a few more). A stack step's items.json and its manifest count pages per type, pages without any structured data and pages with invalid JSON-LD, so a dashboard can chart them week by week.

• A stack step with the seo pack now compares its pages with each other after the walk: shared titles, meta descriptions and H1s within one language (HC121–HC123), canonicals that point elsewhere or to a page that answers an error or says noindex (HC124), hreflang the other walked language version does not answer (HC125), and sitemap pages that answer other than 200, redirect or say noindex (HC126). Advisory like the pack; the findings count pages, and items.json lists the affected addresses in groups.

• Pack locale for multilingual sites: html lang, the page's own hreflang, the language of the text and the decimal separator of prices are checked against the walked locale (HC1001–HC1004). An untranslated page in a language version shows up as HC1003.

• Site checks with health: { site: true } in the scenario, once per site and run: robots.txt parses and does not block the walked start pages (an error on production, a warning elsewhere) or AI crawlers, every sitemap it lists answers and is valid with absolute same-host addresses and sane lastmod dates, /llms.txt is there, and http:// and the other host name (with or without www) redirect in one hop (HC1101–HC1107). kaloko health and the canvas show a row per site; health.site.environments: [production] keeps staging walks from checking.

• capture: { screenshot: thumb } keeps only the 640 px thumbnail of each page as the step's picture: for stacks of hundreds of pages whose checks read the page, not the picture (SEO crawls).

Changed

• kaloko walk --items picks stack pages from every matching page, not only from this week's rotation window, so a fixed page can be walked again in any week; such a walk leaves the rotation state as it was.

• match and exclude apply before limit for every source of a stack step, scripts included.

• The errors pack checks HC601 (soft 404) and HC605 (the explanation) only on error pages: a step with expect_status of 400 or more, or a page that answered 4xx/5xx. On normal pages they are not applicable instead of failing. kaloko validate warns when the pack is on a stack step.

• kaloko validate and kaloko start warn when the config asks every scenario for its own project (a comment, or a product with projects:) and the scenario has no project:.

Fixed

• When kaloko share gets HTTP 404, it says why: which project the run names and which projects the organization has, or that the token belongs to another organization or expired (kaloko login --org <org>). Other commands say the same for a bare "Not found" on an organization or project.

• Changing the Kaloko app's repository access or permissions on GitHub now brings you back to Settings with the repositories read again, instead of an error saying the installation link expired.

• kaloko share checks the run against its limit by the size the service stores (HTML and JSON gzip-compressed), not the size on disk, and when a run is too big it names the largest steps. A 20-language SEO run of 1 400 pages that was refused at 529 MB is about a third of that as stored.

• The personal data check before sharing no longer reads the user part of a URL as an e-mail address (a public Sentry DSN such as https://key@o1.ingest.sentry.io/23 in a meta tag).

0.18.0

Vydáno 2026-10-10

Added

• Every pull request in a repository connected to the Kaloko GitHub App gets one Kaloko check on its newest commit. It waits for the runs shared for that commit, from a coding agent in the cloud, your machine or CI, and turns green when they pass or when the change reaches no scenario. It keeps a stable name, so branch protection can require it before the merge. An optional Kaloko review check follows the decision on those runs.

• Modes per organization and repository in Settings -> Integrations -> GitHub: off, manual, automatic (the default) and strict. The github: section of kaloko.config.yml on the base branch can make the check stricter, never looser.

• One pull request can be controlled with the labels kaloko:skip, kaloko:all, kaloko:review and kaloko:run, with the comments /kaloko skip <reason>, /kaloko walk and /kaloko all, or with the buttons on the check. Only people with the access you choose may skip, never apps or bots, and the check names who skipped it and why.

• kaloko ci plan tells the check which scenarios a commit reaches, following imports like kaloko affected. The workflow from kaloko ci github sends it before walking.

• Settings list the installation's repositories with their mode, last check and what went wrong, in words you can act on, such as a repository that is not in the installation yet.

Changed

• A shared run carries its repository, the project's folder in a monorepo, and whether tracked files differed from

the commit. The pull request is found by commit, so a sandbox without gh still gets its check. A run of uncommitted changes does not count for the commit.

Fixed

• A link to Kaloko from another site (a portal, GitLab, Teams, an e-mail) opens the page for someone who is signed in, instead of asking them to sign in again.

• A link to a step (#step=…, #lane=…, #session=…) still opens that step after signing in.

• Links to Kaloko's public changelog and help now show a preview picture in chat apps and social posts.

• In replay, -> and the Next button follow the diagram: the next screen on the same row, forward before back, instead of whichever connection the run happened to list first.

• Open index on a stack in replay shows the stack's pages; before, they opened hidden behind the replay.

• A step reference shown as an example inside a code block or inline code is no longer resolved as a picture, so kaloko docs check and kaloko docs publish stop warning about examples.

0.17.0

Vydáno 2026-10-09

Added

• Webhooks now tell you about comments and review changes: comment.added, comment.resolved, comment.changed, review.step_changed and run.returned. A QMS portal can show the open comments of a run next to its test cases and link straight to the thread, without polling.

• Each webhook takes the projects you tick, like a Slack or Teams channel. A restricted project's events reach only the webhooks that name it.

• Settings -> Integrations -> Webhooks lists each webhook's recent deliveries. A delivery that fails is tried again for about a day with the same event id. Events, payloads and signature checks are in the help under Webhooks.

• A documentation portal can show a step's screenshot next to its test step: GET /api/v1/orgs/<org>/runs/<run>/steps/<step>/screen with a read token answers the full screenshot or its thumbnail with its SHA-256. Pass the hash you stored as sha256 and a different file answers 409 instead of a newer picture. MCP agents get the same through get_step_screen.

• A run can name the records it serves outside Kaloko, such as a change request in your QMS: kaloko start --ref CC-2026-001 (repeat it for more, or set KALOKO_REFS in CI), and why it was walked with --purpose dry_run, formal or smoke (KALOKO_PURPOSE). The canvas shows both in the run's header.

• GET /api/v1/orgs/<org>/runs?ref=CC-2026-001 lists every run of a record with its verdict, environment and purpose; ?purpose=formal keeps the formal ones. kaloko runs --ref, the MCP tool list_runs and the evidence of a release take the same filters, and the evidence of a run carries run.refs and run.purpose.

• The trace matrix says who performs each step and in which part of the process: every row of trace.csv, trace.json and the evidence names the step's session (the role) and its lane (the sub-process). The evidence and MCP get_run_summary also list the flow's lanes and sessions, so a portal can show them without reading the YAML.

• A canvas link ending in #lane=1 opens the map on that lane with its band highlighted, and #session=qc brings one person's steps forward across all lanes.

Changed

• An invitation code now starts its trial the moment you sign up with it or enter it under Plan & billing. There is no checkout and no card. Plan & billing shows when the trial ends and has an Add a card button. Billing starts only when the trial ends; without a card the organization goes back to Free that day and keeps its runs.

• Organizations that already had the sample run get the current one (with health, requirement references and detail crops) in the same language, once; an organization that deleted its sample does not get it back.

• The sample run in a new organization shows what Kaloko does today: Health finds three small things in the tea shop (a picture without alt text, a missing meta description, a pale note under the delivery options), steps and

criteria carry requirement references, the cart and the payment have crops for user guides, and the keyboard walk leaves focus shots. The guided first review has one more hint, about Health, and points at the requirement and the crop on the way.

Fixed

• Organizations that entered an invitation code but never finished the checkout that followed were left on Free. Their trial now starts for the full length of the code.

• Dragging the map to pan no longer leaves lane headings, step titles and thumbnails highlighted as selected text. Text in the side panels, comments and fields still selects as before.

• On the page that approves kaloko login, picking another organization now reloads its projects, scopes and token lengths, so a project-limited token only ever lists projects of the organization it is for.

• Screenshots of runs shared with the deduplicated upload no longer go missing after an internal storage recount. The recount now leaves every file that points at stored content alone, and unreferenced content is kept for a week before it is cleaned up, so a lost reference can be restored.

0.16.1

Vydáno 2026-10-07

Added

• When you type a command, sub-command or option your kaloko does not know but a newer release has, the CLI says which version has it and how to update (npm i -D kaloko@latest, or npx kaloko@latest …) instead of guessing a typo. It asks kaloko.app/cli/latest.json at most once a day, waits at most 1.5 s and stays quiet offline; KALOKO_NO_UPDATE_CHECK=1 turns it off. kaloko doctor shows when a newer kaloko is available.

• kaloko help --json lists every command with its sub-commands and options, for agents and scripts.

0.16.0

Vydáno 2026-10-07

Added

• Signing in with Google, Microsoft or GitHub when your e-mail has no organization yet now leads to creating one: name it and you are in as its admin, without waiting for an e-mail link. The sign-up page offers the same buttons, and a trial code in its link carries over.

• Trial code links can point to the homepage: https://kaloko.app/?kod=CODE. The visitor first sees what Kaloko is, a slim banner on every page says what the code gives ("You have an invitation: 60 days of Business free, no card."), and the code stays with them until they create an organization, by e-mail or with Google, Microsoft or GitHub. An organization without a subscription starts the offer from Plan & billing with one click.

• Projects from the CLI: kaloko projects create <slug> --name "ECO Podcasty", kaloko projects rename <slug> --name … and kaloko projects sync, all with --json. The config takes project_name: (the name people see) and product: (a product the project is part of); kaloko init --project … --project-name … --product … writes them and, when you are signed in, creates the project on Kaloko. The MCP tool upsert_project does the same for agents.

• kaloko init in a repository with several sites or apps (a monorepo, Rails engines, several Folio sites) lists them instead of guessing routes. --site picks one, --base-url reads its pages from the running site (sitemap, then the links of its home page) and --skeleton writes a scenario with a TODO to fill in.

• Details for guides: name the parts of a screen a guide talks about under details: on a step (an id, a name and a selector). Every walk takes each one in every language and screen size with its section or card around it and a ring around the element, with personal data hidden as on the screenshot. In docs and the changelog, shows it; kaloko docs check warns about a detail the scenario does not declare or no accepted run has yet, exports carry the picture and the MCP tool resolve_step_image takes the detail id.

Changed

• New organizations cannot take an address Kaloko uses or keeps for itself: its routes (hooks, scim, …), words

for likely future pages in English and Czech, two-letter language codes and look-alikes of Kaloko, QAwalk and Sinfin. Existing organizations keep their addresses.

• The first kaloko share of a new project creates it with project_name: from the config, not with its slug. An existing project is never renamed by a share: when the names differ, the share says so and points to kaloko projects sync.

• kaloko init leaves out what is not a page: favicons and other static files, ad and tracking endpoints, API, admin and sign-in routes, routes that need a parameter and, with a running site, anything that does not answer HTML. The drafted scenario says at the top where its pages came from and what was skipped.

