# Uso do recurso “Canvas” do NCL/LUA para desenvolvimento de jogos e aplicativos para Televisão Digital

FABIO CARDOSO *

UNIVERSIDADE ESTADUAL PAULISTA - UNESP

## Resumo

Este trabalho detalha as estratégias de uso do elemento “Canvas” oriundo do NCL para o desenvolvimento de aplicações interativas em NCLua para Televisão Digital, no contexto do projeto “Apolônio e Azulão” da TV Unesp. Aponta os prós e contras de se escolher esta abordagem para programar um jogo que seja executado em um televisor, indica algumas técnicas de otimização para uma melhor performance e também aponta soluções para as diferenças entre as implementações do Ginga em alguns modelos de televisores presentes no mercado.

**Palavras-Chave**: Televisão Digital, GINGA, LUA, interatividade, jogos.

**Contato do autor**:
* fabio@tvu.unesp.br

## 1. Introdução

A TV Unesp, emissora de televisão cultural e educativa sediada em Bauru, São Paulo. e mantida pela UNESP, foi uma das dez emissoras contempladas com um laboratório do projeto GINGA BR Labs, do Ministério das Comunicações. Este projeto entregou a cada uma das concessionárias um laboratório que contém equipamentos capazes de simular uma emissora de televisão de alcance curto, e que transmite um sinal que possui um canal de dados, além dos de áudio e vídeo. Isso possibilita, juntamente com os dois aparelhos de televisão que compõem o laboratório, testar de modo fácil e rápido o desenvolvimento de aplicações interativas em Ginga sem o uso de emuladores ou simulações, partindo para a avaliação dos aplicativos diretamente no ambiente em que eles serão executados quando forem transmitidos de fato pela emissora.

Essa possibilidade acabou se revelando um divisor de águas no que se relaciona com o desenvolvimento de aplicações em Ginga dentro da TV Unesp. Antes, testar um aplicativo em ambiente de produção exigia o envio do mesmo para o sinal que está no ar, transmitindo a aplicação para todos os espectadores da emissora, obrigatoriamente. Como este expediente se mostrou não adequado para os testes frequentes que são comuns nas primeiras etapas de desenvolvimento, os aplicativos que foram desenvolvidos antes do recebimento do laboratório eram testados em máquinas virtuais. Ocorre que existe uma diferença considerável entre o funcionamento das aplicações em uma máquina virtual e em uma televisão, onde o hardware presente é muito mais limitado.

Como primeiro projeto pós implantação do laboratório escolhemos criar um programa infantil, de produção relativamente simples, que possibilitasse o uso de técnicas inovadoras na programação de aplicativos para televisão digital e permitisse a inclusão destas aplicações na narrativa dos episódios. As aplicações deveriam fazer parte do programa e não funcionar como aplicativos isolados. O desenvolvimento do produto uniu as equipes de T.I., produção, operações e design da TV, e todo o projeto gráfico, usabilidade e programação foram feitos dentro da emissora. Ao final, foram produzidas oito aplicações interativas: Cinco jogos, dois infográficos interativos e um “*quiz*”. Todos os aplicativos rodam de modo satisfatório nos dois televisores que fazem parte do laboratório: Um modelo da Sony, que possui a implementação do Ginga mantida pela TQTVD-TOTVS, e um modelo da Philco que possui uma versão do Ginga criada pela Universidade Federal da Paraíba (UFPB). As diferenças entre os aparelhos, bem como as técnicas usadas para a resolução destas inconsistências são detalhadas posteriormente neste artigo.

## 2. O Projeto “Apolônio e Azulão.”

O projeto “Apolônio e Azulão” engloba inicialmente quatro episódios de um programa televisivo, produzido inteiramente com fantoches, que leva ao universo infantil os temas da ciência que são ensinados na escola para as crianças da faixa etária escolhida, que varia entre 7 e 10 anos de idade. O programa se passa em um laboratório, onde o professor Apolônio ensina ao Azulão, uma criatura azul com atitudes de criança, como que os fenômenos da natureza ocorrem: As chuvas, o dia e a noite e as estações do ano, entre outros temas pertinentes ao cotidiano da criança. Cada um destes episódios conta com duas aplicações interativas, integradas ao fluxo de vídeo, que complementam as explicações dadas durante o programa pelo professor Apolônio.

