



Simutrans Squirrel 3.2

Migración de la Script API, compatibilidad y referencia para desarrolladores

Producto	Simutrans Standard	Runtime de Squirrel	3.2 (SQUIRREL_VERSION_NUMBER 320)
Revisiones de integración	r12165 – r12168 (trunk)	Revisión de cierre	r12168
Estado de la documentación	Vigente en la revisión de integración	Versión del documento	1.0
Fecha	2026-08-16	Tipo de documento	Referencia para desarrolladores
Idioma	Traducción revisada — el inglés es la referencia técnica canónica		
Destinatarios	Autores de guiones · Autores de escenarios · Autores de AI · Desarrolladores de Simutrans		

Preparado a partir de la migración validada de Squirrel 3.2 en Simutrans Standard.

Nota. Esta es la traducción al español del documento en inglés. La versión en inglés es la referencia técnica canónica.

RESUMEN EJECUTIVO

Simutrans Standard integra ahora **Squirrel 3.2**. Las fuentes de Squirrel incluidas en el proyecto se han actualizado desde una instantánea de 2021 de la línea de desarrollo 3.1.x hasta la publicación **v3.2** del proyecto original.

Lo importante para quien escribe o mantiene guiones de Simutrans:

- Es una actualización del **runtime del lenguaje**, no un rediseño de la Script API de Simutrans.
- La migración se diseñó a propósito para **conservar la Script API pública**. La actualización del runtime no ha añadido, eliminado ni renombrado ninguna función registrada, y no ha cambiado ninguna lista de parámetros ni el comportamiento de los argumentos opcionales.
- **Los escenarios y los guiones de AI compatibles que ya existen no necesitan reescribirse.**
- Han cambiado varios **puntos internos de integración con la VM**: principalmente la llamada C++ de búsqueda de instancias, la creación de clausuras para vincular un entorno, y la función hash de las cadenas.
- Existe una **diferencia semántica observable** para los guiones: el orden en que se recorren las entradas de una tabla. Se documenta en la sección 6 y es lo único que conviene revisar en un guion existente.
- **Se probaron explícitamente las partidas guardadas antiguas** con estado de guion, y se cargan correctamente.
- El runtime trae dos comodidades nuevas: una forma en línea de vincular el entorno, y `table.map`.

Si mantiene un guion y solo va a leer una sección de este documento, lea la [sección 6](#).

Índice

1.	Resumen ejecutivo	1
2.	Alcance y terminología	3
3.	Qué ha cambiado	3
4.	Qué no ha cambiado	5
5.	Matriz de compatibilidad	5
6.	Orden de iteración de las tablas	6
7.	Guía de migración para autores de guiones	7
8.	Autores de escenarios	8
9.	Autores de AI	9
10.	Paksets	9
11.	Ejemplos	9
12.	Notas para desarrolladores de Simutrans: la capa de integración	11
13.	Estabilidad de la Script API	12
14.	Resumen de la validación	12
15.	Límites conocidos y notas	13
16.	Dónde encontrar cada cosa	14
17.	Disponibilidad y registro de cambios	15
18.	Historial del documento	15

2. Alcance y terminología

Este documento usa tres términos con precisión, y la distinción importa a lo largo de todo el texto:

Término	Significado
Squirrel	El <i>lenguaje de guiones integrado y su runtime</i> , un proyecto externo incorporado a Simutrans.
Script API	La interfaz registrada por Simutrans: las clases, los métodos y las constantes que Simutrans expone a los guiones, documentados en la referencia de la Script API.
Simutrans Standard	La línea principal de Simutrans, a diferencia de Simutrans Extended.

Por tanto, «Squirrel 3.2» describe el runtime. No describe una versión nueva de la Script API, que se versiona por separado y que este trabajo no modifica.

Procedencia del runtime

Las fuentes de Squirrel que acompañan a Simutrans son una copia incorporada del proyecto original alojado en [albertodemichelis/squirrel](https://github.com/albertodemichelis/squirrel).

