Processo de criação de um wrapper para utilização do Wii Remote em aplicativos Lua.

Leandro Freire Paulo Motta

Centro Universitário Carioca, Brasil – Rio de Janeiro RJ

Resumo

Este artigo visa abordar o processo de desenvolvimento de um wrapper da biblioteca Wiiuse, para que seja possível a interação entre o Wii Remote e o computador pessoal em jogos e aplicativos desenvolvidos em Lua.

Keywords: Wii, Wii Remote, Bluetooth, Lua, Löve, Infravermelho, Wrapper, Desenvolvimento, Jogos

Authors' contact:
lf_oliver@yahoo.com.br
prmottajr@gmail.com

1. Introdução

Com a popularização da Internet, jogos no PC evoluíram a passos largos, mas os fabricantes de console prosseguiram com suas pesquisas, e a partir da sexta geração (PlayStation 2, Xbox, GameCube) de consoles, o PC deixa de ser o centro das atenções no mundo dos jogos, passando a ter uma fatia muito pequena da indústria de jogos [BRIGHTMAN 2009]. Com a sétima geração de consoles (PlayStation 3, Xbox 360, Wii) e portáteis (PSP e Nintendo DS), [SAKAZAKI 2006] a supremacia dos consoles sobre o PC se solidifica.

Consoles passaram a utilizar as mesmas arquiteturas de um computador, a forma de comunicação com seus periféricos (joysticks e outros) passa a ser uma amplamente utilizada no mundo PC: o Bluetooth. Essa adoção faz com que a comunicação destes periféricos com outros sistemas que não o original seja possível. Esta possibilidade despertou o interesse em conectá-los ao computador. O Wii Remote, periférico principal de interface com o console Wii da Nintendo e um dos focos deste artigo, mudou o paradigma de interação com os jogos. O conceito de joystick, no qual se utiliza uma haste ou D-pad para interação com o jogo, foi modificado, pois agora o próprio controle, utilizando um acelerômetro embutido e triangulação através de infravermelho, pode ser utilizado para repassar as coordenadas para o console, algo como controlar a mira de uma arma em um jogo de guerra passa a ser feito apenas apontando o controle para a área da tela desejada.

2. Trabalhos Relacionados

Alguns dos programas populares são o GlovePIE para Windows e o DarwiinRemote para o Macintosh. Algumas utilidades extra-wii criadas pela comunidade são as do programador Johnny Lee, que desenvolveu interfaces utilizando rastreamento da mão ou quadros brancos com interatividade multiponto [LEE 2007] e pesquisadores da Universidade de Memphis, que utilizaram para coleta de dados de experimentos de psicologia cognitiva [SCIENCEDAILY 2008] e mesmo a Autodesk, desenvolveu um plugin para o seu programa Autodesk Design Review que dá ao usuário um melhor controle da orientação 3D dos objetos na tela através dos gestos feitos com o wiimote [GARDINER 2008].

Atualmente as tecnologias mais desenvolvidas são o Kinect [Microsoft 2011] e o PSMove [Sony 2011]. Estas tecnologias apresentam maior precisão quando comparadas ao Wii Remote o que levou a Nintendo a lançar um dispositivo extra chamado MotionPlus que incorpora um giroscópio ao projeto [Nintendo 2010]. No entanto não existe ainda suporte a esse dispositivo na biblioteca Wiiuse o que limita sua exploração

Existem outras bibliotecas que desempenham o mesmo papel como a WiiYourself em C++, WiiRemoteJ em Java e a PyWii em python, todas com suporte completo, sendo a última também um wrapper da Wiiuse.

3. Conceitos e Definições

Alguns conceitos e definições básicas precisam ser explanados antes que as soluções possam ser procuradas ou expostas.

3.1 Lua

Lua é uma linguagem de script rápida e leve, projetada para estender aplicações.