## 3. As aplicações interativas

O primeiro episódio trata do ciclo da água e ensina a criança como que ocorrem as chuvas. Na primeira aplicação interativa, logo no começo do episódio, Azulão questiona Apolônio sobre suas roseiras premiadas, e é convidado a regar uma das roseiras do

professor. Usando as setas do controle remoto, o espectador precisa regar a planta usando um regador, tendo que enchê-lo de água em uma torneira situada do lado oposto da tela. Ao final, cumprindo a missão, o espectador ganha uma medalha virtual. O segundo aplicativo interativo é um infográfico que mostra o ciclo da água, em etapas. Este último aplicativo é exibido ao final do episódio, para auxiliar na fixação do conteúdo por parte da criança. As aplicações interativas do primeiro episódio são de fácil entendimento e execução por parte da criança, sem grandes desafios; o objetivo é que o espectador se acostume com o modus operandi das aplicações, e assimile o paradigma interativo do produto audiovisual que está consumindo.

No segundo episódio as aplicações começam a ganhar robustez e complexidade, exigindo mais interação da criança, com a esperança de que ela já tenha se acostumado durante o primeiro episódio. Neste programa, Apolônio está ensinando ao seu pupilo o conceito de "plantar": Azulão chega para o professor com a ideia esdrúxula de "plantar brinquedos" já que sua mãe tinha lhe dito que bastaria plantar as coisas para que elas surgissem da terra.

A primeira aplicação do segundo episódio convida o espectador a plantar algo em vários vasos, obedecendo a seguinte ordem: primeiro a terra, depois a semente, e por último a água, para que a planta germine. A criança deverá, usando o controle remoto, mover uma mão com as setas e usar os botões coloridos para trocar os itens que a mão carrega. Pressionando a seta para baixo, o item é colocado no vaso. Plantando os quatro vasos corretamente, a criança ganha mais uma medalha.

A segunda aplicação do segundo episódio trata da questão da diferenciação entre "flores" e "frutos", explicada pelo professor Apolônio durante o episódio. Neste jogo, o espectador é convidado a mover uma câmera fotográfica usando as setas do controle remoto, e pressionar "OK" para bater uma foto. No cenário, estão dispostas três plantas que dão flores, e três plantas que dão frutos, e a criança precisa tirar as fotos de acordo com o que a aplicação pede: primeiramente as três plantas que dão fruto, e por último as três plantas que dão flores. A câmera é movimentada livremente pelo cenário, e não existe nenhuma limitação da localização em que a criança pode bater uma foto, e nem do local exato da planta que ela necessita apertar o botão. Fotos da mesma planta não contam (a aplicação avisa caso uma foto seja repetida) e tirar fotos de frutos quando é pedido que se tire de flores ou vice-versa conta como um erro.

O terceiro episódio fala sobre o conceito de dia e noite. A primeira aplicação é um *quiz* interativo, onde a criança deve dizer se sabe o que é uma lenda, assunto que está sendo tratado durante o episódio de forma secundária. Já a segunda aplicação leva a criança ao "espaço", onde lhe é apresentada uma vista do planeta Terra com uma figura do Azulão parada sobre o Brasil. A criança é convidada a escolher o que o Azulão está fazendo naquele determinado momento do seu dia, baseada apenas no modelo do planeta e da incidência de luz sobre o país.

O quarto episódio fala sobre as estações do ano, e na primeira aplicação o espectador é convidado a descrever o "Dia de Verão" do personagem. São apresentadas várias situações e objetos que remetem a variadas estações do ano, e a criança deve escolher os objetos que fazem parte do verão. Os objetos escolhidos vão aparecendo no corpo do Azulão desenhado na tela.

A segunda aplicação do quarto episódio novamente leva a criança para o espaço, desta vez com um modelo da Terra que indica a ela as várias estações do ano. A criança deve dizer qual estação do ano é correspondente ao momento em que a Terra está.

## 4. Programando as aplicações

Todas as aplicações foram desenvolvidas em NCL/Lua, com o NCL possuindo um papel secundário, de mero interlocutor do middleware com os scripts LUA, que desenhava todas as imagens e as movimentava em um elemento "Canvas" dentro do NCL. O suporte a animações, eventos e até a velocidade de renderização oferecida dentro da implementação do NCL puro não foi satisfatória em nenhuma das plataformas, restando a alternativa de se trabalhar as primitivas de desenho e descarte dos elementos manualmente, via código. A ideia é usar técnicas similares as da produção interativa usando o elemento *canvas* do HTML5, onde o JavaScript faz o mesmo papel que o LUA e atua como linguagem base para a produção de um jogo.

