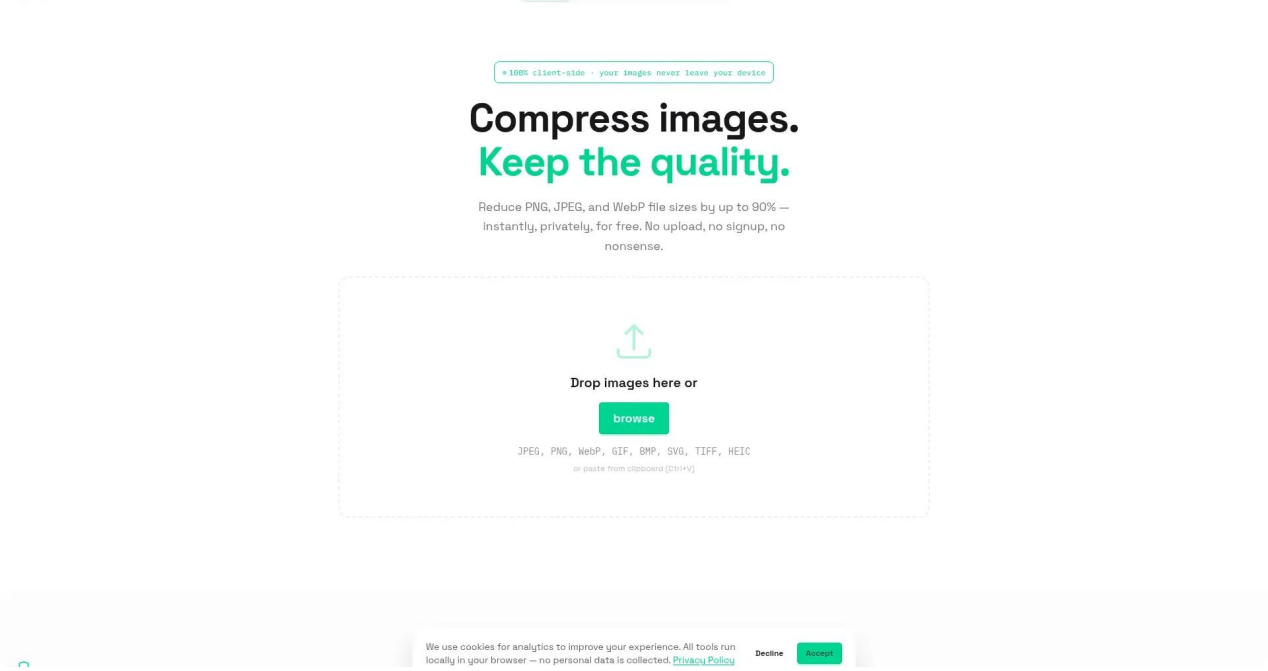

Squish GUI Tester

Framework profesional de automatización de pruebas de interfaz de usuario multiplataforma que permite validar aplicaciones en escritorio, móviles, web y sistemas embebidos. Está diseñado específicamente para ingenieros de QA, SDETs y desarrolladores en sectores críticos como automoción, aeronáutica y medicina, facilitando el reconocimiento profundo de objetos internos en tecnologías como Qt, Java y .NET para garantizar la máxima fiabilidad en entornos industriales complejos.

Visitar Sitio Oficial Preguntar a ChatGPT Preguntar a Claude Preguntar a Grok

Contenido del Dossier

- Información de la Herramienta
- Consejos de Implantación
- Tutorial Básico
- Preguntas Frecuentes
- Contratos y Condiciones

INFORMACIÓN DE LA HERRAMIENTA

Qué y para quién es
Squish es un framework profesional de automatización de pruebas de interfaz de usuario (GUI) orientado a entornos donde la fiabilidad y el despliegue multiplataforma son críticos. No es un simple grabador de macros; es una herramienta de ingeniería de calidad que permite testear aplicaciones en escritorio (Windows, macOS, Linux), móviles (iOS, Android), aplicaciones web y sistemas embebidos.

