Havok Script

Efficient Lua Scripting In Game Production

HavokTM Confidential. ©Copyright 2011 Havok.com (and its licensors). All Rights Reserved. Confidential Information of Havok.

SCRIPTING

HavokTM Confidential. ©Copyright 2011 Havok.com (and its licensors). All Rights Reserved. Confidential Information of Havok.

Scripting Languages

- No definitive computer science definition:
  - "a programming language that allows control of one or more applications" – Wikipedia.
  - E.g. a game!
- For game dev, usually means an **interpreted language**:
  - a language where programs are not compiled to machine code
  - Instead, its programs are compiled to **bytecode**
  - **interpreted** by a **virtual machine** running on an actual machine

havok

Havok™ Confidential. ©Copyright 2011 Havok.com (and its licensors). All Rights Reserved. Confidential Information of Havok.

Scripting provides

- Faster iteration
  - Don't have to recompile the game
  - Deploy and modify scripts while debugging
- Requires less programming experience
  - Designers can script
- Fewer dependencies
  - Puts a layer between the engine and the game logic
  - Isolation from platform

havok

HavokTM Confidential. ©Copyright 2011 Havok.com (and its licensors). All Rights Reserved. Confidential Information of Havok.

Scripting enables

- Updates and DLC
  - Without expensive and time consuming certification
  - Script “data” can contain:
    - Game logic updates, new challenges, UI updates, etc.
- Third-party or end-user content
  - Without security concerns – Sandboxing
  - Modding

HavokTM Confidential. ©Copyright 2011 Havok.com (and its licensors). All Rights Reserved. Confidential Information of Havok.

LUA

HavokTM Confidential. ©Copyright 2011 Havok.com (and its licensors). All Rights Reserved. Confidential Information of Havok.

Lua in Games

- Extremely popular scripting language for games
- Popularized by Grim Fandango in 1998
- Numerous games since
- Numerous Game Engines & Middleware use Lua

havok

HavokTM Confidential. ©Copyright 2011 Havok.com (and its licensors). All Rights Reserved. Confidential Information of Havok.

Why Lua (1)

- Fast
- Light-weight
- Stable
  - Lua 5.1: 2006
  - Lua 5.2: expected 2011
- Easy to learn

HavokTM Confidential. ©Copyright 2011 Havok.com (and its licensors). All Rights Reserved. Confidential Information of Havok.

Why Lua (2)

- Designed to be embedded
  - Easy-to-use C API.
- Dynamically create and execute code
  - Small compiler can fit on target machines.
- Helpful and active community
- Language features...

havok

HavokTM Confidential. ©Copyright 2011 Havok.com (and its licensors). All Rights Reserved. Confidential Information of Havok.

Efficient Lua Scripting in Game Production

LUA FEATURES AND PATTERNS

HavokTM Confidential. ©Copyright 2011 Havok.com (and its licensors). All Rights Reserved. Confidential Information of Havok.

Tables for arrays, structures and objects

- Lua has one really flexible data structure, the **table**.
  - t = {1, 2, 3, x = 2.0, f = function (self) return self.x * self.x end}

- An *array*: the 1,2,3 part

- A *structure*: t.x → 2.0

- An *object*: t:f() → 4.0

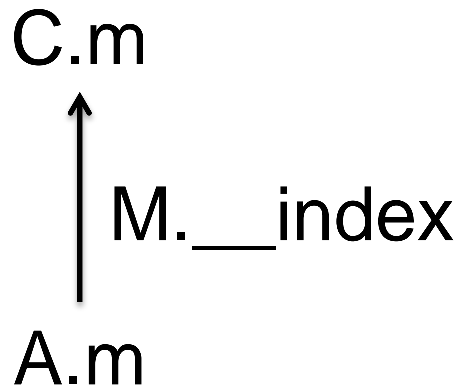
havok

HavokTM Confidential. ©Copyright 2011 Havok.com (and its licensors). All Rights Reserved. Confidential Information of Havok.

Metatables for class-based OO

- Access on a table can fall back to another table:

  - C = { m = function (self) print (“Hello from ”..self.name) end }
  - A = { name = “Bob” }
  - M = { __index = C }
  - setmetatable(A, M)
  - print(A:m())



HavokTM Confidential. ©Copyright 2011 Havok.com (and its licensors). All Rights Reserved. Confidential Information of Havok.