Quando se opta por sair do ambiente controlado do NCL e parte-se para um paradigma onde o programador é o responsável pelo gerenciamento de todos os elementos de sua programação, cresce a complexidade do código, mas ganha-se muito no desempenho e possibilidades. Por exemplo, as animações de elementos usando NCL puro são lineares e sem *easing*, enquanto usando um *canvas* desenhado através de um script LUA lhe proporciona a possibilidade de criar suas próprias equações de movimento, dando realismo as animações.

## 5. As limitações dos aparelhos de TV e as soluções encontradas para este problema

Outro ponto chave do uso do elemento *canvas* para a produção dos aplicativos é a otimização das áreas de desenho. Quando testados em um computador, os aplicativos atingem facilmente o desempenho máximo, desenhando a tela em constantes 60 frames por segundo, sem exigir otimizações. Porém, quando

executados em um televisor, a taxa de frames cai muito, inviabilizando o uso do aplicativo. Isso ocorre porque o hardware do televisor é muito mais limitado do que o de um computador.

Durante o desenvolvimento dos aplicativos, além do acesso aos dois televisores do laboratório, houve a possibilidade de testar os produtos em um outro aparelho, também da Sony, porém fabricado em 2014 e com hardware de ponta. Como comparativo, o televisor Sony do laboratório (fabricado em 2012), possui uma capacidade de processamento informada pelo aparelho de 400 Mips, que é equivalente a um processador Intel 486 DX2. Todas as aplicações em Ginga precisam desenhar vários elementos gráficos no equivalente a uma área de 1280 pixels de largura por 720 pixels de altura, que é o padrão usado pelas diversas implementações do Ginga para desenho na tela. O valor causa estranheza, já que os televisores comumente possuem uma resolução maior, de 1920 pixels de largura por 1080 pixels de altura, porém é consistente entre todas as implementações do Ginga testadas até hoje.

Embora a situação do desempenho tenha melhorado no televisor mais novo, que apresentou uma taxa de desempenho informada de 1980Mips, portanto bem maior do que a do outro televisor de mesma marca, a criação de aplicativos interativos para televisão digital deve se pautar pelo menor denominador comum, já que é impossível saber qual o televisor que o espectador possui em casa. Este cenário exigiu uma série de otimizações de código em todas as aplicações, para que não fosse perdida a fluidez de movimento do produto e nem a parte gráfica ficasse comprometida.

A principal mudança necessária foi não desenhar e apagar a tela a cada frame. Uma das características do elemento “Canvas”, tanto no Ginga quanto no HTML, é de funcionar de modo semelhante a uma “lousa mágica”, onde tudo que for desenhado nela permanece até que seja apagado. Desenhos subsequentes no mesmo “Canvas” sem que se crie uma rotina de apaga-lo de alguma forma irão ser “carimbados” na tela em cima das imagens que já estavam lá, o que nem sempre é o efeito desejado para o programador, embora o uso criativo de certas técnicas de desenho possa aproveitar este comportamento.

Devido a um problema com o aplicativo do Ginga para Windows desenvolvido pela PUC, todo o desenvolvimento inicial em Ginga feito pela TV Unesp usando o “Canvas” exigia que a tela fosse limpa de modo completo a cada desenho na tela. O Ginga4Windows não possui a implementação correta do comando “clear()” do NCLua e apaga o quadro todo quando esta função é invocada. A ação de apagar o quadro todo e redesenhar tudo é muito custosa para o hardware limitado do televisor e derruba a performance do aplicativo a níveis inaceitáveis. Uma maneira de se contornar a limitação do Ginga4Windows é desenhar um quadrado transparente no local que você quer apagar, usando “canvas:attrColor()” e depois “canvas:drawRect()”. Isso possibilita que o programador otimize os “redesenhos” de sua aplicação, limitando a área de pintura do “Canvas” a apenas a área desejada. Porém, no Ginga4Windows, cada redesenho parece ocupar um pedaço da memória e desenhar um quadrado transparente aparentemente não limpa aquela área alocada na memória do sistema, levando a um “*memory leak*” que acaba finalizando a aplicação quando ela atinge os 4 Gigabytes de memória máximos que uma aplicação pode tomar para si dentro do sistema Windows.