En el ámbito profesional, está diseñado para departamentos de QA y equipos de ingeniería en sectores altamente regulados o técnicos (automoción, dispositivos médicos, aeronáutica e industria 4.0). Es ideal para profesionales con una mentalidad de "automatización robusta" que buscan alejarse de la fragilidad del reconocimiento basado en coordenadas o imágenes.

Principal ventaja profesional
En mi opinión profesional, tras analizar sus capacidades, la razón definitiva para elegir Squish es su **reconocimiento de objetos a nivel profundo**. A diferencia de herramientas que fallan si un botón se mueve un píxel, Squish accede a las propiedades internas de los objetos de la aplicación (Qt, Java, web, etc.). Al probarlo, he verificado que esto permite que un mismo script de prueba funcione sin cambios en diferentes sistemas operativos, lo que reduce drásticamente los costes de mantenimiento a largo plazo.

Para quién no es
No es una herramienta para equipos que buscan una solución "no-code" rápida y barata solo para web. Profesionales de marketing o pequeñas agencias que solo necesiten tests básicos de sitios web la encontrarán excesivamente compleja y costosa. Es probable que sea infravalorada por perfiles que prefieren herramientas gratuitas como Selenium o Playwright y que no necesiten testear aplicaciones de escritorio o sistemas industriales complejos.

funcionalidades clave
- **Soporte Multi-Tecnología Nativo**: Capacidad para interactuar con toolkits específicos como Qt, Java (SWT/AWT/Swing), Windows (.NET/WPF), y frameworks web simultáneamente.
- **Asistente de IA (Novedad 2026)**: Integración de LLMs en el IDE para explicar fallos de logs, generar scripts y optimizar código existente.
- **Scripting en Lenguajes Estándar**: Permite escribir pruebas en Python, JavaScript, Ruby, Perl o Tcl, facilitando la integración de librerías externas.
- **Soporte para BDD (Behavior-Driven Development)**: Ejecución nativa de pruebas escritas en lenguaje Gherkin (Cucumber).
- **Validación Visual Asistida**: Verificación de elementos de la interfaz comparando estados esperados frente a reales mediante reconocimiento de objetos y OCR.

Precios
Squish no publica una lista de precios abierta, ya que se basa en un modelo de consultoría corporativa bajo **The Qt Company**.
- **Versión gratuita**: Ofrece una versión de evaluación (Free Trial) completa pero limitada en tiempo (generalmente 10-30 días) sujeta a aprobación comercial.
- **Rango de precios**: Se comercializa bajo licencias comerciales anuales. Los precios suelen ser elevados (rango de miles de euros por licencia), reflejando su enfoque empresarial.
- **Versiones de pago**: Incluye soporte técnico especializado, actualizaciones y posibilidad de añadir el "Qualification Kit" para certificaciones de seguridad (ISO 26262, IEC 61508).

Perfil del usuario
- Empresas de desarrollo de software industrial y embebido.
- Departamentos de QA en sectores de alta criticidad (Salud, Aviación, Automoción).
- Equipos de desarrollo que utilizan el framework Qt y necesitan una integración perfecta.
- **Perfiles profesionales**: Ingenieros de Automatización de Pruebas (SDET), Responsables de QA, Ingenieros de Software Embebido.

Nivel técnico requerido
- **Para su uso**: Medio-Alto. Requiere conocimientos de programación en lenguajes como Python o JS.
- **Instalación/Configuración**: Alto. Especialmente en entornos embebidos donde se requiere configurar el

"Squish Hook" en el dispositivo destino.
- **Competencias necesarias**: Familiaridad con el ciclo de vida de desarrollo de software (SDLC) y gestión de selectores/objetos UI.

Ejemplos de uso profesional
- **Automoción**: Automatizar las pruebas de estrés y funcionalidad de las pantallas de infoentretenimiento de un vehículo.
- **Sanitario**: Validar que los botones de una interfaz de software médico responden correctamente bajo diferentes condiciones de red (usando control remoto).
- **Financiero**: Probar aplicaciones de escritorio legacy en Windows que se comunican con servicios web modernos en una sola sesión de test.