Lua combina sintaxe simples para programação procedural com estrutura para descrição de dados baseadas em tabelas associativas e semântica extensível. Lua é tipada dinamicamente, é interpretada a partir de bytecodes para uma máquina virtual baseada em registradores e tem gerenciamento automático de memória com coleta de lixo incremental. Essas características fazem de Lua uma linguagem ideal para

configuração, automação (scripting) e prototipagem rápida.

Lua foi criada em 1993 no Brasil por Roberto Ierusalimschy, Luiz Henrique de Figueiredo e Waldemar Celes, integrantes do Grupo de Tecnologia em Computação Gráfica – Tecgraf, da PUC-Rio (Pontifícia Universidade Católica do Rio de Janeiro).

3.2 Wiiuse

A Wiiuse é uma biblioteca escrita na linguagem C para que seja possível a comunicação entre o computador e os periféricos do Wii programaticamente.

3.3 Wrapper

Wrappers consistem de uma camada de poucas linhas de código que traduzem ou abstraem a interface existente de uma biblioteca em uma interface compatível com outra linguagem. Wrappers auxiliam nos fatores de qualidade do software, como facilidade de uso, alto nível de correção, extensibilidade e portabilidade.

Wrappers também proporcionam facilidade de uso e de aprendizado, ao encapsular detalhes sutis de programação, proporcionando interfaces compactas e mais simples [SCHMIDT 1992].

4. Descrição da Implementação

O wrapper foi escrito em C utilizando a API C de Lua, permitindo a interação entre os dois ambientes. Existem duas abordagens para utilização de Lua: como linguagem de extensão e como linguagem extensível. Utilizaremos a segunda abordagem na qual o código C é visualizado como uma biblioteca. Desta forma, cada função da biblioteca Wiiuse terá uma correspondente no ambiente Lua.

Todo o processo de comunicação entre Lua e C pode ser definido em: gerenciamento de dados, interface e registro de funções dos quais detalharei a seguir:

Gerenciamento de dados - O componente mais importante na comunicação entre Lua e C é uma Pilha abstrata, capaz de armazenar qualquer valor Lua. Toda troca de dados entre C e Lua é feito pela Pilha de modo que quase todas as funções da API C operam nos valores da mesma. Além de fornecer a base da interoperabilidade, ajuda a resolver o problema da tipagem dinâmica de lua contra a tipagem estática de C assim como a desalocação de recursos que em Lua ocorre automaticamente e em C manualmente. A manipulação dos dados na Pilha consiste basicamente de funções luapush* correspondentes a cada tipo primitivo em C (e.g., luapushnumber), que adiciona valores à Pilha, e de funções luaL_check* que recuperam um valor na pilha, fazem um cast do valor recuperado para o tipo escolhido e lança uma exceção caso não seja o tipo esperado (e.g., luaL_checkint). Lua mantém todo seu estado (variáveis globais) inclusive a pilha em uma estrutura dinâmica chamada lua_State e uma referência para esta estrutura é passada como argumento para todas as funções dentro de Lua. A Cada retorno de função, Lua limpa automaticamente o conteúdo da Pilha que está abaixo do resultado [Ierusalimschy 2006].

Interface - Para que seja possível utilizar funções C em Lua, devemos seguir uma especificação, que consiste em capturar os argumentos passados e retornar a quantidade de resultados, além da assinatura definida em lua.h: typedef int (*lua_CFunction) (lua_State *L).