Environments for sandboxing

- Explicitly define a function to run in an environment where only print is defined:
  - setfenv(f, { print = print })

Havok™ Confidential. ©Copyright 2011 Havok.com (and its licensors). All Rights Reserved. Confidential Information of Havok.

Coroutines for state machines

```
function Person:asleepState()
	local count = 0
	while true do
		count = count + 1
		if count >= 1000 then
			return self.wanderState
		end
		self:snore()
		coroutine.yield()
	end
end
```

YIELD THE COROUTINE FROM INSIDE THE LOOP

HavokTM Confidential. ©Copyright 2011 Havok.com (and its licensors). All Rights Reserved. Confidential Information of Havok.

Tail calls for live update

- Tail-calls do not "use up" the stack:

```
function foo()
  print("Hello world")
  coroutine.yield()
  return foo()
end
```

- If we deploy a new definition of "**foo**" to the VM, it will be called on the next iteration

Havok™ Confidential. ©Copyright 2011 Havok.com (and its licensors). All Rights Reserved. Confidential Information of Havok.

DEBUGGING AND PROFILING LUA

Havok™ Confidential. ©Copyright 2011 Havok.com (and its licensors). All Rights Reserved. Confidential Information of Havok.

Debugging Lua

- Lua comes with some build-in introspection tools
  - The debug library
  - Not a user-friendly debugging experience
- In reality, most people debug Lua using print
  - ☹

havok

HavokTM Confidential. ©Copyright 2011 Havok.com (and its licensors). All Rights Reserved. Confidential Information of Havok.

Profiling Lua performance

- Easy to determine overall script cost
- Put timers around calls into the VM, and calls out of the VM
- What happens in between is typically a bit of a mystery

havok

Havok™ Confidential. ©Copyright 2011 Havok.com (and its licensors). All Rights Reserved. Confidential Information of Havok.

Profiling Lua memory

- Easy to determine how much memory Lua is using as a whole
- Track the requests Lua makes to its memory allocator
- What script code is causing the allocations is typically a bit of a mystery

havok

Havok™ Confidential. ©Copyright 2011 Havok.com (and its licensors). All Rights Reserved. Confidential Information of Havok.

HAVOK SCRIPT

Havok™ Confidential. ©Copyright 2011 Havok.com (and its licensors). All Rights Reserved. Confidential Information of Havok.

Havok Script (1)

- A drop-in replacement for Lua's VM
- Better performance on each supported platform:
  - PC, PS3, Xbox 360, Wii, iOs
- Language extensions
  - For performance and improved memory

Havok™ Confidential. ©Copyright 2011 Havok.com (and its licensors). All Rights Reserved. Confidential Information of Havok.

Havok Script (2)

- Powerful debugging tools
  - A debugger which integrates into Visual Studio 2008
  - A stand-alone debugging application
- Powerful profiler

Havok^{TM} Confidential. ©Copyright 2011 Havok.com (and its licensors). All Rights Reserved. Confidential Information of Havok.

Havok Script Plugin

- Integrates into Visual Studio
- The IDE becomes a common environment for developing Lua and C++
- Cross-language debugging:
  - from C++ into Lua
  - from Lua into C++

HavokTM Confidential. ©Copyright 2011 Havok.com (and its licensors). All Rights Reserved. Confidential Information of Havok.

Havok Script Debugger

- Much lighter weight.
- Offers all the debugging features of the Plugin
  - Except C++ debugging
- Ideal for “scripters”

HavokTM Confidential. ©Copyright 2011 Havok.com (and its licensors). All Rights Reserved. Confidential Information of Havok.

Both debuggers offer

- Callstack
- Locals
- Breakpoints (conditions, hit count, *coroutine*)
- Watches
- Coroutines
- Interactive console
- Registry