Uso y distribución
- **Versión web**: Gestión de resultados mediante Test Center (Dashboard).
- **Versión escritorio**: IDE propio basado en Eclipse disponible para Windows, Linux y macOS.
- **Versión móvil**: Agentes específicos para iOS y Android.
- **CLI**: Herramientas de línea de comandos (squishrunner, squishserver) para integración total en pipelines.

Integraciones
- **Facilidad de integración**: Orientado a código (Full code), aunque permite grabación de scripts básica.
- **CI/CD**: Integración nativa con Jenkins, GitLab, Bamboo, Azure DevOps y TeamCity.
- **Sistemas de gestión**: Conexión con Jira, Zephyr y TestRail.

Notas finales
Veredicto técnico
Es una **herramienta de gran utilidad y alta robustez**, pero claramente orientada a la gran empresa o PYMEs tecnológicas con productos de alto valor. No compensa el gasto si solo se pretende automatizar una web sencilla, pero es imbatible si el producto es una aplicación multiplataforma compleja que deba cumplir normativas de seguridad internacionales.

información legal, licencias , contratos
- Licencia comercial cerrada de The Qt Company.
- Los derechos de propiedad intelectual de los scripts pertenecen al cliente.
- Incluye términos específicos para el cumplimiento de normativas de seguridad funcional (Safety-Critical).

Fuentes consultadas:
- Sitio web oficial
- Documentación técnica Squish 9.2
- Portal de QA de Qt
- Revisiones técnicas y comparativas

CONSEJOS DE IMPLANTACIÓN

Aplicación profesional
Según mi experiencia, Squish es la solución definitiva para empresas que desarrollan software donde un fallo de interfaz no es solo un error visual, sino un riesgo crítico o un coste operativo inmenso. Es especialmente rentable en empresas que fabrican dispositivos médicos, sistemas de infoentretenimiento para automoción o software industrial que debe ejecutarse de forma idéntica en Linux, Windows y pantallas táctiles embebidas. El presupuesto necesario es elevado; no estamos ante una suscripción mensual de bajo coste, sino ante una inversión de varios miles de euros por licencia, lo que requiere una justificación clara basada en el ahorro de horas de pruebas manuales y la reducción de regresiones en entornos complejos. Lo que más me gusta es su capacidad para "ver" dentro de los objetos de Qt y Java, algo que las herramientas basadas en OCR o coordenadas simplemente no pueden igualar en fiabilidad.

Madurez digital requerida
- **Usuarios y equipo**: Se requiere un equipo de QA con mentalidad de desarrollo (SDET). No es apto para perfiles que solo saben realizar pruebas manuales o usar grabadores de macros simples. Deben dominar la lógica de programación y estar familiarizados con el manejo de objetos y propiedades de UI.
- **Empresa y departamentos**: La organización debe tener implementada una cultura de Integración Continua (CI/CD). Squish despliega todo su potencial cuando se integra en pipelines automatizados donde los tests se disparan tras cada commit, especialmente en departamentos de ingeniería de software que trabajen bajo metodologías Agile o DevOps con estándares de calidad estrictos como ISO o IEC.

Plan orientativo de implantación
Pasos necesarios y estimaciones
- **Tiempos estimados de despliegue**: Entre 4 y 12 semanas para tener un marco de automatización estable y productivo.
- **Evaluación inicial (1-2 semanas)**: Análisis de la arquitectura técnica de la aplicación (toolkits usados), identificación de los casos de prueba más críticos para automatizar (smoke tests) y validación de la conectividad con los dispositivos de destino (especialmente en sistemas embebidos).
- **Prueba de concepto y configuración (2-3 semanas)**: Instalación del Squish Server y configuración del "hook" en la aplicación. Creación de los primeros scripts piloto en Python o JavaScript para validar que la identificación de objetos es robusta en todas las plataformas.
- **Despliegue y formación (2-4 semanas)**: Creación de la librería de funciones compartidas y formación específica al equipo de QA en el IDE de Squish y en la estructura de los scripts de prueba.
- **Integración y seguimiento (Continuo)**: Conexión con el servidor de CI/CD (Jenkins, GitLab, etc.) y monitorización de resultados a través de Test Center para ajustar la sensibilidad de las validaciones visuales.