	Commit	Descripción
Base incorporada anterior	<code>23a0620658714b996d20da3d4dd1a0dcf9b0bd98</code>	Una instantánea de la línea de desarrollo 3.1.x, con fecha 2021-09-16
Base nueva	<code>f92bc298784ceea459b12e2de33bdff672bfeb83</code>	Etiqueta de publicación <code>v3.2</code> del proyecto original, con fecha 2022-02-10

La instantánea anterior se tomó de la línea de desarrollo *posterior* a la publicación de la 3.1, así que no era ni una 3.1 exacta ni una 3.2. Describir la actualización simplemente como «de la 3.1 a la 3.2» sería inexacto, y por eso se citan los commits.

3. Qué ha cambiado

3.1 Resumen

Cambio	¿Visible para los guiones?	¿Visible para quien programa en C++?
Fuentes de Squirrel actualizadas a la v3.2 original	de forma indirecta	sí
La API de búsqueda de instancias gana un parámetro	no	sí

Cambio	¿Visible para los guiones?	¿Visible para quien programa en C++?
Sintaxis en línea de <code>bindenv</code> añadida al lenguaje	sí, como opción nueva	sí
<code>table.map</code> añadido a la biblioteca estándar	sí, como opción nueva	no
Cambia la función hash de las cadenas	sí, véase la sección 6	sí

3.2 API de búsqueda de instancias

Squirrel 3.2 cambió la firma de la función de la API en C que recupera el puntero de una instancia nativa desde la pila de la VM. Gana un quinto parámetro que decide si un desajuste de tipo lanza un error de Squirrel o se comunica al llamante como un simple fallo:

```
/* antes */
SQLRESULT sq_getinstanceup(HSQIRRELVM v, SQInteger idx, SQUserPointer *p, SQUserPointer typetag);

/* Squirrel 3.2 */
SQLRESULT sq_getinstanceup(HSQIRRELVM v, SQInteger idx, SQUserPointer *p, SQUserPointer typetag,
SQBool throwerror);
```

C

Todos los puntos de llamada de Simutrans se actualizaron pasando el valor que conserva su comportamiento anterior. **A los autores de guiones no les afecta:** es una firma de C++, no un punto de entrada de la Script API, y el comportamiento observable de las llamadas afectadas no cambia.

Puntos de llamada de Simutrans modificados: `src/simutrans/script/api/api_map_objects.cc`, `src/simutrans/script/api_class.h`, `src/squirrel/sq_extensions.cc`, además de los ficheros de la biblioteca estándar incorporada.

3.3 `bindenv` y la forma en línea del entorno

Vincular un entorno a una función es una característica del lenguaje Squirrel, no una clase ni un registro de la Script API de Simutrans.

El método `bindenv` que ya existía **no ha cambiado y sigue funcionando**. Squirrel 3.2 añade una forma en línea en la que el entorno se escribe entre corchetes, entre la palabra clave de la función y la lista de parámetros. Las dos formas aparecen en la [sección 11](#).

La forma en línea es una comodidad. No da acceso a nada que la forma con método no permitiera ya, y **los guiones existentes no necesitan ningún cambio**.

Internamente, esto obligó a que la instrucción de creación de clausuras transportase el objetivo vinculado, así que se actualizaron tanto el compilador como la VM.

3.4 `table.map`

Squirrel 3.2 añade un método `map` a las tablas en la biblioteca estándar. Los arrays ya lo tenían.

Conviene conocer dos propiedades antes de usarlo:

- `table.map` devuelve un **array** con los resultados del callback, no una tabla. En esto se diferencia de `table.filter`, que devuelve una tabla.

- Los resultados aparecen en el orden de iteración de la tabla, así que el orden del array devuelto está sujeto a la [sección 6](#). Conviene ordenarlo si el orden importa.

3.5 Función hash de las cadenas

El proyecto original cambió la función hash que se aplica a las claves de tipo cadena. Esta es la causa de la única diferencia semántica visible para los guiones en esta migración, y se trata en detalle en la [sección 6](#).

