Manual do Multi-Platform Console

MPC v. 4.7

UFSM – Departamento de Computação Aplicada

Curso de Ciência da Computação

{cicero,pozzer,favera,rissetti,weber}@inf.ufsm.br

Cesar Tadeu Pozzer (Orientador)

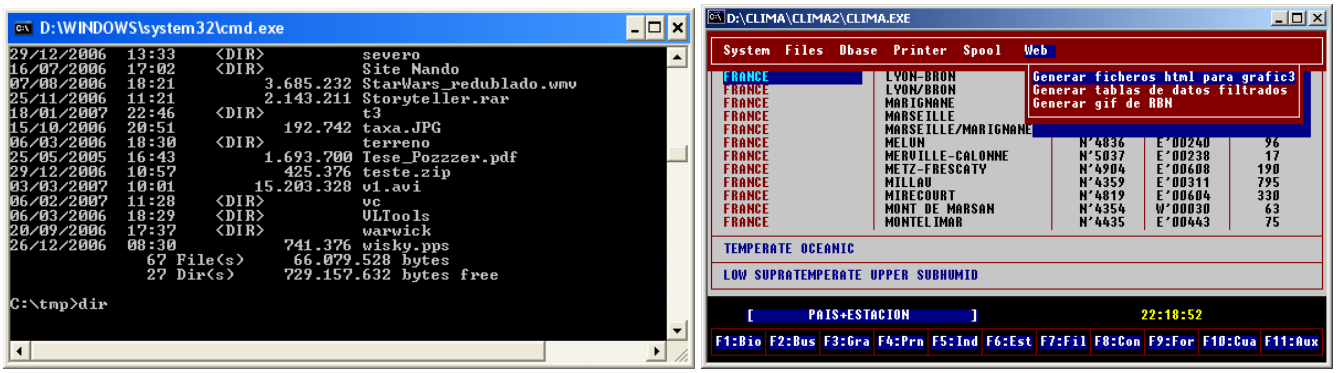
Cicero Augusto de Lara Pahins
Eduardo Ceretta
Fábio Weber
Gustavo Rissetti

Ver. 12/07/2018

Índice

Origem Histórica

Sistemas operacionais mais antigos, como o MS-DOS, bem como compiladores para esta plataforma, disponibilizavam recursos e bibliotecas que permitiam a criação de aplicativos com interfaces em modo texto que possuiam menus, caixas de diálogo e outros *widgets* que tornavam mais amigáveis a utilização dos aplicativos. Estas interfaces, porém, executavam em modo textual, e nelas não era possível exibir gráficos e imagens. São exemplos aplicativos desenvolvidos em Clipper para a plataforma MS-DOS.



Atualmente, na plataforma Windows, bem como no Linux, continuam existindo consoles textuais semelhantes aos consoles do antigo MS-DOS, porém os compiladores modernos não disponibilizam mais recursos para programação visual neste ambiente, necessários para a criação de janelas, caixas de diálogo e definição de cores do texto e background como existia antigamente.

Pode-se questionar a necessidade de aplicações deste tipo em tempos atuais onde a gama de recursos gráficos é extremamente alta, disponibilizada por linguagens modernas orientadas a objetos como o Java e APIs gráficas como QT, Fox, GTK, etc, para construção de interfaces gráficas em linguagem C++. Deve-se observar, porém, que o nível de abstração e o grau de experiência exigido dos programadores para uso destas APIs e linguagens modernas é elevado.

Neste contexto foi desenvolvido a API MPC (*Multi-platform Console*), como forma de permitir que mesmo programadores com pouca experiência em programação possam desenvolver aplicativos com interfaces textuais/gráficas para qualquer plataforma. O MPC é uma **ferramenta educacional** que visa oferecer um conjunto mínimo de recursos, exigindo que o aluno desenvolva todos os elementos de interface, como janelas, diálogos, botões, etc., a partir de elementos básicos.