• The admin of a project can rename it with their API token (kaloko projects rename); everything else about a project (visibility, people, removal) stays in the app.

• Teammates need this CLI version or newer once kaloko.config.yml has project_name: or product:: older ones reject keys they do not know.

• The focus shots of the keyboard walk show on the Health tab under the focus check and beside the tab order of the shown language and screen. In Step attachments, which are now grouped into details, keyboard focus and crops, each focus shot says "Keyboard focus · HC309" and opens that check.

• The scenario browser leads with the scenario's name (up to two lines) and shows its id as a detail; the runs header names the scenario; the kind reads Task/Flow (Úkol/Průchod).

• A run opens in the reader's language when it has screens in it (an English reader sees the English screens first), and that language leads the locale list; an embedded canvas always follows its page.

Fixed

• Design drafts on the scenarios page follow its filters: a project shows only its own drafts, run filters leave them out, and each draft shows its name and project.

0.15.0

Vydáno 2026-10-07

Added

• Requirement references: refs: on a scenario, a step or a criterion (refs: [URS-12, FS-3.4], free text). They reach the run, the acceptance record and the trace matrix. Runs also record the SHA-256 of the scenario file as walked and the CLI, Node.js, operating system and browser versions.

• Evidence for your own documentation (Business): for any shared run, accepted or not, and for a release. It says whether the run was accepted and whether it was signed in Kaloko, and lists the criteria as walked with who assessed each result (a person, a deterministic check or a named AI model, AI always advisory), screenshot hashes, decisions, signatures, the audit trail and the trace matrix. kaloko evidence <run> | --release <version> (--json, --csv), the MCP tool get_evidence, the API (…/runs/<id>/evidence, …/trace.csv, …/audit.csv, …/evidence?product=&version=) and the format qawalk.evidence.v1 with a JSON schema.

• The evidence package: kaloko signoff <run> --package or Evidence package (ZIP) on the canvas downloads the signed record (JSON and PDF), the evidence, the trace matrix, the audit trail and the screenshots with a signed list of every file. kaloko signoff --verify checks a package as well as records.

• Electronic signatures, optional and off by default (Business): an admin turns them on in Settings -> Security for every project or some. Accepting a run (and, if chosen, approving a step) then asks for the meaning of the signature from the organization's own list and a fresh second step; the record shows the printed name, e-mail, time with time zone and meaning. Changing a signed decision asks for a reason and the signature stays in the record as withdrawn. Tokens and agents never sign.

• The webhook event evidence.ready: a run's evidence is complete (shared, accepted or returned) or a release was recorded, with the state and where to fetch the evidence and the package.

• The acceptance record and its PDF list signatures, traceability, tools and environment, the AI models behind results, and advisory results such as health checks.

• Help: Kaloko in regulated work (GxP) and a supplier summary for quality assurance.

• kaloko health lists the health findings of a run by pack and step, with what is new or fixed since the previous run and how to fix each check; --json for agents, --remote for the shared run, --fail-on error for a team that wants health to stop a pipeline.

• Health checks have their own ids: HC101…HC705 (HC1xx seo, HC2xx usability, HC3xx a11y, HC4xx perf, HC5xx console, HC6xx errors, HC7xx app). Acceptance criteria stay AC. The number is the check's, so a pack kept in acceptance adds AC302 where the health check is HC302. Runs and scenario files written before keep working: health checks stored as AC302 show as HC302, the old ids work in kaloko health adopt, MCP and links, nothing stored is rewritten, and captures stay current.

• kaloko health adopt <step>:<check> [--text "…"] writes a health check into the scenario as an acceptance criterion of the step, as it is or in your words, so it counts from the next walk: home:HC103 becomes AC103 with from: HC103, and the step's health stops listing HC103. On the canvas, Adopt as criterion does the same, and kaloko health pull brings those adoptions into the scenario file.

• Health on the canvas has its own tabs: Health beside Summary in the run panel (counts, change since the previous run, findings by pack or by step) and beside Criteria in the step detail, which now holds acceptance criteria only. Arrow keys move between tabs. Review mode links to the step's Health tab. Plus a health badge on run lists and a health page per project with the trend over time. MCP: get_health and adopt_health_check.

• After a decision the step shows it: the approval panel opens with who approved or returned the step and when, the signature's meaning and time when signed, the note, the attached files and earlier decisions, and Change decision. Approvals are no longer tucked away as resolved comments. The step header and its card on the canvas show the decision too (with a mark when signed), and the run summary shows who accepted the run.

• Signature meanings come in the signer's language. Reviewed, Approved and Verified are translated; an organization adds a Czech label to its own meaning after a bar (QA release | Uvolnění QA). The signature keeps the label the signer saw, its id and the language. The signing dialog fills in the name from the profile, or from company sign-in.

• A signature now also covers the decision: its note and the SHA-256 of the attached files, pin and drawing. Earlier signatures keep verifying as they were made. The acceptance record lists the files attached to each step decision and the step's earlier decisions, and the evidence package holds the files under decisions/<step>/, listed in the signed manifest.

• Requirement references show on the canvas as chips: in the run info, under the step title, beside each criterion and in review mode. The index finds steps by them.

• Hosted runs and runs shared from CI tell you when health gets worse: a new error-level finding notifies the scenario's followers, the Slack and Teams channels that take "Health got worse" and webhooks with health.worse. Each schedule chooses errors (the default), warnings or no alerts, in Settings or with kaloko schedule add --health-alert.

• The agent skill checks health before handing work over and fixes what it can, without hiding findings.

Changed

• The semantic evaluator's prompt version is kaloko-jev-v1 (it was qawalk-jev-v1); answers cached before keep being used.

• Deleting a decision on a step goes into the audit log, and decisions on a run stay in the audit log as long as the run does (other entries still for two years).

• Check packs no longer decide acceptance. The checks of seo, a11y, perf, usability, console, errors and app still run on every walk, but they make up the run's health: shown apart on the canvas, counted by severity, compared with the previous run, and never part of a step's or run's verdict, the review queue, carried approvals or the acceptance record. A run no longer comes back red because of a meta description nobody asked for. The design pack still counts for acceptance. Runs shared earlier keep their verdicts.

• To keep a pack deciding as before, list it in the scenario: health: { acceptance: [seo] }.

0.14.0

Vydáno 2026-10-07

Added

• An admin of a paid organization can start another organization under the same e-mail: Plan & billing has the link, and the sign-up page says the new one gets the same address. Free organizations and code trials without a card do not open the way, so they cannot be chained.

• --json is a contract now: stdout holds only the JSON answer, progress and notes go to stderr, and a failed command ends stderr with one line {"error":{"message":"…","code":"…","next":"…"}}. New on share (the run id and its URL), evaluate, build, validate, scenarios, auth list, browser status and ci preview-url.

• Two more exit codes: 3 when the Kaloko service cannot be reached (no connection at all), 4 when the token is missing, invalid, expired or revoked. 0, 1 and 2 mean what they meant; an error answer from the service other than 401 is still 1, and docs check, drift and design lint still report their findings with 1.

• Lists page through the API: runs, comments, notifications, tokens and the audit log take ?cursor= and name the next page in X-Next-Cursor and Link: <…>; rel="next". Bodies keep their shape. The MCP tools list_runs, list_notifications and list_comments take cursor and answer next.

• kaloko runs --all follows those pages to the last run instead of stopping at 300; --limit N lists at most N and --cursor goes on where a list stopped.

• kaloko export run|steps|docs|live|design as sub-commands, each with its own --help. The flag forms (export --live, --design, --format, --scenario) keep working.

• kaloko affected --fail-if-none: exit 1 when nothing is affected. --check does the same and says it has a new name; workflows written by kaloko ci use the new one.

• Durations take units: --timeout 90s, ci preview-url --wait 10m, share --expires 72h, prune --older-than 14d. A bare number means what it meant before.

• default_token_mode in a design environment names its default token mode; tokens_mode still works.

• A step picture in docs and changelogs plays the step's recording. When the accepted run has a recording (kaloko walk --record), the picture shows a play button; a click or tap plays it without sound, and the picture comes back at the end. Nothing plays by itself. ?media=video in a step reference says the step needs its recording, and kaloko docs check warns when the run has none; ?media=image keeps the picture only. It works in "Walk through it step by step" too.

• Recordings that published docs or a changelog play are kept as long as they play them, instead of expiring after 30 days, and they count in storage.

• Settings -> Security -> Experiments shows how Clef triage does in your organization: useful findings per kind, how well its probabilities hold, findings turned into comments, the time to the first review on runs with findings and without, and triage requests per useful finding. The same numbers come from GET …/findings/stats and MCP get_triage_report.

• Exports carry the recordings: Markdown, Docusaurus, VitePress and offline HTML get a player with the MP4 and WebM in assets/, MkDocs a picture linked to the recording; the PDF keeps the picture. MCP resolve_step_image and GET …/steps/resolve answer video with the MP4 and WebM addresses and the length.

• Required approvers are asked by name. When a run waits for them, each approver whose e-mail is in your Slack workspace gets a direct message with a link straight into review mode, and Slack and Teams channels that take the new event "Waiting for approval" name them (Teams workflows cannot write to one person). One reminder follows after 24 hours if nobody has decided; the hours, the reminder and quiet hours are in Settings -> Integrations, and admins can read and change them through GET and PATCH /api/v1/orgs/<org>/approval-messages. Slack connected earlier needs to be connected once more so Kaloko can look people up.

• Hosted runs start after a deploy. kaloko schedule hook <id> (or Settings -> Integrations -> Hosted runs -> Run after a deploy) makes a trigger URL and a secret, shown once. A POST with the secret as a bearer token, an

HMAC signature of the body (X-Kaloko-Signature) or Netlify's signed deploy notification starts the run at the next quarter hour. The hosted runs guide has copy-paste examples for curl, Vercel and Netlify. Business plan and up, at most 30 triggers an hour per schedule, each one in the audit log.

• An invitation that cannot be delivered says so. When the mail provider reports a hard bounce or a spam complaint, the pending invitation shows "Address bounced: check the spelling" in Settings -> People and in the Share dialog, the person who sent it gets an inbox item, and Kaloko stops sending it. GET /invitations names the mark in bounced.

• Links to the pricing page, the comparisons, the situations, the "for…" pages and every guide show their own card in Slack, LinkedIn, X and other previews: the page's title and description on the Kaloko card, in the page's language. The other pages keep the default card.

Changed

• New API tokens start with kal_. Tokens that start with qwk_ keep working until they expire.

• Domain verification shows the TXT record _kaloko.<domain> with kaloko-verify=…. A _qawalk record added earlier still verifies the domain.

• Names from before the rename print one line on stderr when used: the qawalk command, QAWALK_* variables, qawalk.config.yml and ~/.config/qawalk/.env. They keep working; kaloko doctor lists them with their new names.