4. Qué no ha cambiado

Esta sección es explícita a propósito, porque es la parte que más necesitan la mayoría de autores de guiones.

- **La Script API de Simutrans no ha cambiado.** No ha cambiado ningún nombre de clase. No ha cambiado ningún método público. No ha cambiado ningún nombre de registro. No ha cambiado ninguna lista de parámetros ni el comportamiento de los argumentos opcionales. Esto se midió, no se supuso: véase la [sección 13](#).
- **Los puntos de entrada de escenarios y de AI no han cambiado.**
- **El formato de las partidas guardadas no ha cambiado.**
- **La forma en que los guiones se cargan, se suspenden y se reanudan no ha cambiado.**
- **No hace falta ningún cambio en los paksets:** véase la [sección 10](#).

5. Matriz de compatibilidad

Área	Estado de compatibilidad	Acción del desarrollador
Nombres de la Script API existentes	Sin cambios. El número de registros y las convenciones de llamada se midieron idénticos antes y después.	Ninguna.
Guiones de escenario	Compatibles , con la salvedad del orden de iteración.	Probar con una compilación actual; revisar cualquier recorrido de tabla que tome una decisión.
Guiones de AI	Compatibles. Los dos jugadores de AI que acompañan a Simutrans se ejecutaron con el runtime nuevo y siguen siendo compatibles.	Igual que los escenarios. Cabe esperar que las decisiones de planificación y de rutas varíen en el detalle: véase la sección 9.
Partidas guardadas con estado de guion	Las partidas antiguas se cargan correctamente , comprobado con los jugadores de AI incluidos. También se probó guardar y recargar con 3.2.	Ninguna. Haga copia de seguridad antes de una campaña de pruebas amplia, como siempre.
Sentido inverso: partida de 3.2 en una versión anterior	No se afirma. No formó parte de este trabajo ni se probó.	No confiar en ello.

Área	Estado de compatibilidad	Acción del desarrollador
Vínculos con la VM (capa C++ de integración)	Han cambiado. La búsqueda de instancias gana un parámetro; la creación de clausuras transporta un objetivo vinculado.	Solo afecta al código que toca <code>src/squirrel/</code> o la capa de registro de la Script API. Véase la sección 12.
Orden de iteración de las tablas	Ha cambiado. Función hash distinta para las cadenas, y por tanto orden de <code>foreach</code> distinto.	Revisar cualquier guion que dependa del orden que produzca una tabla. Véase la sección 6.
Orden de iteración de los arrays	Sin cambios. Los arrays son contenedores ordenados; su orden forma parte de su contrato.	Ninguna.
Diagnósticos de error	No se observaron errores de Squirrel nuevos en la campaña de validación.	Ninguna.
Contenido de los paksets (<code>.dat</code> , objetos, gráficos)	No afectado.	Ninguna.

Nota sobre el alcance. Las afirmaciones de compatibilidad anteriores describen los guiones y las rutas de código que cubrió la campaña de validación: los jugadores de AI que acompañan a Simutrans y los sentidos de carga de partidas indicados. No son una garantía de que cualquier guion de terceros quede inalterado.

6. Orden de iteración de las tablas

ATENCIÓN

No se debe depender del orden de iteración de las tablas de Squirrel, salvo que la API lo documente explícitamente.

Squirrel 3.2 cambia la función hash que se aplica a las claves de tipo cadena. Las tablas son contenedores hash, de modo que esto cambia el orden en que `foreach` recorre sus entradas.

6.1 Qué significa esto en la práctica

La consecuencia para un guion es indirecta, pero real. Si un guion recorre una tabla para elegir la siguiente tarea, el orden nuevo puede cambiar:

- qué entrada se examina primero,
- el orden en que se planifica el trabajo,
- qué candidato gana cuando varios son igual de buenos,
- y, por tanto, a cuál de varios resultados igual de válidos llega el guion.

