HelixQA

HelixQA

تنسيق فحص الجودة المضاد للخداع — جلسات مستقلة عبر المنصات حيث يحمل كل نجاح دليلاً ملموساً على أن المستخدم الحقيقي قادر على استخدام الميزة.

ملخص

HelixQA هو إطار عمل لتنسيق فحص الجودة المضاد للخداع لاختبار المنصات المتعددة (أندرويد، وكشف الأعطال في الوقت، YAML أندرويد تي في، الويب، سطح المكتب) يجمع بين بنوك الاختبارات المدعومة برؤية الحاسوب LLM الفعلي، والتقاط الأدلة خطوة بخطوة، وجلسات فحص الجودة الذاتية لإثبات عمل الميزات بشكل حقيقي من البداية إلى النهاية. وهو نوع الاختبار الإلزامي وفقاً لـ Constitution (§11.4.169).

وصف مختصر

منسق فحص الجودة المضاد للخداع (Go) الذي يشغل بنوك الاختبارات المكتوبة وجلسات فحص الجودة الذاتية بالكامل المدفوعة بـ LLM والرؤية الحاسوبية عبر المنصات — يكشف الأعطال، ويتحقق من كل خطوة مقابل الأدلة الملتقطة (لقطات الشاشة، سجلات النظام، مقاطع الفيديو، تتبع المكدس)، وينشئ تلقائياً تذاكر غنية بالأدلة لخطوط إصلاح AI.

وصف مفصل

يتمحور تصميمه الوحيد وغير القابل للتفاوض حول القاعدة التنفيذية لـ Go هو إطار عمل HelixQA Constitution §11.4: معيار الشحن ليس "اجتياز الاختبارات" بل "قدرة المستخدمين على استخدام الميزة"، لذا يجب أن يحمل كل نجاح يصدره دليلاً إيجابياً تم التقاطه أثناء التنفيذ — لا دليل، لا نجاح، بلا استثناءات. يعمل هذا الإطار بنمطين متكاملين يغطيان معاً السيناريوهات المكتوبة وغير المعروفة. أولاً، **بنوك الاختبارات المكتوبة** مجموعات YAML من حالات الاختبار مع TC-XXX — استهداف المنصة، الأولوية، الخطوات المرتبة (الاسم/الإجراء/المتوقع)، العلامات، والمراجع الوثائقية (ADB لأندرويد، ومراقبة العمليات للويب/سطح المكتب)، وجمع الأدلة المركزية، وإنشاء تذاكر Markdown تلقائياً جاهزة لخطوط إصلاح AI. ثانياً، **جلسة فحص الجودة الذاتية بالكامل** التي تسلم التطبيق لوكلاء مدعومين بـ LLM والرؤية الحاسوبية ليقودوه دون تدخل عبر أربع مراحل منظمة: الإعداد (اختيار نماذج اللغة الكبيرة، بناء خريطة الميزات من وثائق المشروع، إطلاق وكلاء CLI، تهيئة محرك الرؤية)، التحقق

المدفوع بالوثائق الذي يتحقق من كل ميزة موثقة، الاستكشاف المدفوع بالفضول الذي يختبر حالات الحافة والسلوك غير الموثق، ثم إعداد التقرير والتنظيف إلى صيغة Markdown/HTML/JSON مع ربط كل نتيجة بدليل مرئي مؤرخ بالفيديو.

الأهم من ذلك، أنه لا يصحح عمله بنفسه: فهو يدمج أربعة وحدات فرعية خارجية Go ويعيد (LLMsVerifier، LLMOrchestrator، VisionEngine، DocProcessor) بحيث لا يكون المكون الذي يتنقل في التطبيق هو نفسه المكون الذي يحكم على نجاحه. تخضع مجموعته الخاصة لنفس المعيار الذي يفرضه على الآخرين عبر `make anti-bluff` (فحص ثابت + بيان مرساة السلوك + عجلة الطفرات) وتحدي منسق من ثماني مراحل مع طفرة مدمجة في §1.1. تربط مصفوفة تغطية أنواع الاختبار المكونة من 15 صفاً كل قدرة معلن عنها بأصل تنفيذي ملموس وشكل محدد للأدلة الملتقطة — بحيث تكون ادعاءات الإطار عن نفسه مقيدة بالأدلة تماماً كما هي الأحكام التي يصدرها بشأن المنتجات التي يختبرها.