• Public docs and changelogs take their language from ?lang=, as the app does. Links, the canonical address, hreflang, the sitemap and llms.txt use it; the former ?language= (docs) and ?locale= (changelog) redirect there, so old links keep working.

0.13.0

Vydáno 2026-10-06

Added

• kaloko <command> --help (or -h) and kaloko help <command> show the help of that one command, with the options every command takes and the exit codes: 0 done, 1 checks found problems or the command failed, 2 a usage error. kaloko --version, -v and kaloko version print the version.

• kaloko start --json prints the run id and its folder, kaloko status --json what each step has captured.

• --org <slug> on every command works with another organization than org: in the config.

• API errors name their reason in code next to error. When the plan does not include something (HTTP 402) the answer links the organization's Plan & billing in billing_url, and the CLI prints that link.

• Webhooks carry Kaloko-Signature and Kaloko-Event next to QAwalk-Signature and QAwalk-Event, with the same values, so receivers written before the rename keep working.

• The trust page names a contact for reporting abuse (phishing, malware) on kaloko.app, files.kaloko.app and design previews at *.kaloko-usercontent.com.

• The trust page explains how Kaloko hides personal data in captures (masks, [hidden] in the stored HTML and the outline evaluators read, pii: redact) and links the how-to. The subprocessor list now names Workers AI at Cloudflare, which Clef triage uses only when an admin turns the experiment on.

• Every guide is also plain Markdown at its own address with .md added (/guides/ci.md, /cs/pruvodci/ci.md), for agents and answer engines. llms-full.txt now carries the pricing table and billing questions, the comparisons, the situations and the trust facts.

• The comparison pages answer the questions people ask about them.

Changed

• An option a command does not take gets a "Did you mean --env?" instead of being skipped quietly. prune, trash, restore, keep, archive, public, release and comment stop before they change anything; other commands say it and go on.

• A usage mistake (unknown command or option, missing argument) exits with code 2 and points to kaloko help

<command>, without the support line.

• runs, prune and screens take --env like the other commands; --environment still works. archive takes --off like keep and public (--undo still works), and prune takes --dry-run, which is what it does without --yes.

• A service account token on a plan without service accounts is answered with HTTP 402 (it was 403): the plan, not the token, has to change.

• Webhook requests say User-Agent: Kaloko-Webhooks/1.

• Pricing: the print-quality row is gone, a row shows that hiding personal data works on every plan, and another that the GitHub check on the pull request comes with Team. The site says plainly that your data stays readable on every plan and that exporting a run as a ZIP comes with Team.

• Terms of service, version 4: a trial started with an invitation code needs no card; if none is added, the organization returns to Free when the trial ends. The terms and the pricing page call the billing settings Plan & billing, as the app does.

• Kaloko's public changelog and help have one address per language, which search engines and link previews now understand. The Czech changelog page says that the release notes are written in English.

• /design and /cs/navrh lead straight to the guide on designing with your agent.

Fixed

• kaloko prune --env preview --yes pruned the runs of every environment: --env was not read. It now prunes only that environment.

• --key=value keeps everything after the first =: --map="Sign in=login" arrives whole.

• Commands run without org: in the config say how to set it instead of failing with a 404 on /orgs/undefined.

• kaloko runs says when the list stops at the newest 300 and how to narrow it.

• kaloko design import --json prints only the JSON on standard output; progress goes to standard error.

0.12.2

Vydáno 2026-10-06

Fixed

• Czech changelog headings (Přidáno, Změněno, Opraveno, Odstraněno, Zastaralé, Bezpečnost) count as Added, Changed, Fixed and the other Keep a Changelog groups, so a Czech CHANGELOG.md is grouped like an English one.

• A form that needed a fresh second sign-in step (inviting someone, changing a setting) is sent again after the step instead of coming back empty. The fields wait in the browser tab, never on the service.

Changed

• Screens on the canvas cards stay sharp: thumbnails are 640 wide and scaled down step by step (one big jump made small text fall apart), and when you zoom in past what a thumbnail holds, the visible cards load the full screenshot. Runs shared before this get the zoom part too.

• Videos play on: once you play a step's video, the next step's video starts by itself in the step detail, and replay with video moves to the next step when a video ends (a decision waits for you, a still step shows its screenshot for three seconds). The next step's video loads while the current one plays.

• A step's video is at least 2.5 seconds long, held on its last frame, so a quick step no longer flashes past. A step where nothing moved keeps no video at all, and the canvas shows its screenshot.

• kaloko.com and www.kaloko.com lead to the same address on kaloko.app.

0.12.1

Vydáno 2026-10-05

Added

• A trial code from us gives an organization's first subscription a longer free trial without a card. Open the sign-up link with the code, or enter it under Plan & billing. Two weeks before the trial ends, admins are asked for a card; without one the organization goes back to Free on its own and keeps its runs.

Changed

• kaloko share refuses unmasked personal data only in a run that hides it (pii on the run, scenario or environment, or mask: auto); other runs are shared with the same list as a warning. The site's own contacts no longer count: e-mails on the environment's domain, e-mails and phone numbers in the page footer, and those of an Organization or ContactPoint in JSON-LD (pii: { site_contacts: false } checks them again). Sample addresses such as you@company.com, placeholder text and SVG drawings are not read as personal data.

• Phone numbers written 603 123 456 count only next to a word like Telefon or Mobil, and amounts, EANs and order numbers no longer read as phone numbers. The nine-digit birth number needs its label (RČ), as does one without the slash.

• pii: pseudonymize gives one value the same stand-in in every run of the scenario, so screenshots and approvals carry over, and two values never share one. KALOKO_PII_SALT adds a secret of your own.

Fixed

• Hidden form fields, value attributes and personal data in URLs (also encoded, jan%40firma.cz) are masked.

• The capture record no longer keeps real data: page title, headings, JSON-LD, microdata, console lines, blocked requests, the URL and check evidence all go through the mask, and a capture whose masked page cannot be read stores nothing.

• Masking no longer changes what the page reacts to: console errors are read before it, data-* attributes are masked only in the stored HTML, and the keyboard walk runs on the real page with masked focus shots.

• Short masked values such as 2024 or Praha are no longer replaced across the whole run (width:100% stays).

• Recordings cover personal data in modal dialogs and popovers, open shadow DOM and same-origin frames, and on a new page from its first frame with everything masked so far. Screenshots of steps mask the same places.

• Values masked in one kaloko process are masked in the next process of the run too.

• Triage (experimental) fits long pages into Clef: two page tiles and smaller recording frames instead of four full tiles, and a request still too large is asked again with half the images. A Cloudflare token in the project that cannot use Workers AI hands triage to Kaloko instead of stopping it, and such a token is used only when evaluators.triage is set in kaloko.config.yml.

• A failure while masking leaves the page as it was.

• Long words without spaces no longer slow down detection.

0.12.0

Vydáno 2026-10-05

Added

• Triage, an experiment an admin switches on in Settings -> Security -> Experiments. kaloko evaluate shows each screen, and a few frames of its recording, to Clef on Cloudflare, which names the kind of problem it sees: overlapping elements, cut-off text, a raw translation key, a placeholder, an error, an endless spinner, a layout that jumps. The canvas lists them under "Possible problems"; one click turns a right one into an ordinary comment, and findings never change a verdict. evaluators.triage in kaloko.config.yml tunes or turns it off.

• MCP tools get_findings and answer_finding.

• Masks hide personal data in everything a step stores, not only in the screenshot. The captured HTML and the page outline the evaluators read get [hidden] of the same length ([skryto] in Czech runs), and traces, recordings, crops, focus shots and the HTML view are covered too. Checks still read the real page, so their verdicts do not change.

• pii: redact on a scenario finds e-mails, phone numbers, birth numbers, IBANs, account numbers, dates of birth,

addresses and names in greetings without a selector, in Czech formats too; capture: { mask: auto } does it for one step. pii.allow in kaloko.config.yml lists values that only look personal, such as your support line.

• pii: pseudonymize replaces personal data with made-up values instead of boxes. One value gets the same stand-in for the whole run, so you can still see a name travel from the form to the summary and the e-mail.

• Captured e-mails hide their recipients (To and Cc, now captured too) whenever a step masks anything; greetings and payer blocks follow the step's masks.

• kaloko share checks the run before it uploads: an unmasked e-mail, phone number, birth number or IBAN stops it, with the step and the element. --allow-pii uploads anyway.

• An environment with pii: required gets no run that would keep personal data; kaloko start --pii redact (or pii: on the scenario or the environment) satisfies it.

• The canvas says "Personal data hidden" on such runs and outlines the masked places in the step detail, so a dark box is not taken for a bug in the app.

Changed

• The masks of a scenario and of its step now add up; before, a step's capture.mask replaced the scenario's.

• kaloko.app serves its own fonts and the canvas you open from disk carries them inside, so no page asks Google Fonts for anything and Google Fonts is off the subprocessor list on the trust page.

0.11.0

Vydáno 2026-10-04

Added

• The approval page of kaloko login says where the request came from (city, country and address), which terminal asked and when, and starts with a plain warning: approve only a sign-in you started yourself, just now. When the request came from another country than the one you are in, the page says so in red.

• SCIM tokens are valid for a year. Settings -> Security -> SCIM shows the date, and "Renew for a year" keeps the same token, so nothing changes in your identity provider. Admins get a reminder in the inbox 30 days before a token expires. Tokens you already have run for a year from this release.

• A company sign-in over SAML set up before "Require a signed response" existed turns it on with one click in Settings -> Security. Kaloko offers the button once it has seen your identity provider sign its responses, so people keep signing in as before.

Changed

• A passkey used as the second step must ask for your PIN, fingerprint or face. A security key that signs without asking no longer counts; your account page names the key and says how to bring it back (set a PIN on it, or add another passkey).

• A run whose title, environment, branch, commit message or another field is too long is refused at the start of the upload with a message that names the field and the limit, instead of an unexpected error halfway through.

Fixed

• A browser that launched but then stopped answering no longer holds kaloko walk, evaluate --recheck or a capture until someone kills it. A new browser must open its first window within 20 seconds or it is closed and started once more, and later windows have the same limit. kaloko browser start, kaloko doctor, kaloko auth save, the PDF export and connecting to the shared browser now have the time limit and the second try too (KALOKO_LAUNCH_TIMEOUT, seconds).

• A js_equals check on a busy machine no longer fails when the page answers a little late: the expression keeps its 2-second limit, and Kaloko waits up to 17 seconds for the page to reply.

Security

• The sign-in cookie can be set only by kaloko.app itself. You stay signed in while it changes over. Your session also gets a new id at every sign-in and right after the second step.

• App and website pages tell the browser which features they never use (camera, microphone, location, payments and the like), and a Kaloko tab opened from another site is cut off from that site's window.