6.2 Qué no significa

Esto no es corrupción ni es aleatoriedad. Con la misma entrada, el runtime se comporta igual siempre. El guion conserva su estado lógico completo y la ejecución sigue siendo válida.

Lo que cambia es un desempate basado en un orden que nunca estuvo garantizado.

6.3 Qué hacer al respecto

Si un guion necesita un orden definido, tiene que imponerlo él mismo en lugar de confiar en el orden que una tabla produzca por casualidad. Recoja las claves y ordénelas:

```
// Frágil: el orden depende de la función hash del runtime.
foreach (key, value in prices) {
    consider(key, value)
}

// Definido: el orden lo elige el guion.
local keys = prices.keys()
keys.sort()
foreach (key in keys) {
    consider(key, prices[key])
}
```

SQUIRREL

Tanto `table.keys()` como `array.sort()` forman parte de los delegates estándar de Squirrel disponibles en Simutrans.

Los arrays no se ven afectados: son contenedores ordenados y su orden forma parte de su contrato. **Prefiera los arrays cuando el orden tenga significado.**

6.4 Estado de guion guardado

Cuando el estado persistente de un guion se vuelve a escribir, las entradas de una tabla guardada pueden aparecer en un orden distinto al anterior. El estado guardado en sí no se ve afectado: se restaura **por nombre, no por posición**.

7. Guía de migración para autores de guiones

7.1 Si su guion ya funciona

No hay ningún paso de migración. En concreto:

1. **Pruébalo con una compilación actual.** Ejecute el escenario o la AI como lo haría normalmente.
2. **Revise los avisos y errores de ejecución.** La campaña de validación no produjo errores de Squirrel nuevos; si su guion produce uno, merece la pena informar de ello.
3. **Revise el código que recorre una tabla para tomar una decisión.** Si gana la primera coincidencia, o la última, determine si ese comportamiento era intencionado y hágalo explícito.
4. **No reescriba nada que no esté roto.** No ha cambiado ninguna API.

Una advertencia sobre el método de prueba: **no compare dos ejecuciones por sus registros.** Dos ejecuciones del mismo guion con versiones distintas de Simutrans pueden tomar decisiones distintas e igualmente

válidas. Un diff de registros mostrará ruido que no es un defecto.

7.2 Si su guion falla

Siga esta secuencia en orden. Está pensada para separar las causas con rapidez.

1. **Capture el error real de Squirrel.** El mensaje y la línea indicada son el punto de partida; no diagnostique solo por los síntomas.
2. **Compruebe el uso de la Script API con la referencia.** La API no cambió en esta migración, así que un error de API suele significar que el guion ya usaba algo que había cambiado en otro momento, o que nunca fue correcto. Consulte el [registro de cambios de la Script API](#) de la versión a la que apuntaba.
3. **Compruebe las suposiciones sobre la iteración.** Busque un `foreach` sobre una tabla en el que el resultado dependa de qué entrada viene primero. Esta es la única categoría a la que esta migración puede haber afectado de forma plausible.
4. **Compruebe si hereda comportamiento del runtime anterior.** Si el guion dependía de una peculiaridad de la instantánea anterior y no del comportamiento documentado de Squirrel, esa es la causa más probable que queda.
5. **Separe los problemas de la Script API de las suposiciones sobre el pakset y el escenario.** Un objeto que falta, una vía renombrada o una diferencia de pakset producen fallos que parecen fallos de guion y no lo son. Confirme cuál de los dos tiene antes de cambiar código.

NOTA

Si el fallo sobrevive a los cinco pasos y es reproducible, merece la pena informar en el foro con el texto del error, la revisión de Simutrans y un guion mínimo que lo reproduzca.

8. Autores de escenarios

Compatibilidad esperada. Los escenarios no necesitan cambios. Los puntos de entrada de escenario, la Script API y el formato de las partidas guardadas no han cambiado.

Validación recomendada.