havok

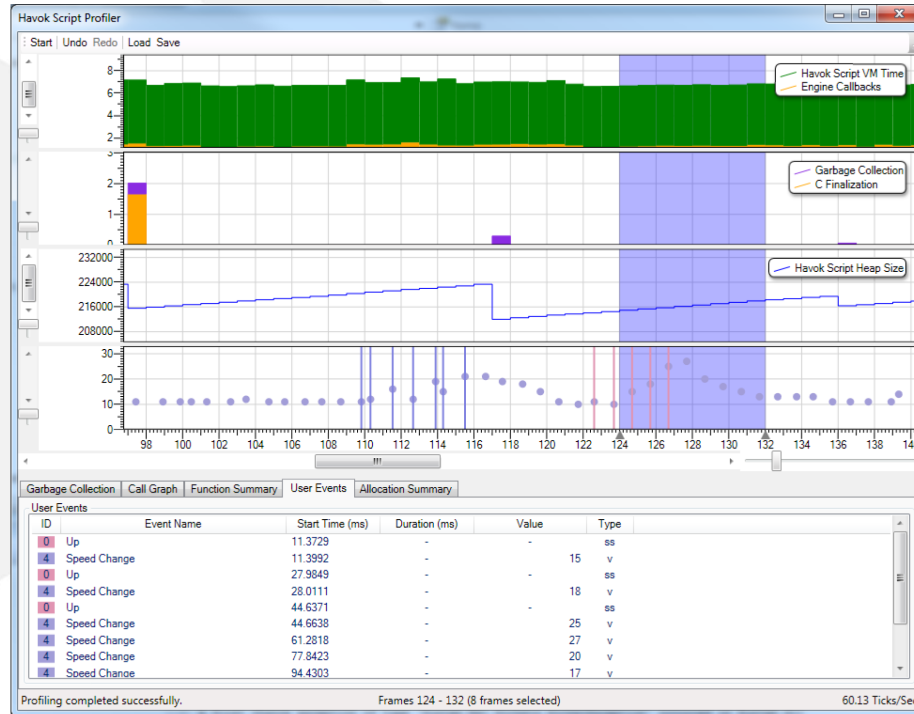
HavokTM Confidential. ©Copyright 2011 Havok.com (and its licensors). All Rights Reserved. Confidential Information of Havok.

Profiling

- Inclusive and exclusive function times and allocations.
- Call graph
- Garbage collection times
- Real-time graphs
- Hazard identification
- Allocation summary
- User event graph (next release)

Havok^{TM} Confidential. ©Copyright 2011 Havok.com (and its licensors). All Rights Reserved. Confidential Information of Havok.

Profiling User Events



HavokTM Confidential. ©Copyright 2011 Havok.com (and its licensors). All Rights Reserved. Confidential Information of Havok.

STRUCTURES

HavokTM Confidential. ©Copyright 2011 Havok.com (and its licensors). All Rights Reserved. Confidential Information of Havok.

Structures (1)

- The flexibility of Lua’s table comes at a cost
  - You can’t say “This table represents a vector and always has entries for x, y and z.”
  - The VM has to account for all possibilities
  - It looks up the key “x” in a hash table
- If you give up some flexibility, you can get a significant performance gain

havok

HavokTM Confidential. ©Copyright 2011 Havok.com (and its licensors). All Rights Reserved. Confidential Information of Havok.

Structures (2)

- Declare a prototype for a **structure**:

```
kstructure Vector2
        x : number
        y : number
end
```

- And then later create an instance s of the structure

- When you subsequently write s.x, the interpreter *knows* that the structure s has an entry for x

Havok™ Confidential. ©Copyright 2011 Havok.com (and its licensors). All Rights Reserved. Confidential Information of Havok.

Structures (3)

- Structures are an optimization:
  - structures were designed as a drop-in replacement for tables
  - You can identify certain tables in existing Lua code as being good candidates for structures
    - The table will have a stable set of keys at run-time.
  - Easy to convert to structures, since the interface is the same
  - Much faster
  - Much lower memory

havok

HavokTM Confidential. ©Copyright 2011 Havok.com (and its licensors). All Rights Reserved. Confidential Information of Havok.

Efficient Lua Scripting in Game Production

CLOSING REMARKS

HavokTM Confidential. ©Copyright 2011 Havok.com (and its licensors). All Rights Reserved. Confidential Information of Havok.

Summary (1): Lua Advantages

- Fast
- Small
- Good set of language features for games

HavokTM Confidential. ©Copyright 2011 Havok.com (and its licensors). All Rights Reserved. Confidential Information of Havok.

Summary (2): Havok Script Advantages

- Performance
  - VM optimized for each platform
  - Language extensions
  - Accurate profiling
- Productivity
  - Havok Script Plugin
  - Havok Script Debugger

Havok™ Confidential. ©Copyright 2011 Havok.com (and its licensors). All Rights Reserved. Confidential Information of Havok.