• Billing events from Stripe and pull request events from GitHub are applied once, also when someone sends a captured copy again.

0.10.0

Vydáno 2026-10-04

Added

• A step can list its other states: states: [empty, error, loading]. Walks run the step script once per state (it gets state), a design draws cart/empty.html next to cart/index.html, and each state gets its own screenshots and verdicts. The canvas has a State switcher next to the language, and review mode and the grid show the state you pick. A criterion with states: [empty] is judged on the empty cart only.

• A design whose token set has several modes, such as two brands, is rendered in each of them. Switch modes on the canvas like colour schemes; tokens_mode in the environment stays the default. Light and dark alone still follow color_schemes. Variants of a stack step are captured in every state and mode too.

• kaloko capture --state, kaloko spec --state and a state option on the MCP step tools (get_step, get_step_spec, get_step_drift, get_step_assets, get_step_recording).

• A design step can come in variants of one template, such as an e-mail for each reason support can pick. Make it a stack step and give each variant a folder, design/<step>/<variant>/index.html. kaloko walk captures every variant in every viewport, kaloko sync sends them to the draft, and in a published version the step opens a list of its variants with their results. Each variant opens live. kaloko design lint reports a variant folder without its page, and both design exports include the variant folders.

• Web fonts on live design steps. An organization admin lists the addresses designs may take fonts, stylesheets and images from (Google Fonts, Adobe Fonts, your CDN) in Settings -> Security, and live steps load them instead of falling back to a system font. Scripts still load only from the design itself. List the addresses under origins of the design environment and kaloko design lint says which of them the organization does not allow yet.

• Prototypes remember things between steps: a cart filled on one step shows on the next, in Live mode and in the presentation. The state stays in the viewer's browser tab, every step opened alone still shows its own sample data, and "Start over" in the presentation clears it. Design agents use window.kaloko.state.

Changed

• Large runs are quicker on the canvas. On a run of 500 steps in seven languages on three devices, with the CPU slowed down to a phone's, a zoom step takes 33 ms instead of 133 ms, switching between desktop and mobile 650 ms instead of 910 ms, and the map is drawn once when it opens (it was drawn twice). Scrolling the step index no longer sorts every step on each frame.

• On kaloko.app a large run's canvas data and a scenario's history page come from a cache after the first visit (27 ms instead of 113 ms, 9 ms instead of 230 ms), and the changelog, the products page, the screen library and review mode ask the database far fewer times.

0.9.0

Vydáno 2026-10-04

Added

• After a release, people who read the product see a short "What's new" bar in the app that links its illustrated changelog. It comes once per release and person; open the changelog or close the bar and it stays gone.

• The canvas tells you when the connection drops or Kaloko cannot save. Comments, approvals and verdicts wait and go out once you are back online, or with "Try again"; nothing is posted twice and nothing is lost quietly.

• kaloko doctor --fix installs a missing browser, creates missing folders and fills in org: from your token. An expired token points you to kaloko login.

• Long stack walks show roughly how much time is left.

Changed

• On a phone, review mode puts Approve and Return under your thumb at the bottom of the screen and says what you just decided and which step comes next.

• The first-steps list links straight to where each step is done: your newest run for comments and approvals, with the command to copy for the steps done in the terminal.

• Every error page has a way to write to support, with the address already in the e-mail, and a link to the status page. The app and website footers link the status page too.

• CLI errors say what happened and what to do next (sign in again, wait and retry, check the network) and end with support@sinfin.cz. Losing the connection no longer prints a stack trace.

Fixed

• A browser video encoder that stops responding no longer holds kaloko walk --record: the recording ends with a note and the walk goes on.

• Dark mode is easier to read: form fields have visible edges, the highlighted plan column on the pricing page is readable, and shortcut keys on the canvas's review buttons stand out.

0.8.0

Vydáno 2026-10-04

Added

• Steps and scenarios say what kind of page they are: page: landing, work, list, settings, form or doc. The usability pack asks questions that fit: "one clear primary action" only on landing pages, and on lists, settings, forms and docs a question of their own. Without page, public pages count as landing pages and app screens as work pages, so a signed-in overview is no longer asked for a single call to action.

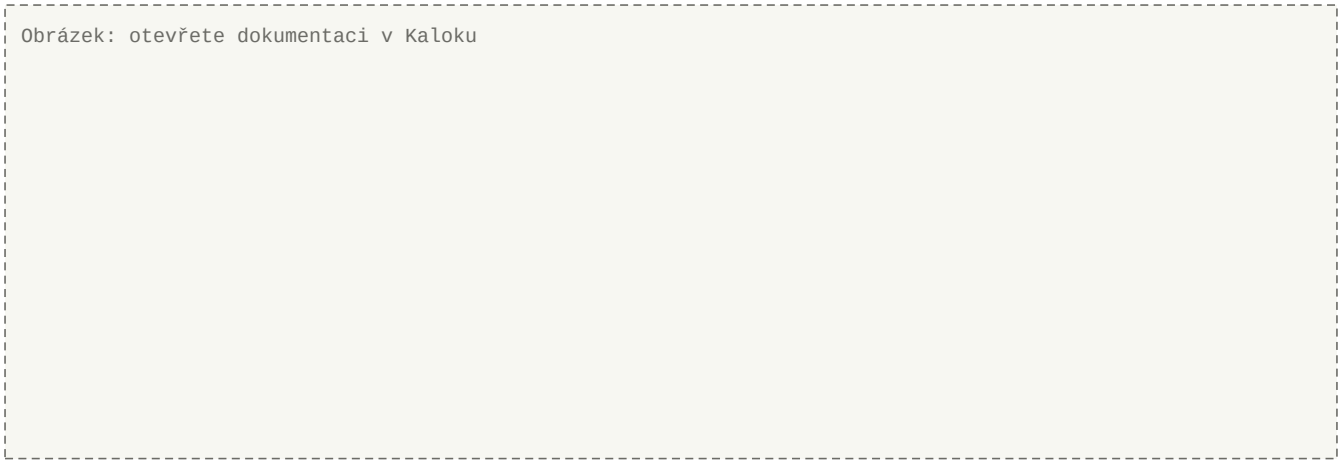
• A criterion can apply to some environments only: environments: [production] on hreflang pairs, a live demo or the production sitemap. On a local or staging walk it shows as not applicable to that environment and does not make the step partial.

• kaloko validate warns when a scenario walks several languages and a text check matches only one of them, for example pattern: "Plans and payment" without the Czech wording.

• Empty pages in the app say what to do next and give the command with a copy button: a product without a release or docs, its changelog and docs, an organization without runs, the runs and projects lists, the inbox and the screen library. People who only read a product see a plain sentence instead of a command.

• Kaloko's help covers the whole product, in English and Czech: two tutorials, how-to guides from writing a scenario to company sign-in, a reference of every CLI command, scenario key, check, pack, config key and MCP tool, and explanations of how verdicts are decided and how Kaloko maps to your applications.

• The website links Kaloko's own public changelog and help, so you can see both working before you set them up.



The home page · cs · desktop

• Company sign-in over SAML can require a signed response: the identity provider must sign the whole response, not only the assertion. New connections start with it on. Connections set up earlier keep working as before, and Settings -> Security recommends turning it on.

Changed

• Error pages say why you got there and how to go on. A missing page offers the overview or another sign-in; a page that needs a role you do not have offers an e-mail to your organization's admins, already written; an expired link says to ask for a new one; an ended session signs you in and brings you back to the same address.

• Unsent text on the canvas survives. The verdict, return and issue notes keep their draft like comments do, and when your session expires while you write, the canvas says so, keeps the text and returns you to the same step after you sign in again.

Fixed

• A browser that starts but never answers no longer leaves kaloko walk, video or a check waiting: each launch has a time limit and one more try (KALOKO_LAUNCH_TIMEOUT in seconds, default 45).

• kaloko init proposes products from npm, Yarn and pnpm workspaces and from packages/ too, not only from apps/; shared libraries without a dev, start or serve script are left out.

• kaloko stability --pull names the button people use on kaloko.app: Mark a changing region.

• A table in published docs can hold a pipe inside a cell, written as \|.

• Hosted runs open only pages whose address leads to the public internet. An address that points into a private network is refused when you schedule the run and again before each run. Saved passwords and cookies are sent only to https addresses.

• Text and attribute checks in hosted runs finish quickly whatever the pattern. Patterns with backreferences or lookahead are reported as not evaluated there, with the reason; kaloko walk still checks them in full.

• Signing in, sign-up, invitation links and the second step have their own limits on repeated attempts. Exports and new API tokens have a generous hourly or daily limit per person.

0.7.1

Vydáno 2026-10-04

Fixed

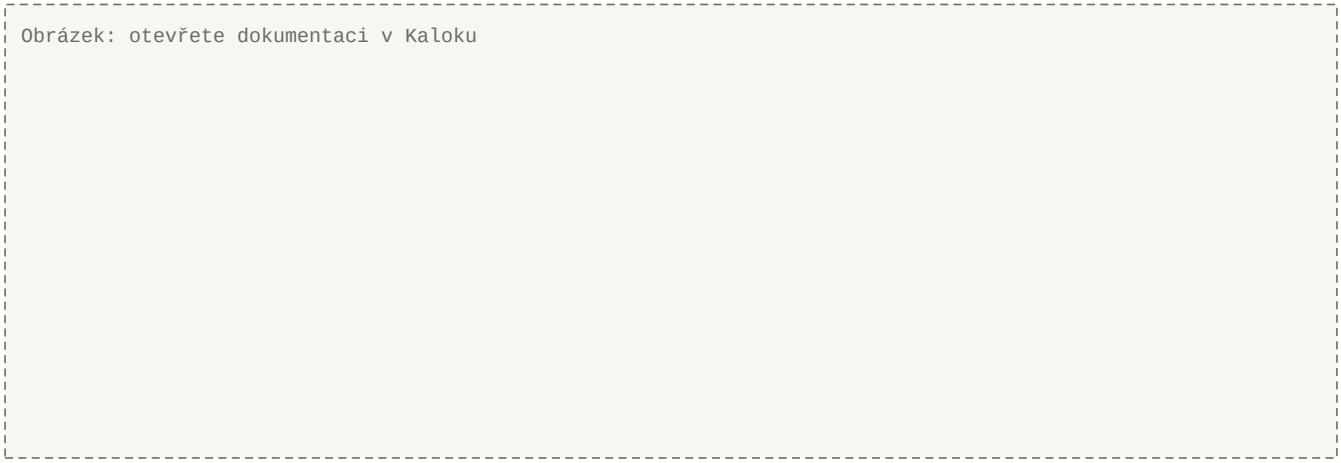
• kaloko walk of an Android, iOS, desktop or Electron app ends with the verdicts, like a walk of a website; before, the criteria stayed unchecked until you ran kaloko evaluate.