- Cargue el escenario con una compilación actual y juegue lo suficiente para ejercitar sus reglas y sus condiciones de victoria y derrota.
- Guarde y recargue a mitad del escenario, y confirme que el estado persistente está intacto.
- Revise cualquier regla que recorra una tabla de fábricas, paradas, ciudades o jugadores para elegir un objetivo o para conceder algo. Si dos candidatos pudieran ser igual de buenos, ahora puede ganar el otro.

Salvedades semánticas.

- El cambio de orden de iteración afecta a *cuál* de varios resultados igual de válidos se produce, no a *si* el escenario funciona. Un escenario cuyo texto diga «la primera fábrica encontrada» quizá quiera decir algo más preciso.
- Si el resultado esperado de un escenario se comprueba contra un valor fijo que salió de un recorrido de tabla, haga explícito ese orden como se muestra en la sección 6.3.

9. Autores de AI

Compatibilidad esperada. Los dos jugadores de AI que acompañan a Simutrans, `sqai` y `sqai_rail`, se ejecutaron con el runtime nuevo y siguen siendo compatibles. En la campaña de pruebas conservaron el estado lógico de sus conexiones, continuaron progresando de forma válida y no produjeron errores de guion ni de runtime. También se comprobó que se carga correctamente una partida guardada con el runtime anterior.

Validación recomendada.

- Ejecute una campaña de duración realista en lugar de una prueba rápida. Las diferencias de comportamiento de la AI se acumulan con el tiempo y una ejecución de dos minutos no las muestra.
- Confirme que el estado persistente de la AI sobrevive a un ciclo de guardar y cargar.
- Vigile una AI que **deje de progresar**, no una AI que progrese de otra manera. Lo segundo es lo esperado.

Salvedades semánticas.

- **Las decisiones de planificación y de búsqueda de rutas pueden variar.** Es la consecuencia esperada del cambio de orden de iteración, y no es un defecto.
- Una AI que recorra sus objetivos candidatos y tome el primero aceptable ahora puede tomar otro igualmente aceptable. Si una prioridad concreta importa, ordene los candidatos explícitamente.
- Por eso mismo, una prueba de AI que compruebe una secuencia exacta de conexiones construidas está comprobando la función hash del runtime, no la AI. Compruebe propiedades: que la red está conectada, que el saldo es positivo, que no se lanzaron errores.

10. Paksets

Esta migración no requiere ningún cambio en los paksets.

En concreto:

- Ninguna migración del formato binario de los objetos pak.
- Ninguna migración de `.dat`. No se ha añadido, eliminado ni reinterpretado ningún parámetro.
- Ninguna migración de gráficos.
- Ningún cambio de `menuconf.tab` ni de configuración.

Los guiones y el contenido de los paksets son cosas distintas. Un pakset que incluya escenarios solo se ve afectado a través de esos escenarios, exactamente como se describe en la sección 8, y solo en el sentido del orden de iteración: el contenido del pakset en sí no necesita nada.

11. Ejemplos

Todos los ejemplos siguientes son Squirrel válido y usan solo características disponibles en Simutrans en la revisión de integración.

11.1 Vinculación de entorno: las dos formas

La forma con método que ya existía no ha cambiado y sigue funcionando:

```
local env = { factor = 2 }
local scale = function(x) { return x * factor }
local bound = scale.bindenv(env) // 'this' inside scale is now env
bound(21)                        // 42
```

SQUIRREL

Squirrel 3.2 añade una forma en línea. El entorno se escribe entre corchetes, entre la palabra clave de la función y la lista de parámetros:

```
local env = { factor = 2 }
local scale = function [env] (x) { return x * factor }
scale(21)                        // 42
```

SQUIRREL

La misma forma con corchetes se admite en funciones con nombre, en funciones locales y en miembros de clases y tablas, incluido `constructor`. Las dos formas producen una clausura cuyo entorno es el objeto indicado, ya sea una tabla, una clase o una instancia.

Nota de portabilidad. La forma en línea es una *característica nueva del lenguaje*. Un guion que la use no funcionará en versiones anteriores de Simutrans. La forma con método funciona en todas.