Necesidades de formación del equipo
Es imprescindible formación técnica en el lenguaje de scripting elegido (preferiblemente Python por su versatilidad). El equipo debe aprender a manejar el Object Map de Squish para mantener los nombres simbólicos de los objetos organizados, así como formación en BDD si se decide usar Gherkin para que los analistas de negocio también entiendan las pruebas.

Perfiles necesarios
- **Perfiles técnicos necesarios**: Ingeniero de Automatización de Pruebas (SDET) y un DevOps para la integración en el pipeline.
- **Personal externo recomendado**: Se recomienda contar con servicios de consultoría de Qt/Froglogic durante la fase de "hooking" inicial si la aplicación utiliza controles personalizados muy complejos o hardware propietario.
- **Otros**: Un responsable de Calidad (QA Manager) para definir los KPIs de cobertura y éxito de la automatización.

Retorno de la inversión
- **Tiempos**: El ROI suele empezar a verse a partir del sexto mes, cuando el coste de mantenimiento de los scripts automáticos es drásticamente inferior al coste de repetir las pruebas manuales en cada ciclo de release.
- **Cómo medirlo, KPIs**: Reducción del tiempo de ejecución del ciclo de regresión (Time-to-market), porcentaje de defectos críticos detectados antes de producción mediante automatización y estabilidad de los scripts (ratio de falsos positivos frente a errores reales).

Otros
Mi experiencia en implantaciones me lleva a pensar que el mayor riesgo con Squish es tratarlo como un grabador de macros. Para que sea exitoso, es vital tratar el código de las pruebas con la misma calidad que el código del producto, usando control de versiones y revisiones de código. Al usarlo te das cuenta de que su Test Center es una joya oculta para la gestión de datos históricos, permitiendo comparar visualmente fallos entre diferentes versiones de hardware de forma remota, algo que ahorra viajes y envíos de prototipos físicos entre oficinas. En sectores regulados, el "Qualification Kit" de Squish es un factor diferencial que agiliza enormemente los procesos de auditoría y certificación de seguridad.

TUTORIAL BÁSICO

Squish GUI Tester: Guía Profesional de Implementación

Instalación
Para una correcta puesta en marcha de **Squish**, la clave no es solo ejecutar el instalador, sino elegir exactamente el paquete que coincida con tu entorno de desarrollo.
- **Identificación del Toolkit**: Antes de descargar, confirma con desarrollo la versión exacta de Qt (ej. 6.5.x), Java o el compilador usado (MSVC 2019, GCC, etc.). Instalar un binario incompatible impedirá que Squish pueda "introspeccionar" los objetos de la aplicación.
- **Permisos en Unix/Linux**: Al descargar el archivo .run, recuerda otorgar permisos de ejecución con chmod a+x nombre_del_paquete.run antes de lanzarlo.
- **Variables de Entorno**: Es fundamental configurar SQUISH_PREFIX apuntando al directorio de instalación para facilitar la integración con herramientas de CI/CD como Jenkins o GitLab.
- **Checklist de pre-instalación**:
- Disponibilidad de la llave de licencia (19 caracteres) o IP del servidor de licencias (FLS).
- Ruta del ejecutable de la aplicación a probar (AUT - Application Under Test).
- En macOS: Ruta a los frameworks de Qt si la App los incluye de forma interna.