0.7.0

Vydáno 2026-10-04

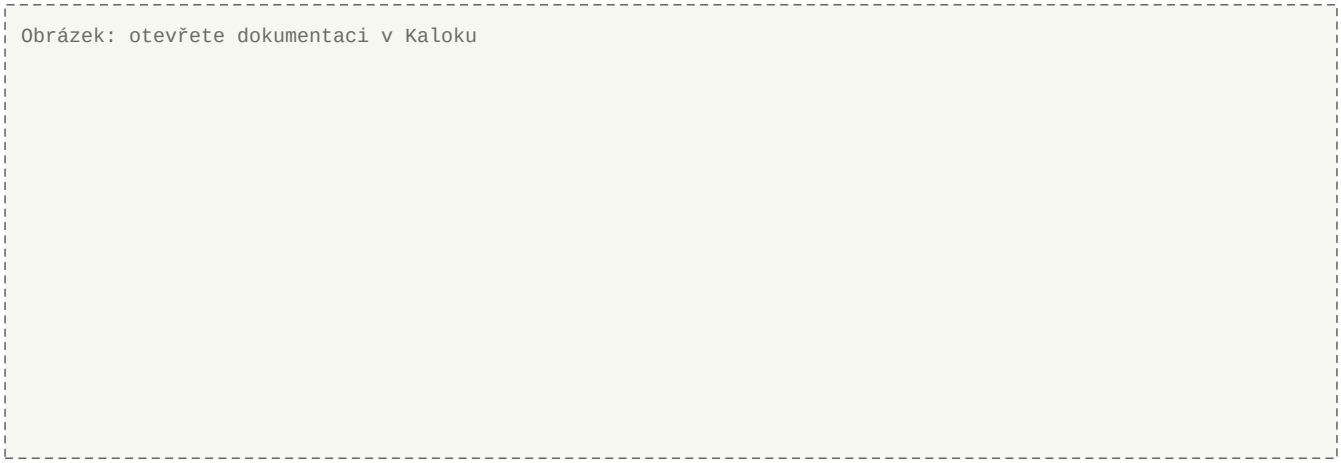
Added

• Products in the app. The main menu has Products: every product you read with its newest version, its docs, its modules and projects, who reads it and links to its changelog and docs. A project page names the products it belongs to. kaloko products sync puts the products of kaloko.config.yml on Kaloko before their first release, and kaloko products list shows which ones are there.



The Products page · cs · desktop

• A trust page at kaloko.app/trust (in Czech at /cs/duvera): where your data lives, how it is encrypted, who can get in, which companies process it and where our SOC 2 preparation stands. Kaloko is not SOC 2 certified yet, and the page says so. The security contact is also in /.well-known/security.txt.



The trust page · cs · desktop

• Seat warnings. When 80 % of the paid seats are taken, and again when all are, organization admins see a banner on the home page, in Settings and on the billing page, and get one e-mail per threshold in each billing period. GET /api/v1/orgs/<org>/usage reports the seats too.

• Yearly Enterprise subscriptions can go above their paid seats. Kaloko keeps the highest number of seats in each quarter; after the quarter, Kaloko billing reviews the true-up and invoices the extra seats for the rest of the subscription year or waives them. Seats added in the last quarter are not charged for that year: a week before the renewal the subscription rises to them and the renewal invoice includes them. The billing page shows the current quarter and every earlier true-up.

• Company sign-in over SAML accepts encrypted assertions. Create the organization's key pair under Settings -> Security -> Company sign-in; its certificate is in the SP metadata, and a new key pair keeps the old one working until you remove it.

• Single logout with your identity provider: signing out at the provider signs people out of Kaloko, and signing out of Kaloko can sign them out at the provider too.

• Sign-in from the app tile of your identity provider, off until an admin turns it on, with a default page people land

on.

Fixed

• The keyboard checks of kaloko walk no longer report a trap inside a third-party widget such as an anti-bot check, and judge a link wrapped over two lines and a small radio inside its label as people see them. Tab lists with arrow keys count as one tab stop.

• The canvas works better from the keyboard: every control shows a clear focus ring in light and dark mode, a "Skip to content" link jumps past the top bar, and while a step detail or another overlay is open, Tab stays in it instead of wandering through the hidden canvas. Screen readers get named side panels and dialogs.

• The canvas no longer jumps while it loads.

• A link from a comment notification opens the step with its comments in view, on a phone too.

• On a phone, a live page in the step detail fills the room above the sheet instead of a short strip, an open sheet leaves a strip of the preview that you can tap to lower it, and "Ctrl+Enter sends" shows only with a keyboard.

0.6.0

Vydáno 2026-10-03

Added

• A visual changelog. kaloko changelog draft writes the Unreleased entry of CHANGELOG.md from the pull requests since the last release and adds before and after pictures of the screens your accepted runs show differently. kaloko release <version> records the release with its pictures, and kaloko.app shows it as a page you can share with clients, support or anyone outside development.

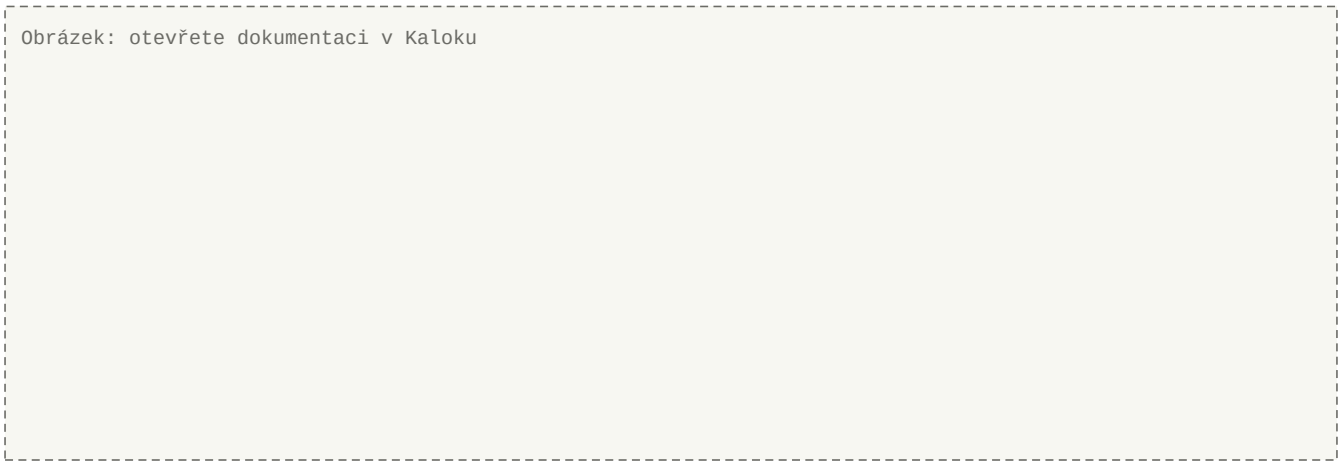
• Step references in Markdown: always shows the newest accepted screen, and ?version=2.4.0 the screen of that release.

• Products and modules in kaloko.config.yml for repositories with several apps; scenarios say which product they show (product:, module:). Without them the project is the product.

• Readers from your domain: everyone who signs in from a verified domain reads the changelogs, also of restricted projects (Business and up).

• kaloko init asks whether to keep a visual changelog, kaloko doctor checks that every product's version can be read, and agents get get_changelog, draft_changelog_entry and resolve_step_image over MCP.

• Docs that keep up with the app. Guides in your docs/ folder show steps instead of screenshots; kaloko docs publish puts them on kaloko.app with a version picker, language and screen switches and search, and every picture opens its step on the canvas.



A guide with its pictures · cs · desktop

• When a screen changes after its guide was written, the section says "the screen changed in 2.4 — check the text", and kaloko docs check lists it with broken references, missing translations and changelog items without a picture, so CI can stop on them.

• A guide plays as a walkthrough, one step at a time with its words as the caption, and readers tick off the steps they tried.

• Exports for your own docs site: kaloko export --format md, docusaurus, vitepress, mkdocs, html, pdf or json, with the pictures as files. The docs and changelog pages offer the same downloads.

• A "What's new" e-mail: readers ask for it on a product's changelog and get each new release with its pictures in their language (Business and up).

• kaloko ci github --release records every release that release-please, changesets or semantic-release publishes.

• Issues in Jira and Linear from a returned step or a failing criterion, on the canvas, in review mode or with kaloko issue create. The issue gets the screenshot, the criterion, the notes and a link back, and the comment on the step is resolved when the issue is done. Branches and pull request titles that name an issue link the run to it.

• Slack and Teams channels per project, each with the events it wants and quiet hours. Restricted projects reach only channels chosen for them.

• A short list of first steps on the home page (share, comment, approve, invite, CI, release) until the team has done them.

• Hosted runs: Kaloko opens the pages of a read-only scenario every day or week and shares the run (kaloko schedule add, Business). kaloko schedule export github writes a scheduled workflow for the full walk.

0.5.2

Vydáno 2026-10-03

Added

• The first version of the visual changelog: kaloko changelog draft, kaloko release, step references and the changelog page on kaloko.app with a before and after slider. It is described in full under 0.6.0.

0.5.1

Vydáno 2026-10-03

Added

• Swipe and onion skin compare next to side by side and diff, in the step detail and in review mode.

• Regions to ignore can be drawn by hand on a capture. The canvas diff skips them, and kaloko stability --pull writes them into the scenario.

• A grid (key G) shows one step in every language and screen size at once, also on a phone and in review mode.

• Comments can carry drawn boxes, arrows and lines, and attached images. Agents get the marked region in pixels.

• An accepted run has a signed acceptance record: a PDF with thumbnails and a printable page. kaloko signoff --verify checks the signature.