11.2 `table.map`

```
local prices = { coal = 10, oil = 25 }
local labels = prices.map(function(key, value) {
    return key + "=" + value
})
```

SQUIRREL

El callback se llama una vez por entrada y recibe la clave y el valor, con `this` vinculado a la tabla. `labels` es un `array`, y el orden de sus elementos sigue el orden de iteración de la tabla: ordénelo si eso importa.

Nota de portabilidad. `table.map` es una *característica nueva de la biblioteca estándar*. Un guion que la use no funcionará en versiones anteriores de Simutrans.

11.3 Iteración con un orden definido

```
// Recoger las claves, imponer un orden y recorrerlas.
local keys = prices.keys()
keys.sort()
foreach (key in keys) {
    do_something(key, prices[key])
}
```

SQUIRREL

Conviene aplicar este patrón allí donde la decisión que se toma dentro del bucle sea visible en el resultado: el objetivo al que construye una AI, la fábrica que elige un escenario, la entrada que un informe lista primero.

12. Notas para desarrolladores de Simutrans: la capa de integración

Esta sección es para quien toque `src/squirrel/` o la capa de registro de la Script API. Es un resumen de lo que se movió, no un recorrido por el código fuente.

12.1 Búsqueda de instancias

`sq_getinstanceup()` gana un quinto parámetro, `SQBool throwerror`, que decide si un desajuste de la etiqueta de tipo lanza un error de Squirrel o devuelve un fallo simple al llamante.

Todos los puntos de llamada se actualizaron conservando su comportamiento anterior:

- `src/simutrans/script/api_class.h` y `src/simutrans/script/api/api_map_objects.cc` pasan `SQTrue`, con lo que se mantiene el contrato de Simutrans de que un tipo equivocado es un error de guion.
- `src/squirrel/sq_extensions.cc` pasa `SQTrue`.
- Los ficheros de la biblioteca estándar incorporada (`sqstdblob.cc`, `sqstdio.cc`, `sqstdstream.cc`, `sqstdstring.cc`) siguen al proyecto original, que usa `SQFalse` donde el fallo de la búsqueda se gestiona localmente.

Si añade una clase nueva a la Script API, siga `api_class.h`: pase `SQTrue` para que un desajuste de tipo llegue al autor del guion como un error de Squirrel y no como un nulo silencioso.

12.2 `bindenv` y la creación de clausuras

`SQVM::CLOSURE_OP()` gana un parámetro `boundtarget`, y el compilador lo emite para la forma en línea `function [env] (...)`. Nada de la capa de registro de la Script API depende de esto; solo importa si se modifica el compilador o la VM.

12.3 Hash de las cadenas

`_hashstr()`, en `src/squirrel/squirrel/sqstring.h`, se sustituyó por la versión de la 3.2 original. Recorre la cadena en sentido contrario y usa un cálculo del paso distinto. Esta es la causa del cambio de orden de iteración de las tablas descrito en la sección 6.

Consecuencia práctica para el trabajo sobre el motor: cualquier cosa que recorriera una tabla de Squirrel y dependiera del orden resultante — incluidas las expectativas de las pruebas — tiene que imponer su propio orden. Esto vale para el código C++ y para la suite de pruebas tanto como para los guiones.

12.4 Invariantes de registro

La migración se condicionó a que la Script API no derivase. Los invariantes que se mantuvieron constantes, y que conviene mantener en el trabajo futuro sobre esta capa:

Invariante	Valor	Ámbito
Sitios <code>register_function(vm</code>	79	<code>src/simutrans/script/</code>
Sitios <code>log_squirrel_type</code>	7	<code>src/simutrans/script/api/</code>
Deriva de registro	0	—
Deriva de convención de llamada	0	—

Invariante	Valor	Ámbito
Deriva de tipos expuestos públicamente	0	—

Los ámbitos se indican de forma explícita porque es fácil equivocarse: contar `register_function(vm` solo bajo `src/simutrans/script/api/` da 77, no 79. Las dos cifras son correctas para su propio ámbito; el invariante es únicamente la más amplia.