المحتوى

لماذا قمنا ببنائه

تسمح أنظمة ضمان الجودة التقليدية بإعطاء الضوء الأخضر بناءً على "نجاح التأكيد"، وهذا بالضبط ما يسمح لفئة الأخطاء التي يسميها Constitution بـ التضليل — ميزة يُبلّغ عن عملها بينما هي معطلة بالنسبة للمستخدم الحقيقي. بُني HelixQA لمنع حدوث ذلك تحديداً في ضمان الجودة: فهو يرفض تسجيل نتيجة "نجاح" دون دليل مادي ملموس (لقطة شاشة، سجلات النظام، فيديو، تتبع المكدس، تقرير) تم التقاطه أثناء التنفيذ الفعلي، ويعامل السطر الأخضر في الملخص الخالي من هذه الأدلة كخلل حرج يعادل غياب ميزة كاملة. كما يحل مشكلة العمل اليدوي — إذ لا يمكن لضمان الجودة اليدوي الشامل عبر منصات متعددة أن يتوسع — بجعل الجلسات مستقلة تماماً.

لماذا يعد تغييراً جذرياً

إنه يدمج شيئين نادراً ما يجتمعان في أداة واحدة: بوابة ضمان جودة صارمة قائمة على الأدلة واستكشاف مستقل ذاتي القيادة. يعمل وكيل LLM الفعلي المدعوم بالرؤية على فتح التطبيق، والتحقق من كل ميزة موثقة، والبحث عن الأخطاء غير الموثقة التي لم يكتب لها أحد اختباراً، ويُنتج في الوقت نفسه مسار أدلة بجودة محكمة — هذا هو الفيديو، وهذه سجلات النظام — ليحل محل عبارة "لقد اختبرناها" بعبارة "هذا هو التقرير". ولأنه وحدة فرعية لضمان الجودة تحمل اسم Constitution، فإن تبنيه لا يرفع مستوى أمانة ضمان الجودة لفريق واحد فحسب — بل يرفع الحد الأدنى لكل منتج مستهلك في العائلة بحركة واحدة.

ما الجديد فيه

- **عقد الأدلة المضاد للتضليل** — يرتبط نجاح كل فحص بأدلة تشغيلية ملتقطة؛ فالسطر الأخضر في التكامل المستمر يُعتبر ضرورياً لكنه غير كافٍ أبداً، والسطر الأخضر الخالي من الأدلة يُقيّم كخلل حرج.
- **استكشاف مستقل مدفوع بالوثائق والفضول** — يخرج عن النص ثم يتحقق من كل ميزة موثقة، مستكشفاً الحالات الحدية التي يواجهها المستخدمون الحقيقيون (مدخلات فارغة، تفاعلات سريعة، مسارات غير موثقة) والتي لم يتوقعها أي مجموعة اختبارات مكتوبة يدوياً.

- **أوراكل الرؤية** — واجهة حرفياً ليرى LLM API ورؤية GoCV يجمع بين الرؤية الميكانيكية لـ المستخدم أثناء التشغيل، ملتقطاً حالات الانهيار البصري التي تتجاوزها تأكيدات المستوى الرمزي والخاصيات.
- **بنوك الاختبارات الهيكلية لا النصية** — في وقت LLM تصف سلاسل البنوك الهيكل وتولد أسئلة لذا يعمل بنك واحد عبر اللغات بدلاً من الانهيار لحظة ترجمة نص، (CONST-046) التشغيل واجهة المستخدم.
- **تقارير مصممة لخطوط إصلاح AI** — تصل تقارير ماركداون المولدة تلقائياً مع حزمة الأدلة الكاملة مرفقة، جاهزة للتسليم مباشرة لوكيل الإصلاح بدلاً من مسؤول الفرز البشري.