0.5.0

Vydáno 2026-10-03

Added

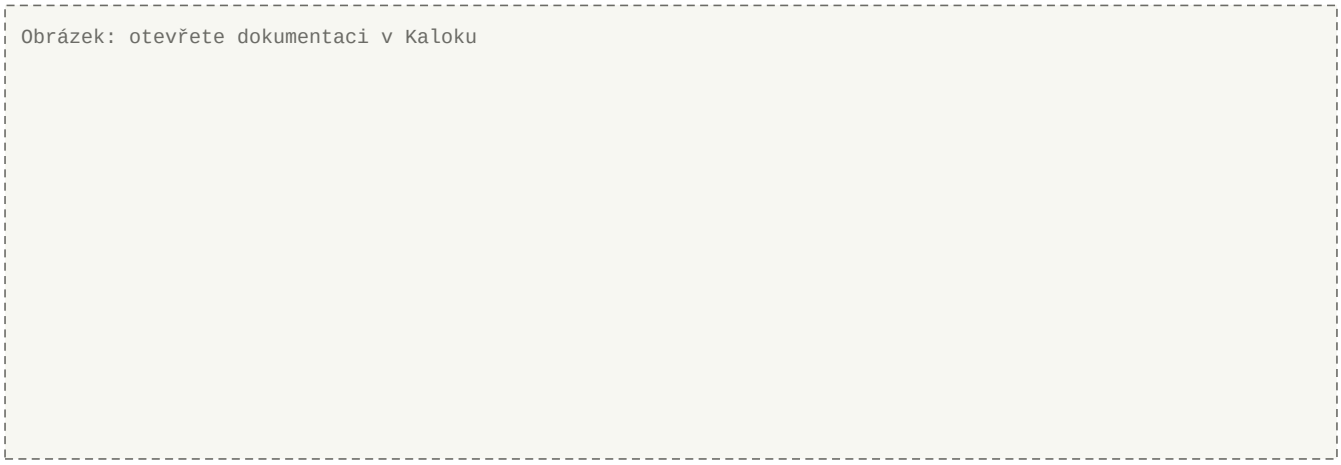
• Checks of the page itself: the address a step ends on, computed styles of an element and values the page exposes, so scenarios no longer need data-qa markers.

• Runs in WebKit and Firefox besides Chromium, on device presets (phones, foldables, tablets, TV), in dark mode, with reduced motion, forced colours or as an installed web app. Every step is captured in each combination.

• App walks on several devices at once: Android emulators, iOS simulators, a real iPhone and Android

WebViews, and Windows apps next to macOS ones.

• One picker on the canvas for language, browser and colour scheme that stays usable with many values.



The canvas of a run · cs · desktop

• Workflows written by kaloko ci install the browser engines your scenarios walk.

Fixed

• Design drafts keep the browsers, schemes and device presets of their scope.

• An e-mail captured during a walk in several browsers counts once.

• Language and theme switchers work in the phone menu.

• Ignored regions, imported screenshots and cached evaluator answers carry over between capture variants.

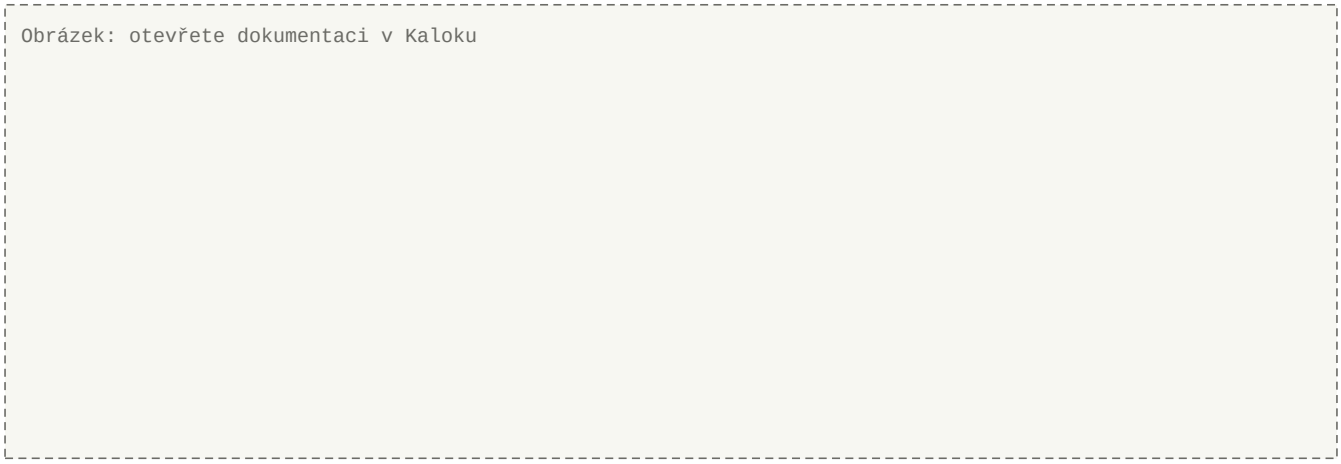
0.4.11

Vydáno 2026-10-03

Added

• Recordings: kaloko walk --record films each step from the previous screen to its capture, or the whole flow with a chapter per step, and keeps a trace of actions, requests and console messages for every step.

• The canvas plays a step's recording and shows its trace.



A recording on the canvas · cs · desktop

• kaloko video export turns the recordings into one walkthrough video with a title card per step.

• Shared runs carry their recordings on the Business and Enterprise plans, kept for 30 days unless the organization chooses otherwise.

Fixed

• A recording too large to upload stays in the local preview instead of failing the share.

0.4.10

Vydáno 2026-10-03

Added

• kaloko ci github|gitlab writes a workflow that walks what each pull request changes on its preview deployment and comments on the pull request; kaloko ci preview-url finds the preview address in CI.

• kaloko import lays out screenshots, recordings and traces that an agent or Playwright already produced as a run.

• kaloko walk --affected and kaloko affected walk only the steps that the changed files reach, and say why.

• kaloko walk --heal proposes a new selector when the page lost the old one, and applies it once the step passes.

• Several languages walk at the same time (--concurrency).

• kaloko init recognises the framework, the dev server, languages and main routes, and drafts a first scenario.

• Uploads skip screenshots the service already has and resume after a broken connection.

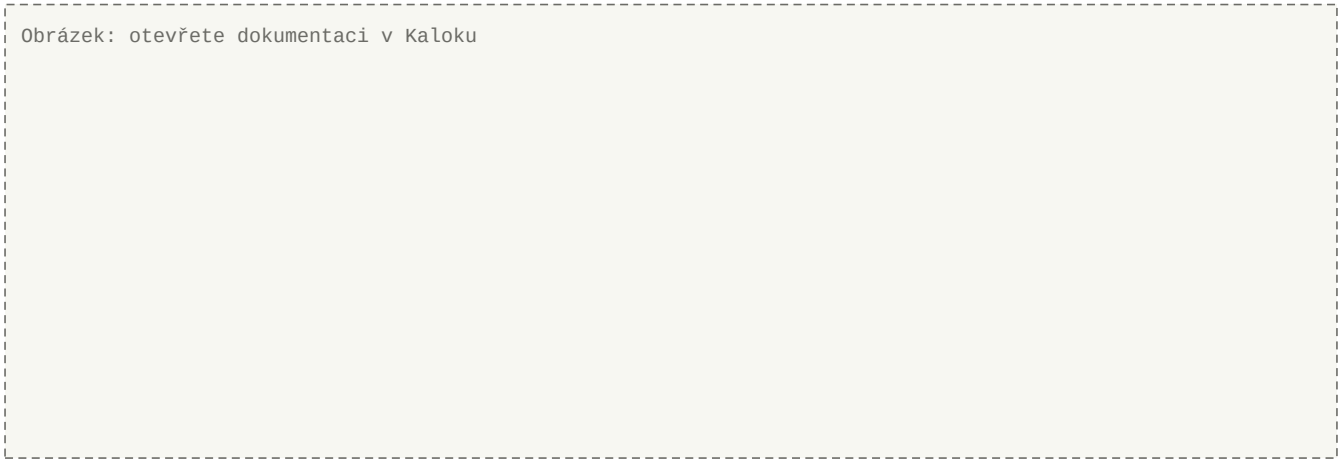
0.4.9

Vydáno 2026-10-03

Added

• A vision evaluator that looks at the screenshot and says why, keyboard and screen reader checks, and regions that change on every run can be ignored.

• Dev mode in the step detail of a design: box model, tokens, redlines, states and assets.

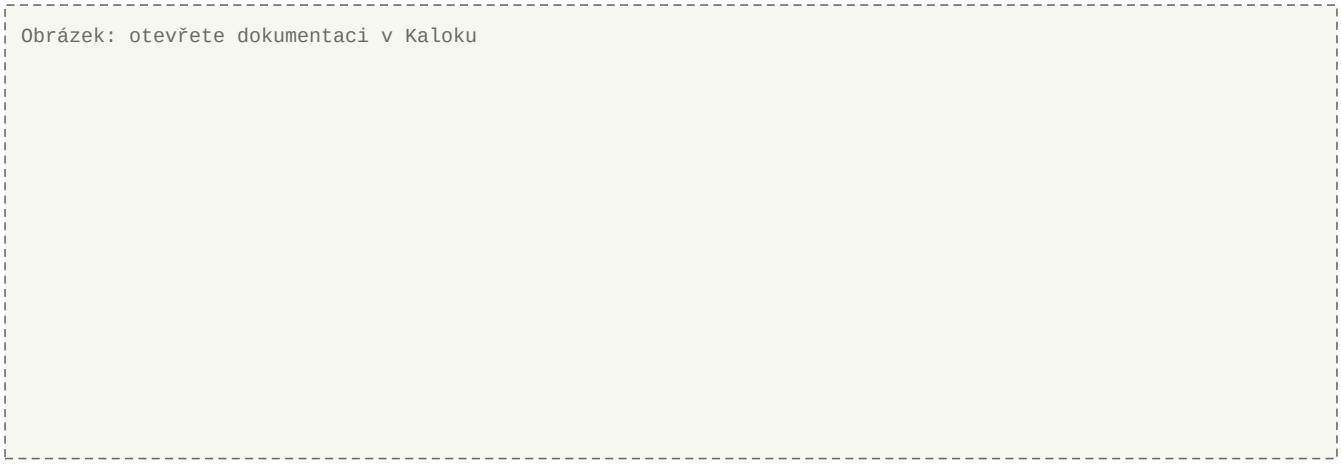


Dev mode in the step detail · cs · desktop

• kaloko spec tells an agent what to build for an approved design step, and kaloko drift what its implementation does differently.

Fixed

• The canvas on phones: a smaller stack badge, readable stack lists, the step header above its tabs and no sideways scrolling.



A stack page on a phone · cs · mobile

• Page counts in Czech use the right plural.

0.4.8

Vydáno 2026-10-03

Added

• Review mode: the canvas walks a person through what still needs a decision, with keyboard shortcuts, and kaloko review lists the same for agents.

• Approvals carry over to steps whose screenshots did not change since the last accepted run.

• Every new organization gets a sample run with a guided first review.

• Two-step verification with passkeys or an authenticator app, and an organization rule that requires it.

• An audit log for admins, tokens limited to chosen projects, and company sign-in with SAML or OIDC, SCIM and an IP allowlist on Enterprise.