12.5 El principio de conservación

La regla que siguió esta migración, y que conviene conservar para cualquier actualización futura del runtime: **una actualización del runtime puede cambiar cómo se gobierna la VM, pero no debe cambiar lo que expone la Script API**. Si un cambio del runtime obliga a un cambio visible de la API, eso es un trabajo aparte, con su propia revisión.

13. Estabilidad de la Script API

La migración se validó buscando deriva no intencionada de la API pública, en lugar de darla por segura. Las comprobaciones se ejecutaron sobre un checkout limpio del trunk oficial en r12168.

Resultado: ninguna deriva de ningún tipo.

- Número de funciones registradas: **79**, sin cambios respecto al registro congelado previo a la migración (ámbito: `src/simutrans/script/`).
- Sitios de registro de tipos: **7**, sin cambios (ámbito: `src/simutrans/script/api/`).
- Deriva de registro: **0**.
- Deriva de convención de llamada: **0**.
- Deriva de tipos expuestos públicamente: **0**.
- Ficheros modificados por la revisión de documentación que contienen código: **0**.

Dicho en claro: todas las funciones que la Script API ofrecía antes de la migración se ofrecen después, con el mismo nombre, con los mismos argumentos y exponiendo los mismos tipos.

14. Resumen de la validación

Todas las cifras siguientes proceden de un **checkout limpio del trunk oficial en r12168**, compilado y ejecutado después de publicar los commits, no de la copia de trabajo desde la que se hicieron.

284 / 284

REGRESIÓN

10 / 10

PRUEBAS DE VM

2 / 2

PRUEBAS HUÉRFANAS

Comprobación	Resultado
Suite de regresión completa	284 / 284
Pruebas específicas de la VM	10 / 10 presentes y correctas
Pruebas huérfanas (las que desaparecerían en silencio si la suite retrocediera)	2 / 2 presentes y correctas
Errores de Squirrel nuevos	0
Deriva de la Script API	0 (véase la sección 13)
Compilación de la documentación	269 ficheros HTML, 0 diagnósticos nuevos respecto a la línea base aceptada

Validación adicional realizada durante la migración:

- Los dos jugadores de AI incluidos se ejecutaron en una campaña con el runtime nuevo.
- Una partida guardada con el runtime anterior se cargó varias veces, para confirmar que el resultado era reproducible y no una ejecución afortunada.
- Un estado cargado con 3.2 se volvió a guardar y se recargó correctamente.
- Los dos estados intermedios del runtime se comprobaron como entregables de forma independiente antes de publicar nada, así que ninguna revisión del rango r12165–r12168 deja el árbol en un estado roto.

La suite se comprueba por nombre y no solo por número: las pruebas de VM y las huérfanas se verifican **por su nombre**, de modo que una suite que hubiera retrocedido en silencio a una lista antigua se detectaría en lugar de aparecer como un total correcto.

15. Límites conocidos y notas

15.1 Orden de iteración de las tablas

Documentado en la [sección 6](#). Es la única diferencia de comportamiento que puede observar un guion existente, y es consecuencia de un cambio del proyecto original, no de una decisión de Simutrans.

15.2 Alcance de la afirmación de compatibilidad

La afirmación de compatibilidad cubre los jugadores de AI que acompañan a Simutrans y los sentidos de carga de partidas indicados en la sección 5. **No se han probado guiones de terceros.**

15.3 No se afirma el sentido inverso

Cargar una partida guardada con Squirrel 3.2 en una versión anterior de Simutrans no formó parte de este trabajo y no se afirma que funcione.

15.4 Las características nuevas no son compatibles hacia atrás

La forma en línea de `bindenv` y `table.map` son nuevas. Un guion que use cualquiera de las dos no funcionará en versiones anteriores de Simutrans. Si un guion debe funcionar en ambas, use `bindenv` como método y evite `table.map`.