Listagem 1: *conexão com um Wii remote.*

```
#include "lua.h"
#include "lualib.h"
#include "lauxlib.h"
#include "wiiuse.h"

 wiimote **remote;

 static   int   wiiuse_connectL(lua_State
*L){
   1 int n = luaL_checkint(L,1);
      int           connected           =
wiiuse_connect(remote,n);
   2 lua_pushnumber(L,connected);
      return 1;
 }
```

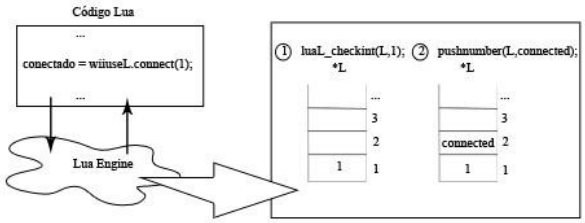

Figure 1: *Troca de dados*

O header lua.h define as funções básicas como lua_pushnumber, para manipulação da Pilha. O header lauxlib.h define as funções fornecidas pela auxiliary library (auxlib), como luaL_checkint. A auxlib disponibiliza funções definidas em lua.h em um nível de abstração elevado. As funções em lua.h e lauxlib.h são prefixadas por lua_ e luaL_ respectivamente [Ierusalimschy 2006].

Registro de funções - Neste momento, temos uma funcionalidade encapsulada porém inacessível. Lua fornece um mecanismo em que as funções são armazenadas em uma tabela (principal estrutura de dados disponível em Lua) o que permite um acesso direto a seu endereço em memória, tornando desnecessário a dependência de um nome de função específico, regras de visibilidade/escopo ou localização de pacotes. A função luaL_register é responsável pelo link entre as funções C e Lua.

Listagem 2: *registro de funções*

```
static const struct luaL_Reg wiiuseL[] =
{            {"connect",wiiuse_connectL},
{NULL,NULL}};

int luaopen_wiiuseL(lua_State *L){
   luaL_register(L,"wiiuseL",wiiuseL);
   return 1;
}
```

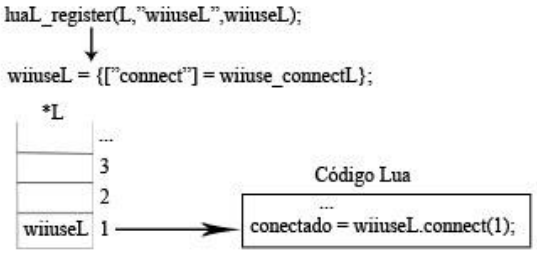

Figure 2: *Registro de funções*

Declaramos um array, wiiuseL, contendo elementos do tipo luaL_Reg, uma estrutura com dois campos: um string, que representa o nome da função a ser chamada em Lua e uma referência para a função relacionada. Devemos adicionar uma referência em wiiuseL para cada função que desejamos utilizar em Lua. Por fim, um último elemento {NULL,NULL} indicando o fim da lista. A próxima função segue a mesma especificação descrita anteriormente, com exceção do nome que deve seguir o formato luaopen_*. Lua é composto por pequenas bibliotecas específicas, como IO e math, cada grupo de funções é registrado em tabelas através de funções luaopen_* (e.g., luaopen_io) definidas em lualib.h. Quando declaramos luaopen_wiiuseL, estamos especificando uma maneira de utilizarmos uma nova biblioteca. A chamada a luaL_register cria uma tabela "wiiuseL" e a preenche com o array "wiiuseL", em seguida retornamos a quantidade de valores adicionados a "L".

Após o término da biblioteca, devemos conectá-la ao interpretador, criando uma biblioteca dinâmica (wiiuseL.dll no Windows, wiiuseL.so em sistemas Unix) e carregando diretamente em Lua com "require" (e.g., require "wiiuseL") que encontra a função luaopen_wiiuseL, registra como função C e a chama, possibilitando o uso das funções da nova biblioteca [Ierusalimschy 2006].

5. Conclusão

Este artigo apresentou um estudo sobre como criar um wrapper em Lua, com o objetivo de possibilitar a utilização do controle Wii Remote da Nintendo em um jogo de PC.

Um jogo de plataforma 2D foi desenvolvido a fim de testar o wrapper. Utilizamos o framework Löve2D, feito em Lua, no qual obtivemos total sucesso na integração do Wii Remote, fazendo uso de seu acelerômetro, ir tracking e botões. Testamos também no LuaSDL, um wrapper da biblioteca SDL para Lua e no LuAllegro, wrapper da biblioteca Allegro.