A implementação do middleware Ginga nos televisores se mostrou consideravelmente superior do que o Ginga for Windows, obedecendo todas as normas relativas aos desenhos em um “Canvas” previstas nos documentos do middleware Ginga na ABNT. Isso tornou possível usar as funções nativas de limitação de desenho e executar todas as aplicações com uma taxa de frames por segundo superior a 30 em ambos os aparelhos testados.

Outra limitação encontrada nos aparelhos testados foi o tamanho da memória de vídeo disponível para as aplicações. Ambos os televisores possuem 32Mb de RAM de vídeo, e não há um modo confiável de se checar a utilização desta memória durante a execução dos scripts. É importante frisar que a maneira que esta memória de vídeo é usada não torna simples calcular o seu uso: Uma imagem é alocada na memória de vídeo do televisor como um bitmap não comprimido, e não com a extensão original do arquivo de imagem usado: Um arquivo png de 1280x720, mesmo que composto somente pela cor preta, ocupará quase um quarto da memória de vídeo disponível para a aplicação. Todas as chamadas de desenho feitas após o preenchimento completo da memória de vídeo usam uma espécie de “memória swap” do televisor, derrubando a performance da aplicação.

Alguns truques, aproveitando-se dos recursos que o *canvas* oferece, juntamente com a possibilidade da gerência de memória através do “garbage collector” do LUA, que se encarrega, quando chamado manualmente, de eliminar as imagens alocadas na memória de vídeo do televisor quando não são mais necessárias na aplicação, ajudam a driblar estas limitações. Alocar uma imagem na memória do televisor, e posteriormente aplicar-lhe um filtro de escala usando o comando “canvas:attrScale()”, não funcional na máquina de Ginga da PUC do Rio, mantém a imagem menor na memória e apresenta uma imagem escalada na tela, embora com perda de definição devido aos filtros aplicados na operação. No entanto, devido a estética minimalista dos gráficos das aplicações, que utilizavam poucas cores e traços grandes, não houve perdas significativas com o uso deste recurso, e foi possível manter todas as aplicações dentro deste limite de memória, mesmo com muitas delas utilizando-se de imagens grandes ou muitos frames sequenciais para as animações, como os fundos

animados de introdução das aplicações do primeiro e segundo episódios e uma sequência de mais de 70 imagens do globo terrestre alternadas durante as aplicações do terceiro e quarto episódios.

## 6. Considerações Finais

A utilização do "Canvas" para a criação de aplicações NCLUA funcionais é uma solução a se considerar quando se quer driblar as inconsistências ou limitações do NCL puro para a construção de aplicativos interativos para televisão digital. Embora o esforço de programação seja consideravelmente maior, os resultados obtidos justificam esta escolha. Como o "Canvas" é um elemento bastante extensível, aceitando qualquer tipo de imagem e contando com poucas funções de desenho, torna-se possível o desenvolvimento de várias bibliotecas que utilizem-se do recurso, como por exemplo extensões de animação, bibliotecas de física, gerenciamento de objetos, e até plug-ins de importação e exportação de código da tag "Canvas" do HTML para Ginga, ampliando o leque de possibilidades de programação de aplicativos para televisão digital.

## Referências

CERQUEIRA, R.,SANT'ANNA, F., SOARES, L.F.G., 2008. NCLua: Objetos imperativos LUA na linguagem declarative NCL. *Proceedings of the 14th Brazilian Symposium on Multimedia and the Web.* ACM, 83-90.

RUIZ, J.H.L., 2004. Contribuições para o desenvolvimento de aplicações interativas para TV digital baseadas em Ginga-NCL. *Masters Degree thesis, UNICAMP.*

GEARY, D., 2012. Core HTML5 Canvas: Graphics, Animation, and Game Development. *Pearson Education, 2012.*

MANYEN, M., SCHUYTEMA, P., 2005. Game Development with LUA (Game Development Series). *Charles River Media, 2005.*

BRANDÃO, L.R.G., SEGUNDO, R.M.C., TAVARES, T.A., 2010. Jogos eletrônicos na TV Digital brasileira: gêneros e desafios para esta nova plataforma. *Proceedings of Games and Digital Entertainment, Brazilian Symposium, 132-137.*