15.5 Compilar la documentación de la Script API

Esta nota solo es relevante si compila usted mismo la referencia de la Script API con Doxygen.

`documentation/DoxyfileSQAPl.in` lleva `FILTER_PATTERNS = *.cc`, que las versiones recientes de Doxygen rechazan. Esto es **preexistente y ajeno al runtime de Squirrel 3.2**: ya estaba antes de esta migración y se dejó a propósito en lugar de arreglarlo dentro de un cambio que no venía al caso. Merece su propio parche pequeño.

Un segundo punto, de entorno, para quien compile en Windows: la compilación de la documentación debe invocar el filtro de entrada a través de un envoltorio de shell. Sin él, el Doxygen nativo de Windows termina con éxito mientras descarta en silencio toda la referencia de la API. Compruebe el número de ficheros generados, no el código de salida.

16. Dónde encontrar cada cosa

Este documento no duplica la referencia de la Script API. Úselo como mapa.

Si busca	Vaya a
Detalles de la API: clases, métodos, parámetros, constantes	La referencia de la Script API, generada desde <code>src/simutrans/script/api/</code> (<code>api_doc.h</code> es su página principal)
Qué ganó o cambió la API, y cuándo	<code>src/simutrans/script/api/changelog.h</code> → la página <i>Changelog</i>
Migración y compatibilidad de Squirrel 3.2	Este documento, y <code>src/simutrans/script/api/squirrel_32_en.h</code> → la página <i>Squirrel 3.2 runtime update</i> , que es su equivalente dentro del código
Lo mismo, en español	<code>src/simutrans/script/api/squirrel_32_es.h</code> → <i>Actualización del runtime de Squirrel 3.2</i>
Errores frecuentes al escribir guiones	La página <i>Common pitfalls</i> de <code>api_doc.h</code> (en inglés)
Funciones disponibles de la biblioteca estándar de Squirrel	La página <i>Available functions from the squirrel standard lib</i> de <code>api_doc.h</code> (en inglés)
Guiones de ejemplo	Las páginas de ejemplo de <code>api_doc.h</code> (en inglés)

La página del código en `squirrel_32_en.h` es la **referencia técnica canónica** de esta migración. Este documento es su presentación pública e independiente: añade las orientaciones por audiencia de las secciones 7 a 10, las cifras de validación de las secciones 13 y 14, y las notas de integración de la sección 12, y no omite nada sustancial de la página del código.

La página en español de `squirrel_32_es.h` es una **traducción revisada**; la página en inglés sigue siendo la canónica allí donde las dos pudieran llegar a leerse de forma distinta.

17. Disponibilidad y registro de cambios

Integrado en el trunk de Simutrans Standard: r12165 – r12168, siendo r12168 el estado de migración y documentación completado.

Disponible en las compilaciones que incluyan r12168 o posterior. Aquí no se afirma ninguna publicación estable numerada; consulte las notas de publicación de la compilación que utilice.

Registro de cambios

Revisión	Cambio
r12165	CODE: update Squirrel instance lookup API for 3.2 — adaptar todos los puntos de llamada a la firma nueva de <code>sq_getinstanceup()</code> , conservando el comportamiento anterior.
r12166	CODE: update Squirrel bindenv support for 3.2 — transportar el objetivo vinculado a través de la creación de clausuras para que la forma en línea del entorno compile y se ejecute.
r12167	CODE: reconcile remaining Squirrel 3.2 runtime changes — el hash de cadenas del proyecto original, <code>table.map</code> , y el resto de diferencias de la VM y de las cabeceras, además del fichero de licencia del runtime.
r12168	DOC: document Squirrel 3.2 runtime migration — la referencia bilingüe de migración dentro del código y sus enlaces desde la página principal y el registro de cambios.

18. Historial del documento

Versión	Fecha	Cambio
1.0	2026-08-16	Primera publicación. Preparado a partir de la migración validada en el trunk r12168.