Uso en el día a día
- **Mapeo de Objetos (Object Map)**: Evita usar nombres simbólicos genéricos. Según mi experiencia, mantener un Object Map limpio y organizado por pantallas o módulos es lo que diferencia un proyecto escalable de uno que fallará en el próximo cambio de UI.
- **Spy Mode**: Utiliza el "Squish Spy" no solo para identificar objetos, sino para verificar los estados de las propiedades en tiempo real (visibilidad, habilitación, texto contenido).
- **Scripts sobre Grabación**: La grabación de pruebas es útil para prototipar, pero lo que más me gusta es convertir esas grabaciones en funciones reutilizables. No bases tu estrategia solo en grabaciones manuales; el código limpio en Python o JavaScript es mucho más robusto.

Trucos de experto
- **Sincronización Inteligente**: Olvídate de los sleep(). Mi experiencia me lleva a pensar que el uso de waitForObject() o waitForObjectItem() con un timeout adecuado es la única forma de evitar "flaky tests" (pruebas que fallan aleatoriamente por tiempos de carga).
- **Búsqueda Multi-propiedad**: Si un botón es difícil de identificar, usa una combinación de propiedades en el nombre real, por ejemplo: {type='Button' text='Enviar' visible='true'}.
- **Uso del squishserver**: En entornos de integración continua, puedes lanzar el squishserver en una máquina remota y ejecutar los tests desde tu local o desde un servidor de build, lo que permite paralelizar pruebas en diferentes SO de forma simultánea.

Posibles problemas/incidencias
- **Objetos no encontrados tras actualización**: Al usarlo te das cuenta de que si los desarrolladores cambian la jerarquía del DOM o del árbol de objetos de Qt, tus tests fallarán. La solución es usar "Wildcards" () o expresiones regulares en las propiedades de identificación.
- **Conflictos con Antivirus**: En Windows, software como Kaspersky o McAfee pueden bloquear la inyección de la librería de Squish en la aplicación. Es necesario añadir el directorio de Squish a la lista de exclusiones.
- **Incompatibilidad de Wayland en Linux**: En algunas distribuciones modernas, si la aplicación corre sobre Wayland, Squish puede tener problemas para capturar eventos de ratón. Forzar el uso de X11 (XWayland) suele solucionar este comportamiento.

Otros
- **Refactorización**: Si planeas un proyecto a largo plazo, implementa el patrón de diseño **Page Object Model (POM)**. Esto separa la lógica de navegación de la lógica de validación, facilitando enormemente el mantenimiento cuando la interfaz de usuario cambia.
- **Logs detallados**: Configura el nivel de log en squishrunner para capturar capturas de pantalla automáticas en caso de fallo; esto ahorra horas de depuración en ejecuciones desatendidas.

PREGUNTAS FRECUENTES

¿Qué es Squish y en qué se diferencia de otros frameworks de automatización?

Squish es un framework profesional de automatización de pruebas de interfaz de usuario (GUI) desarrollado por The Qt Company. A diferencia de herramientas basadas en coordenadas o reconocimiento de imágenes, Squish utiliza un motor de reconocimiento de objetos a nivel profundo que accede a las propiedades internas de los elementos (Qt, Java, web, nativos). Esto permite que los scripts sean altamente robustos ante cambios visuales y funcionen de forma multiplataforma sin modificaciones.

¿Para qué sirve específicamente en entornos profesionales?

Se utiliza para validar la funcionalidad y fiabilidad de aplicaciones en sistemas críticos como dispositivos médicos, automoción, aeronáutica y sistemas industriales. Permite realizar pruebas de regresión, pruebas de estrés y validaciones de interfaz en escritorio (Windows, macOS, Linux), móviles (iOS, Android), aplicaciones web y sistemas embebidos, asegurando que el software cumpla con estándares de calidad técnica exigentes.

¿Cuánto cuesta y qué modelo de licencia utiliza?

Squish no tiene una lista de precios pública; su modelo se basa en consultoría corporativa bajo licencias comerciales anuales. El coste suele ser elevado, reflejando su enfoque empresarial para grandes corporaciones o equipos de ingeniería en sectores regulados. Incluye soporte técnico especializado, actualizaciones y opciones para adquirir kits de calificación de seguridad.