Uma das vantagens de se desenvolver um wrapper, além da modularidade que é proporcionado, é a velocidade em prototipagem e desenvolvimento, aliado a toda estrutura disponibilizada por Lua, que facilitou a implementação e reutilização de código. A desvantagem existente é a dependência de um código externo e a própria limitação que pode ocorrer, no qual destacamos o não suporte ao motion plus e ao audio do Wii Remote pela Wiiuse.

Para trabalhos futuros, o objetivo é fazer com que a wiiuseL suporte os outros periféricos do Wii, assim como a Wiiuse, expandir o suporte para outros sistemas operacionais e aumentar a compatibilidade com outras Engines, como a Baja Engine, Apocalyx, GeexLab dentre outras. A versão para Windows da WiiuseL, a documentação e o jogo podem ser baixados em `https://github.com/leandrofre/WiiuseL`.

Referências

BRIGHTMAN, JAMES. PC GAMES SALES BRING US INDUSTRY TO $18.85 BILLION IN '07. GAMEDAILY, 2008. DISPONÍVEL EM <HTTP://WWW.GAMEDAILY.COM/ARTICLES/NEWS/PC-GAME-SALES-BRING-US-INDUSTRY-TO-1885-BILLION-IN-07>. ACESSO EM JUN 10.

GARDINER, GARIN. ANOTHER USE FOR A WIIMOTE. IN THE MACHINE, 2008. DISPONÍVEL EM <HTTP://MFGCOMMUNITY.AUTODESK.COM/BLOGS/BLOG/VIEW/6/ANOTHER_USE_FOR_THAT_WIIMOTE/>. ACESSO EM MAI. 2010.

IERUSALIMSCHY, ROBERTO. PROGRAMMING IN LUA, 2ND ED., 2006, 217-224, 241-245 .

LEE, JOHNNY C. LOW-COST MULTI-POINT INTERACTIVE WHITEBOARDS USING THE WIIMOTE. JOHNNY CHUNG LEE, 2007. DISPONÍVEL EM <HTTP://JOHNNYLEE.NET/PROJECTS/WII >. ACESSO EM MAI 10.

MICROSOFT. XBOX 360 KINECT SENSOR, INSTRUCTION MANUAL, 2011.

NINTENDO. WII MOTIONPLUS, INSTRUCTION MANUAL, 2010.

SAKAZAKI, LLOYD. SEVENTH GENERATION GAMING CONSOLES: THINKING OUTSIDE THE BOX. SEEKING ALPHA, 2006. DISPONÍVEL EM <HTTP://SEEKINGALPHA.COM/ARTICLE/22075-SEVENTH-GENERATION-GAMING-CONSOLES-THINKING-OUTSIDE-THE-BOX>. ACESSO EM NOV 10.

SCHMIDT, DOUGLAS C. SYSTEMS PROGRAMMING WITH C++ WRAPPERS: ENCAPSULATING INTERPROCESS COMMUNICATION SERVICES WITH OBJECT-ORIENTED

INTERFACES, C++ REPORT, SIGS, VOL. 4, NO. 7, SEPTEMBER/OCTOBER, 1992, P1-6. .

SCIENCEDAILY. NINTENDO WII WITH A NEW MISSION: WIIMOTE AS AN INTERFACE BRIDGING MIND AND BODY. SCIENCEDAILY, 2008. DISPONÍVEL EM <HTTP://WWW.SCIENCEDAILY.COM/RELEASES/2008/03/080304200905.HTM>. ACESSO EM DEZ 10.

SONY. PLAYSTATION®MOVE NAVIGATION CONTROLLER, INSTRUCTION MANUAL, ABOUT PLAYSTATION®MOVE PRODUCT, 2011.