Fixed

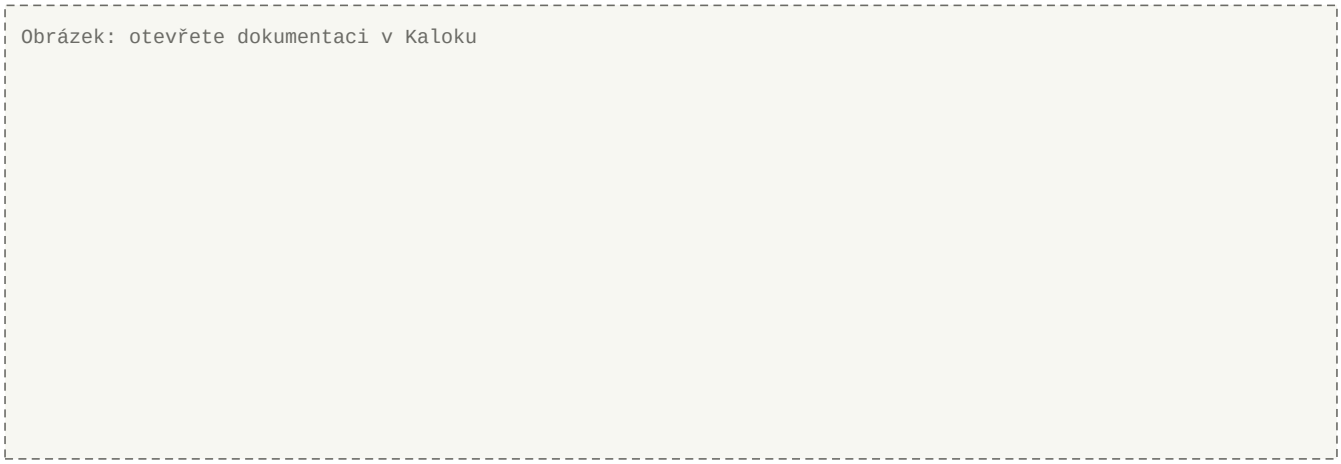
• Plan badges on the pricing page no longer overflow on phones.

0.4.7

Vydáno 2026-10-03

Added

• Sign in with Google, Microsoft or GitHub where the service offers it.



The sign-in page · cs · desktop

• kaloko login approves a token in the browser instead of copying it from Settings.

• Invitations: people outside the organization join single projects as guests, and the Share dialog of a project shows who has access.

0.4.5

Vydáno 2026-10-03

Added

• A Designer role and API tokens with chosen scopes (read, comment, design, upload).

• Focus mode shows the preview alone, and the step detail has a layout for phones.

0.4.4

Vydáno 2026-10-02

Fixed

• kaloko evaluate --recheck says that it asks the semantic questions again too.

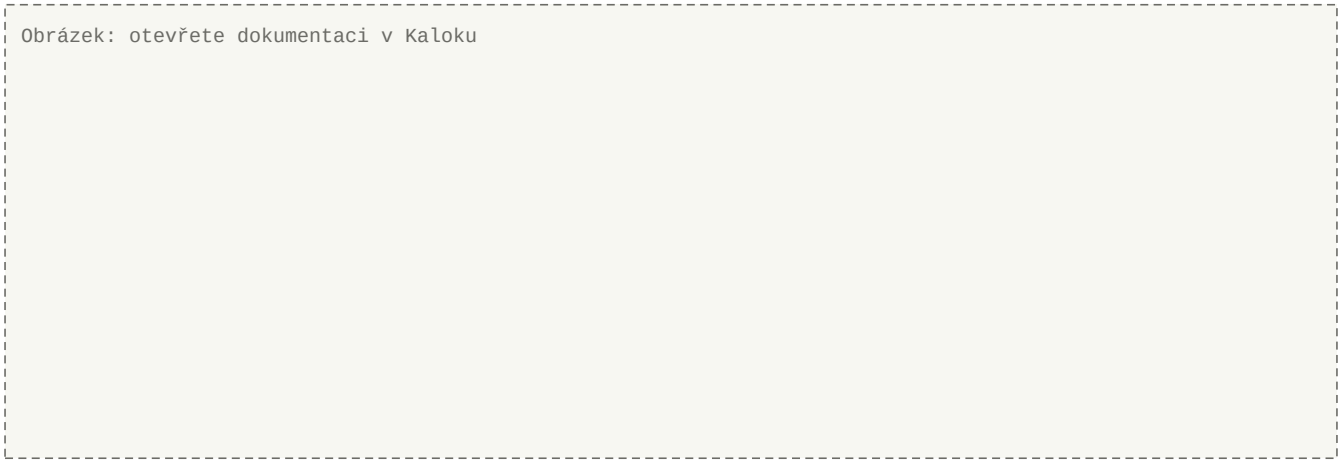
• A lane on the canvas is as tall as its tallest card.

0.4.3

Vydáno 2026-10-02

Fixed

• Signed-in pages that Kaloko's own walk found wanting: labelled fields, real settings tabs, cards on phones and the whole inbox.



Settings · cs · desktop

• The canvas has a main landmark, readable badges and a visible compare button, and no longer shifts while loading.

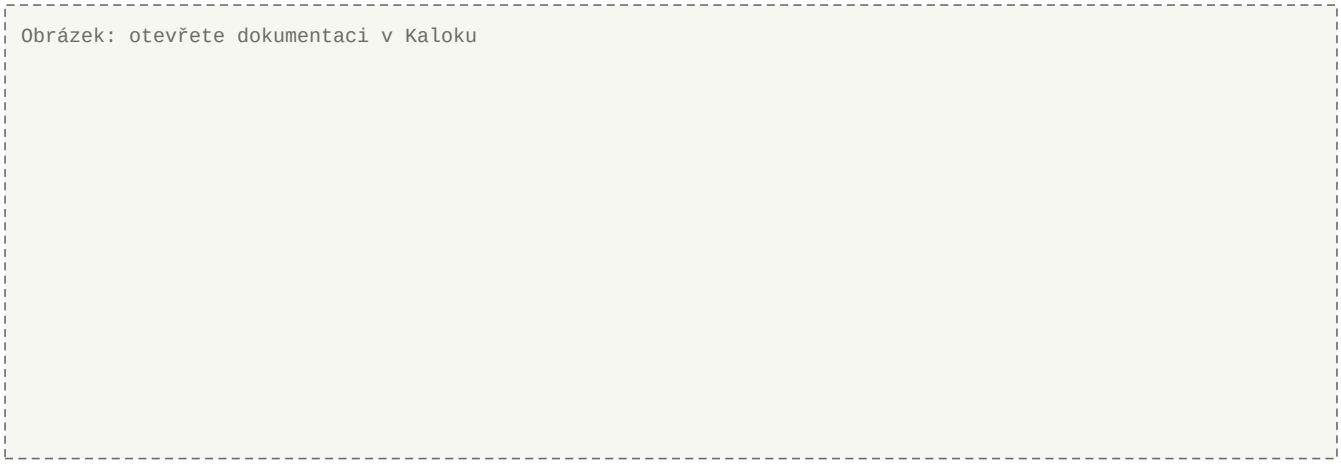
• Pages load their fonts sooner and keep their layout when the font arrives.

0.4.2

Vydáno 2026-10-02

Added

• The Kaloko wordmark with a yellow marker and a new icon.



The home page · cs · desktop

• The Czech website lives under /cs/, so each language has its own address.

• Pages of a stack may carry their own purpose for the semantic evaluator.

• Live design revisions open on their own domain.

Fixed

• kaloko doctor prints the config file it actually read.

0.4.1

Vydáno 2026-10-02

Added

• links_ok and hreflang_pairs checks for pages and stacks.

Fixed

• Unknown addresses on kaloko.app answer 404 instead of the sign-in form.

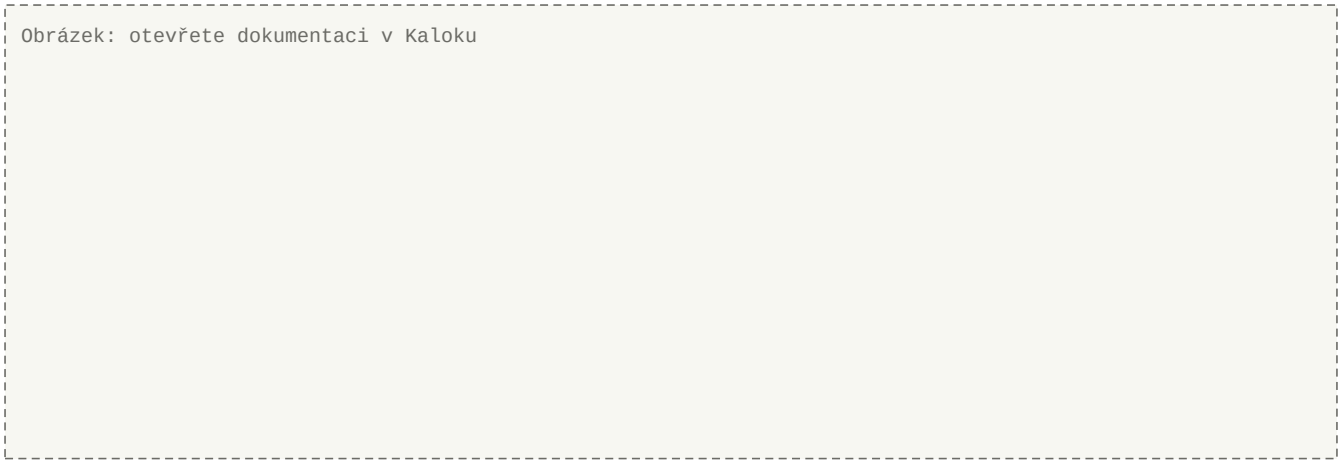
• Every language version of the website answers at its own address.

0.4.0

Vydáno 2026-10-02

Changed

• QAwalk is now Kaloko: the CLI is kaloko (the qawalk command and qawalk.config.yml keep working) and the service lives at kaloko.app.



The landing page · cs · desktop

Added

• Kaloko Design presents interactive HTML designs, lints pages that must work from disk, and exports a design in open formats.

0.3.0

Vydáno 2026-10-02

Changed

• The CLI is on npm as kaloko instead of @qawalk/cli. The qawalk command, qawalk.config.yml, the QAWALK_* variables and ~/.config/qawalk/.env keep working, and kaloko init replaces the old skill.

• The app and the website share kaloko.app. Pages on qawalk.com redirect there; the API, MCP, webhooks and file links on the old addresses keep answering.

Added

• A "Group by issue" switch on the lists of runs and scenarios. The lists remember it, and "Latest per scenario" too.

0.2.22

Vydáno 2026-09-30

Fixed

• Large runs open faster on the canvas, and kaloko evaluate needs less memory for big stacks.

0.2.21

Vydáno 2026-09-30

Fixed

• A kept run cannot be trashed, and releasing it gives it at least a week. Kept design versions are not pruned.

• The "no project" filter on runs works, failed step details load again, and an unusual cookie no longer breaks the lists.

• Lists, the home page and scenario history load faster.