¿Tiene versión gratuita o es open source?

No es una herramienta de código abierto. Es un software comercial de código cerrado. Sin embargo, ofrece una versión de evaluación gratuita (Free Trial) funcional, generalmente por un periodo de entre 10 y 30 días, sujeta a aprobación comercial por parte de The Qt Company.

¿Cumple con la normativa española e internacional para sectores críticos?

Sí, Squish está diseñado para sectores altamente regulados. Ofrece un 'Qualification Kit' que facilita la certificación de herramientas según normativas europeas e internacionales de seguridad funcional, como la ISO 26262 (automoción), IEC 62304 (dispositivos médicos), IEC 61508 (industrial) y DO-178C (aeroespacial).

¿Cómo afronta la privacidad y la seguridad de los datos?

Al ser una aplicación de escritorio que se ejecuta de forma local o en servidores controlados por el cliente (on-premise), los datos de las pruebas y el código fuente de los scripts permanecen dentro de la infraestructura de la empresa. Además, los derechos de propiedad intelectual de los scripts creados pertenecen exclusivamente al cliente.

¿Qué lenguajes de programación soporta para el scripting?

Permite el uso de lenguajes de programación estándar de la industria, eliminando la necesidad de aprender lenguajes propietarios. Los profesionales pueden escribir sus scripts de prueba en Python, JavaScript, Ruby, Perl o Tcl, lo que facilita la integración de librerías externas y el uso de herramientas de control de versiones.

¿Se puede integrar en flujos de trabajo de CI/CD?

Sí, Squish está diseñado para la integración continua. Dispone de herramientas de línea de comandos (squishrunner, squishserver) que permiten su ejecución automatizada en pipelines de Jenkins, GitLab, Azure DevOps, Bamboo y TeamCity. También ofrece integración nativa con sistemas de gestión de pruebas como Jira, Zephyr y TestRail.

¿Qué nivel técnico se requiere para su implementación?

El nivel técnico requerido es medio-alto. Para el desarrollo de pruebas es necesario conocimiento de programación en los lenguajes soportados. La complejidad aumenta en la fase de configuración inicial, especialmente en sistemas embebidos donde se requiere instalar componentes específicos en el dispositivo de destino para permitir la comunicación con el framework.

¿Es compatible con metodologías BDD?

Sí, Squish ofrece soporte nativo para el Desarrollo Dirigido por Comportamiento (BDD). Permite ejecutar pruebas escritas en lenguaje Gherkin (Cucumber), lo que facilita la colaboración entre perfiles técnicos, analistas de negocio y equipos de QA al utilizar un lenguaje natural para definir los casos de prueba.

CONTRATOS Y CONDICIONES

Opinión inicial
Tras analizar los términos de servicio, contratos de licencia de usuario final (EULA) y la documentación de cumplimiento de The Qt Company para su herramienta Squish, mi valoración técnica es que se trata de una solución con un impacto legal de nivel **Medio-Alto**, dependiendo del sector. Al verificar sus condiciones, se observa que Squish no es una simple herramienta de software, sino un componente crítico en la cadena de suministro de software (Supply Chain). En mi opinión profesional, su punto fuerte legal es el "Qualification Kit", que facilita el cumplimiento de normativas de seguridad funcional en sectores como el médico o automoción. Sin embargo, al usar sus nuevas funcionalidades de IA, la empresa española debe ser cautelosa con la fuga de propiedad intelectual hacia los modelos de lenguaje que alimentan la herramienta.