كيفية استخدامه عبر جميع المنتجات (القوى التي يمنحها)

الفرعية `helix_qa` إلى وحدة Constitution §11.4.169 يشير) **ركيزة جودة إلزامية** باعتباره كل منتج في العائلة مجموعة القدرات ذاتها HelixQA كأحد أنواع الاختبارات المطلوبة)، يمنح:

- **جلسات ضمان جودة مستقلة:** `helixqa autonomous --project … --platforms android,desktop,web` تطلق أمر واحد وكيل LLM المدعوم بالرؤية ليقود التطبيقات الحقيقية دون إشراف نحو هدف تغطية محدد، وينتج تقارير وتذاكر وفيديوهات دون تدخل بشري.
- **بنوك/مجموعات الاختبارات:** بنوك YAML (≤30 219 الحد الأدنى للطابق)، مستهدفة للمنصات، مرتبة حسب الأولوية، وقابلة للتتبع سطراً بسطر إلى الوثائق التي تتحقق منها.
- **الأدلة الملتقطة:** لقطات الشاشة، سجلات النظام، الفيديوهات، تتبع المكدس، والجدول الزمني الكامل — مركزية ومُرتبطة بكل تقرير، بحيث يمكن إعادة تشغيل أي حكم وتدقيقه لاحقاً.
- **الأحكام المستقلة (§11.4.141 مبدأ الاستقلال):** يحكم `issuedetector` المدعوم بـ LLM وأوراكل الرؤية على سلوك التطبيق قيد التشغيل بشكل مستقل عن الوكيل الذي قام بالتنقل فيه، مما يستبعد بشكل هيكلي الفشل الكلاسيكي الذي يحدث عندما يصنف النظام عمله بأنه صحيح.
- **البوابة ومفتاح الطفرات:** تضمن أوامر `make qa-all` / `make anti-bluff` و`challenges/scripts/helixqa_orchestrator_challenge.sh` (8 مراحل، بشكل مستمر — ولا يوجد عمداً منفذ HelixQA طفرة مدمجة §1.1) إثبات أمانة `--skip-helixqa` لإلغاء هذا الانضباط تحت ضغط المواعيد النهائية.

المحتوى

أكبر التحديات التقنية وكيفية التغلب عليها

- **منع الإيجابيات الكاذبة في ضمان الجودة نفسه** — الأداة التي تكشف التلاعب يجب ألا تصبح تلاعباً بحد ذاتها ← كل خطوة يتم التحقق منها مقابل الأدلة الملتقطة، ويُحتسب النجاح بدون دليل كخلل بدلاً من نجاح، كما يربط بيان السلوك المرتبط بكل قدرة معلن عنها باختبار قابل للتنفيذ (CONST-035) بحيث لا يمكن الادعاء بأي قدرة دون وجود شيء يمارسها.
- **تشغيل منصات غير متجانسة من عقل واحد** — أندرويد، أندرويد تي في، الويب، وسطح المكتب واحدة تعمل كطبقة تجريدية فوق `navigator` لا يشتركون في نموذج إدخال موحد ← حزمة منفذي الإجراءات الخاصين بكل منصة (ADB، Playwright، X11) وكاشفات الأعطال لكل منصة (أندرويد/ويب/سطح المكتب)، بحيث يُكتب منطق التنسيق مرة واحدة وتبقى الاختلافات بين المنصات على الأطراف.
- **جعل الوكلاء المستقلين مفيدين وليس فوضويين** — غير الخاضع للإشراف LLM يمكن لوكيل بتقييم واختيار النماذج المناسبة LLMsVerifier، أن يتجول في التطبيق إلى الأبد ← يقوم الوكلاء غير المرئيين CLI (opencode، claude-code، gemini، junie، qwen-code)، ويدير LLMOrchestrator خريطة الميزات ويبني DocProcessor على أن كل قرار يستند إلى البكسلات التي تمنح الاستكشاف هدفاً، ويحافظ VisionEngine الحقيقية على الشاشة بدلاً من تخيلات النموذج.

- **بنوك آمنة للتوطين** — مجموعة اختبارات تعتمد على نصوص واجهة المستخدم الإنجليزية تتعطل في خمس عشرة لغة ← تصف البنوك الهيكل فقط، ويُحمل نص المطالبة المعروض للمستخدم بحيث يتحقق البنك نفسه من السلوك، (CONST-046) الموارد أثناء التشغيل/LLM بواسطة ذاته بغض النظر عن اللغة.
- **إثبات أن البوابات ليست خداعاً** — البوابة المضادة للتلاعب التي لا يمكن أن تفشل هي الخداع الأسمى ← تُزيل الطفرات المزدوجة في الفقرة 1.1 التقاط الأدلة أو تأكيدات مكافحة التلاعب وتتطلب من البوابة أن تفشل، كما يمنع مقياس الطفرات هذا الضمان من التآكل بصمت مع مرور الوقت.