0.2.20

Vydáno 2026-09-30

Added

• The project you pick is remembered across the lists and the scenario browser on the canvas.

• Scenarios and runs of one issue, or of one pull request when there is no issue, are listed together.

0.2.19

Vydáno 2026-09-30

Changed

• Shortcuts and help open from a keyboard button in the top bar, and the list now includes <- -> and Esc.

0.2.18

Vydáno 2026-09-29

Changed

• A shorter top bar on the canvas: Projects, Scenarios and Runs, the view switch, and the inbox at the right end.

0.2.17

Vydáno 2026-09-29

Changed

• Projects, scenarios and runs share one layout with the navigation on the left, so switching views does not jump. "All runs" is now "Runs".

• The run verdict has Accept and Return as the main buttons; note, baseline and public sit below them.

0.2.16

Vydáno 2026-09-29

Added

• A projects page: scenarios, runs and how the newest runs stand in each project.

• Console errors in the step detail are highlighted, wrapped and can be copied one by one or all at once.

• Times follow the reader's time zone, set in Settings -> Profile or taken from the browser.

Changed

• Screen size switches say the name and width (Desktop 1440, Mobile 360), and a click anywhere on a card opens the step.

0.2.15

Vydáno 2026-09-29

Changed

• The step detail has three tabs: Criteria, Page and Technical. Criteria that need attention come first; passing ones take one line.

• When only phones are shown, the map shows larger phone screenshots.

0.2.14

Vydáno 2026-09-29

Changed

• Evidence of built-in checks reads as a sentence in Czech or English ("TTFB 950 ms, the limit is 800 ms"); the raw text stays under Technical detail.

0.2.13

Vydáno 2026-09-29

Added

• Every criterion of the built-in packs says why it matters and how to fix a failure, in Czech and English. The walk summary prints the fix for the agent.

0.2.12

Vydáno 2026-09-29

Changed

• The run summary sorts problems by what to do: what failed and why, what a person should decide, and screens that look off.

0.2.11

Vydáno 2026-09-29

Added

• Quick signals on every screen: the evaluator also says whether the page does what it is for, is in the expected language or looks broken. Signals never change a verdict.

• kaloko walk evaluates while it walks and ends with a list of what failed or is unsure.

0.2.10

Vydáno 2026-09-29

Added

• Every criterion says how it was checked, in words: the check as a sentence, or the question the AI was asked with its score and the pass threshold.

• Hosted AI evaluation from the Team plan up, with a monthly allowance shown on the billing page and in kaloko doctor. Your own evaluator key still works.

• Design tokens in the current DTCG format, with aliases, modes and deprecated tokens; kaloko tokens validate.

Fixed

• Manual decisions show the criterion on its own line, with reason and verdict below.

0.2.9

Vydáno 2026-09-29

Added

• Scenarios come first in the navigation, and Runs show the newest run per scenario unless you filter.

• Keep a run so it never expires, prune old runs (dry run first) and archive scenarios you no longer walk, in the app or with kaloko runs, trash, restore, prune, keep and archive.

0.2.8

Vydáno 2026-09-29

Changed

• A claimed sign-in domain lets people join only once it is verified.

• Comments on public runs name nobody, and upload tokens cannot delete runs.

• Below 1500 px the navigation and view options fold into one menu.

Fixed

• The canvas shows the map sooner and loads step details after it; large runs upload faster.

• When the service asks the CLI to slow down, it waits and tries again.

0.2.7

Vydáno 2026-09-29

Added

• Comments on steps, whole runs and whole flows: threads, pins on the screenshot, resolve and reopen. Mentions are mailed at once, the rest goes to the digest, and agents use kaloko comment or MCP.

• Kaloko Design: design drafts as live HTML steps on the canvas, versions, an inspector with token names, comments pinned to elements, edits on the canvas, branches linked to git, and a design pack that checks an implementation against the approved design.

• An HTML view of captured pages (experimental) that can be forked into a design draft.

Changed

• A calmer run view on smaller screens: one top bar, view options in a popover, help behind ?.

Fixed

• A long step path no longer widens its card over the next one.

• macOS desktop captures ask for the Screen Recording and Accessibility permissions they need.

0.2.6

Vydáno 2026-09-29

Added

• Mobile and desktop apps: Android, iOS Simulator, Electron and macOS windows, with UI tree checks, touch target sizes, crash detection and an app pack. kaloko capture --image imports screens from other tools.

• A run index next to the map (key I): every step in a list with filters, search and CSV, and stacks as folders.

0.2.5

Vydáno 2026-09-29

Added

• The HTTP status of every capture, an errors pack for error pages, and {random} in a path for an address that cannot exist. kaloko draft adds a not-found step.

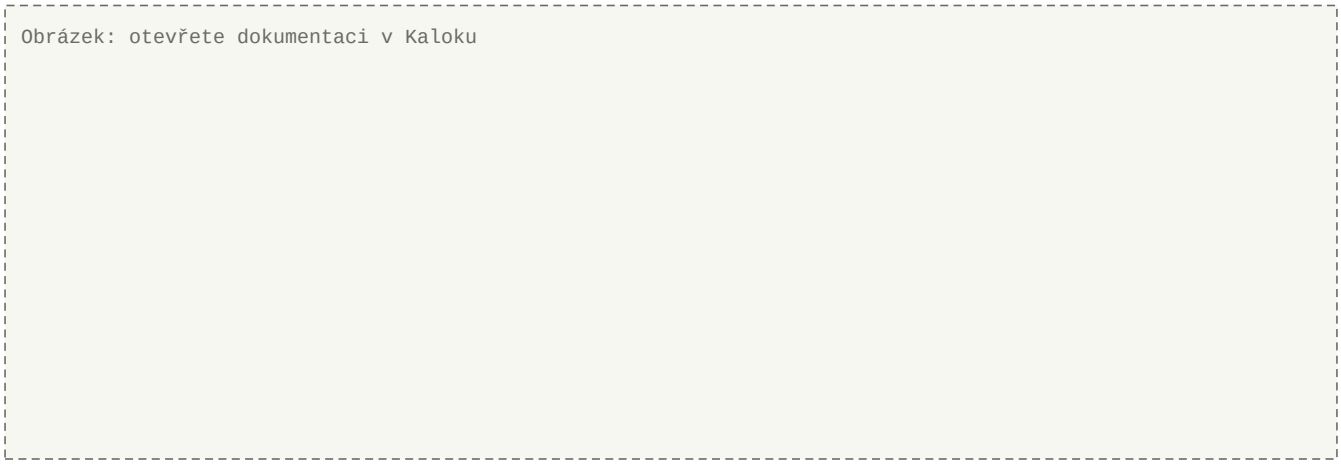
• The run card links the issue next to the pull request.

0.2.4

Vydáno 2026-09-29

Added

• Stacks: one step stands for many pages of one template, taken from a list, a sitemap, a crawl or a script. The canvas shows them as a stack with an index you can filter, search, sort and export as CSV.



A stack opened as a folder · cs · desktop

0.2.3

Vydáno 2026-09-28

Added

• acceptDialogs() for step scripts, flags on text checks, kaloko evaluate --recheck after a scenario fix, and kaloko

walk --ephemeral for parallel walks.

Fixed

• kaloko mail reads only messages received since the run started.

• A walk captures fresh screens after --steps, basic auth survives a redirect, and only one walk at a time uses a shared browser.

0.2.2

Vydáno 2026-09-28

Added

• kaloko draft <url…> writes a first scenario and acceptance plan for public pages.

• Check packs a11y, perf and console.

• Live presence on the canvas: who is looking, their cursors, and following someone's view.

• Large flows stay readable: lanes with headers, long lanes wrapped into rows, tidy edges and a zoomed-out view.

• Scenario history with pass rate and criterion stability, and kaloko calibrate to tune evaluator thresholds.

• Print-quality captures for documentation, with masks over personal data and numbered marks.

• The newest capture of every step at a stable address, a list of changes, signed webhooks and kaloko export --scenario for documentation (Business and up).

• GitHub checks on pull requests that complete when the run is accepted or returned.

• A sign-in domain can be proved by signing in with Google Workspace or Microsoft 365, and Settings shows the DNS host with copyable record fields.

Fixed

• The overview follows reviewers' criterion verdicts.

• Public mailbox domains such as gmail.com cannot be claimed.

• A walk stops a session after three steps fail with the same error, instead of trying a wrong password again and again.

• An open menu or popover stays on the capture, signed links in the page lose their credentials, and text check evidence shows the actual match.

• kaloko share says when the plan grants a shorter lifetime than requested.

0.2.1

Vydáno 2026-09-25

Added

• Protected environments: HTTP Basic, service-token headers and client certificates, sent only to the environment's own addresses.

• Accounts per scenario with password and TOTP, and kaloko auth save for a single sign-on a person does once.

• ${VAR} in base_url for review apps, and secrets named in the config are removed from captures.

• Any mailbox through a script adapter, and kaloko mail --file for saved messages.

0.2.0

Vydáno 2026-09-25

Added

• The CLI is on npm (as @qawalk/cli until 0.3.0).

• Plans with a billing page, a 14-day trial and a public pricing page.

• Named sessions: several people in one flow, each in their own browser, shown in swimlanes on the canvas.

• Run verdicts, a baseline run, required approvers and reviewer verdicts for manual criteria.

• Compare a run with the baseline or any earlier run: pixel differences, changed criteria and evidence, and kaloko compare.

• Replay walks the run one step at a time with a branch choice at decisions, and the canvas has a minimap.

• An inbox with notifications, a daily or weekly digest, and Slack.

• kaloko share --pr comments on the pull request; kaloko export and the canvas download a run as a ZIP.

• An MCP server for agents.

• Check packs seo and usability, plan coverage with kaloko validate --coverage, and flaky criteria.

• A public demo run anyone can open without signing in.

• Projects per scenario, and a runs list filtered by project, scenario, verdict, author, environment and branch.

• Sign-in domains verified by a DNS record, token scopes and expiry, a list of your sessions, and a 7-day trash.

• kaloko doctor checks your setup and says what to fix.

• Guides on the website for use cases and agents.

0.1.0

Vydáno 2026-09-23

Added

• The first version. The CLI walks scenarios in Chrome, captures every step, runs the checks and asks an AI evaluator what a check cannot answer. A read-only policy keeps production safe.

• The canvas shows the run as a map of screens with criteria, page metadata (Open Graph, hreflang, structured data, headings) and links to a single step. A click on a criterion highlights it on the screenshot.

• The agent skill for Claude Code and Codex, installed with kaloko init --agent.

• The service: organizations by e-mail domain, sign-in by link, roles and API tokens. kaloko share uploads a run, reviewers approve or comment on each step, and kaloko feedback brings their notes back to the agent.

• The app and the website in Czech and English, with sign-up for new organizations.