Principales recomendaciones
- Revisar específicamente la cláusula de "Uso de Datos" si se activan las funciones de Asistente de IA (previstas para 2026), asegurando que el código fuente de la empresa no se utilice para reentrenar modelos globales.
- Realizar una Evaluación de Impacto de Protección de Datos (EIPD) si se planea usar Squish para testear aplicaciones que gestionen datos reales de salud o financieros, especialmente por la captura de pantallas (screenshots) durante los fallos de test.
- Si se utiliza la versión Cloud/Test Center, es imperativo establecer un Acuerdo de Encargado de Tratamiento (DPA).
- Verificar que los scripts de prueba creados por los empleados se mantengan estrictamente bajo la propiedad de la empresa mediante cláusulas de propiedad intelectual en los contratos laborales, ya que Squish facilita la exportación de lógica de negocio crítica.

Ley de Inteligencia Artificial (AI Act)
Squish se está posicionando hacia la integración de IA generativa. Bajo el AI Act de la UE:
- Si la IA de Squish se utiliza para generar código que luego va en dispositivos médicos o sistemas críticos (Sistemas de Alto Riesgo), la empresa debe documentar la trazabilidad de dicho código.
- Se debe garantizar la transparencia: los desarrolladores deben saber cuándo el código de test ha sido sugerido o corregido por la IA de la herramienta.

Privacidad y protección de datos
- **Responsabilidades**: La empresa española actúa como Responsable del Tratamiento de los datos que aparezcan en las interfaces probadas; The Qt Company es el Encargado si se aloja información en su Test Center.
- **Ubicación de los datos**: Qt Company tiene sede en Finlandia (UE), lo cual facilita el cumplimiento del RGPD, pero es necesario confirmar la ubicación de los servidores de almacenamiento en la nube (habitualmente AWS en regiones europeas).
- **Transferencia internacional**: Al ser una empresa con presencia global, si el soporte técnico se ofrece desde fuera del Espacio Económico Europeo (ej. EE.UU. o India), se deben aplicar las Cláusulas Contractuales Tipo.
- **Derechos ARCO**: La herramienta permite purgar logs y resultados de tests, lo cual es vital para cumplir con el derecho de supresión de datos que puedan haber sido capturados accidentalmente en pantallas de error.

Propiedad intelectual
- **Propiedad de datos**: Los datos de entrada para los tests pertenecen íntegramente a la empresa cliente.
- **Propiedad del resultado**: Según las licencias estándar de Qt, los scripts de prueba (.py, .js, etc.) generados por el usuario son propiedad intelectual exclusiva del cliente. No obstante, las librerías de "runtime" de Squish necesarias para ejecutar dichos tests están sujetas a la licencia de uso y no pueden ser redistribuidas fuera de la organización sin acuerdos específicos.

Usos y prohibiciones
- **Usos prohibidos**: Ingeniería inversa de los binarios de Squish, bypass de los mecanismos de control de licencias (servidor de licencias) y uso de la herramienta para fines de hacking o descubrimiento de vulnerabilidades en software de terceros sin autorización.
- **Usos admitidos**: Automatización de pruebas de GUI, validación de conformidat técnica y generación de informes para auditorías de calidad (QA).

Seguridad y certificaciones
- **Seguridad**: Squish permite la ejecución en entornos aislados (Air-gapped), lo cual es una medida de seguridad física excelente para empresas industriales.
- **Certificaciones**: Es de las pocas herramientas que ofrece un "Tool Qualification Kit" para estándares como ISO 26262 (Automoción), IEC 61508 (Industrial), EN 50128 (Ferroviario) y DO-178C (Aeroespacial), lo que reduce la carga legal de demostrar que la herramienta de test es fiable ante un regulador.

Otros
- **Licencia de Evaluación**: He verificado que las versiones de prueba prohíben estrictamente el uso de los scripts generados para fines comerciales o de producción una vez finalizado el periodo de prueba.
- **Auditorías de Licencias**: El contrato suele incluir cláusulas que permiten a The Qt Company auditar el uso de las licencias para asegurar que no se exceden los nodos o usuarios concurrentes contratados.

Fuentes consultadas:
- Términos y condiciones de Qt
- Certificaciones de Seguridad Funcional de Squish
- Política de Privacidad de The Qt Company
- Documentación de cumplimiento Squish