مجموعة الأدوات التقنية

- **منسق Go 1.24 +الإصدار** — السبب: يجب أن يعمل ضمان الجودة في أي مكان تعمل فيه المنتجات، لذا فإن ثنائي واحد مترابط بشكل ثابت وسريع وقابل للنقل يتفوق على البدائل الثقيلة الذي يعرض أوامر فرعية قابلة للتكوين مثل `cmd/helixqa` CLI أمر: الكيفية وقت التشغيل؛ `version`و `autonomous`و `report`و `list`و `run`.
- **بنوك الاختبار YAML (`pkg/testbank`)** — السبب: يجب أن تكون مجموعات الاختبارات وصفية وقابلة للقراءة، ويمكن تحريرها بواسطة البشر دون الحاجة إلى لمس Go؛ الكيفية: `platforms`و `priority`و `category`و `id` مع `version/name/test_cases[]` و `documentation_refs[]` مرتبة و `steps[]`و لتتبع العودة إلى وثائق الميزات.
- **كاشفات الأعطال/ANR (`pkg/detector`)** — السبب: أهم حالات الفشل هي تلك التي تحدث أثناء التفاعل المباشر، وليس في تأكيد لاحق؛ الكيفية: ADB (`pidof/logcat/screencap`) للويب وسطح المكتب، لمراقبة العملية أثناء تشغيل الاختبار لها `pgrep`لأندرويد و (`screencap`).
- **جمع الأدلة (`pkg/evidence، pkg/session`)** — السبب: عقد مكافحة التلاعب لا يكون لقطات الشاشة، سجلات النظام، مقاطع الفيديو، وتتبعات المكدس التي تُلتقط في جدول زمني الذي ترتبط به كل `SessionRecorder` التقارير. الكيفية: حقيقياً إلا إذا كان كل نجاح مدعوماً بأدلة مادية؛
- **الجلسة المستقلة (`pkg/autonomous، pkg/navigator، pkg/issuedetector`)** — السبب: لا يمكن توسيع نطاق ضمان الجودة اليدوي الشامل عبر أربع منصات، لذا يجب أن يقود `SessionCoordinator` منسق جلسة رباعي المراحل الذي يشمل LLM وكشف أخطاء (ADB/Playwright/X11) بالإضافة إلى منفذي الإجراءات العيوب البصرية وتجربة المستخدم وإمكانية الوصول والوظيفية. الكيفية: الاستكشاف نفسه بنفسه؛
- **الوحدات الفرعية الخارجية** — السبب: إعادة الاستخدام وفك الاقتران (CONST-051)، والأهم: الكيفية من ذلك فصل الملاح عن المحكم؛ LLMsVerifier (تقييم النماذج)، LLMOrchestrator (الوكلاء غير المرئيين CLI)، VisionEngine (GoCV + رؤية LLM)، DocProcessor (خريطة الميزات/التغطية)، كل منها مكون مستقل مملوك بشكل منفصل.
- **بوابات مكافحة التلاعب ومقياس الطفرات** — السبب: ملتزماً بالعهد HelixQA للحفاظ على `make anti-bluff` فحص: الكيفية الدقيق في الفقرة 1.1 الذي تفرضه على كل شيء آخر؛ بالإضافة إلى بيان السلوك المرتبط ومقياس الطفرات، مع `helixqa_orchestrator_challenge.sh` كمحقق شامل من ثماني مراحل.
- **مصفوفة تغطية من 15 صفاً (`docs/test-coverage.md`)** — السبب: يلزم الكيفية: مجموعة مغلقة ومحسوبة بالكامل لأنواع الاختبارات دون فجوات؛ CONST-050(B) كل صف مرتبط بأصل تنفيذي ملموس وشكل محدد للأدلة الملتقطة، بحيث تكون التغطية حقيقة مؤكدة وليست مجرد ادعاء.

ملاحظات حول الحالة والنزاهة

- **الحالة: تجريبية.** شعار حالة ملف) قيد التطوير النشط README ٢١٩ (النسخة. تخضع لمعايير مكافحة التضليل الخاصة بها.
- **الرخصة: Apache-2.0.** التثبيت: `go install digital.vasic.helixqa/cmd/helixqa@latest`.

طبقة الأولوية: أساسية — ركيزة إلزامية لضمان الجودة ومكافحة التضليل في كيفية التحقق من Helix-تعمل بالفعل Helix أن مزايا عائلة.