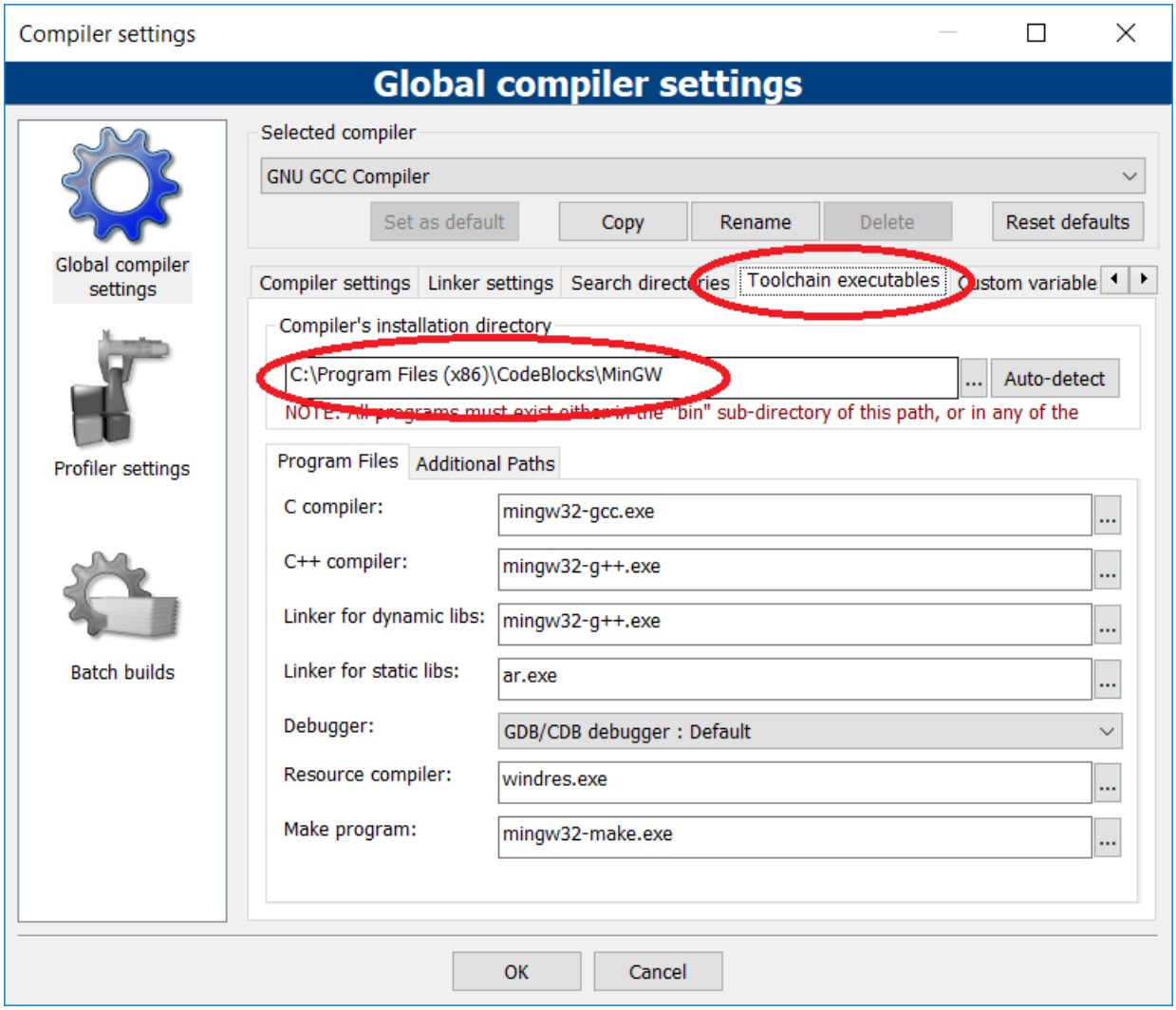
Idéia Geral do MPC

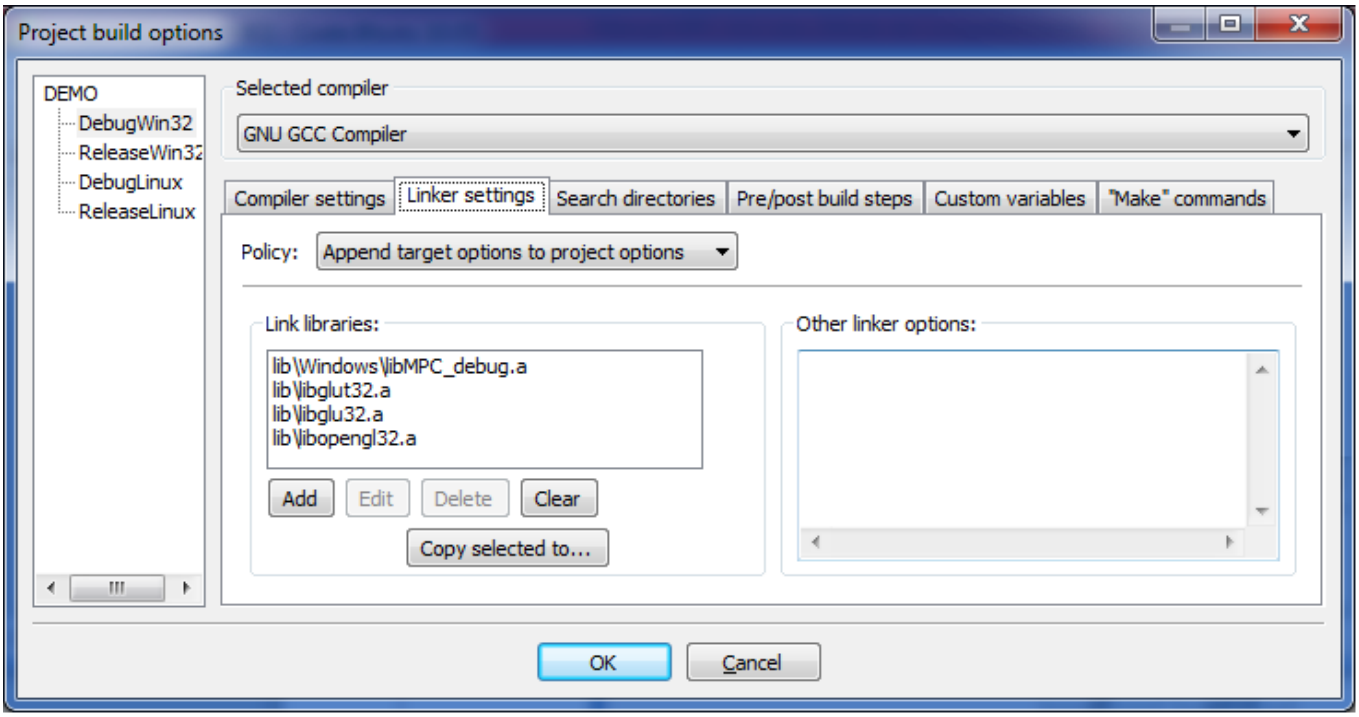
O Multi-Platform Console (MPC) é uma API que emula um console textual. Ele consiste em um conjunto de funções modularizadas que juntas permitem o desenvolvimento de interfaces gráficas baseadas em janelas (baseadas em texto). Ele foi implementado em C++ sobre a API gráfica multi-plataforma OpenGL. Para o tratamento de janelas com o sistema operacional, juntamente com os dispositivos de entrada, utiliza-se a biblioteca auxiliar Glut. O MPC está disponível em uma biblioteca estática (.lib) para o ambiente GNU GCC (Windows e Linux), e com projeto para o ambiente CodeBlocks[1] (ver figura). Deve-se utilizar um compilador C++ para compilar projetos com esta API. A interface com o usuário é por meio de funções em linguagem C, mas todos os

[1] http://www.codeblocks.org/. Essa versão do MPC não dá mais suporte ao Visual Studio visto a frequente mudança de versão do compilador e incompatibilidade com libs criadas para versões mais novas. O Codeblocks, por sua vez, é muito prático, confiável, leve e fácil de usar. Além, também tem versão para Linux, que é um dos propósitos da ferramenta.

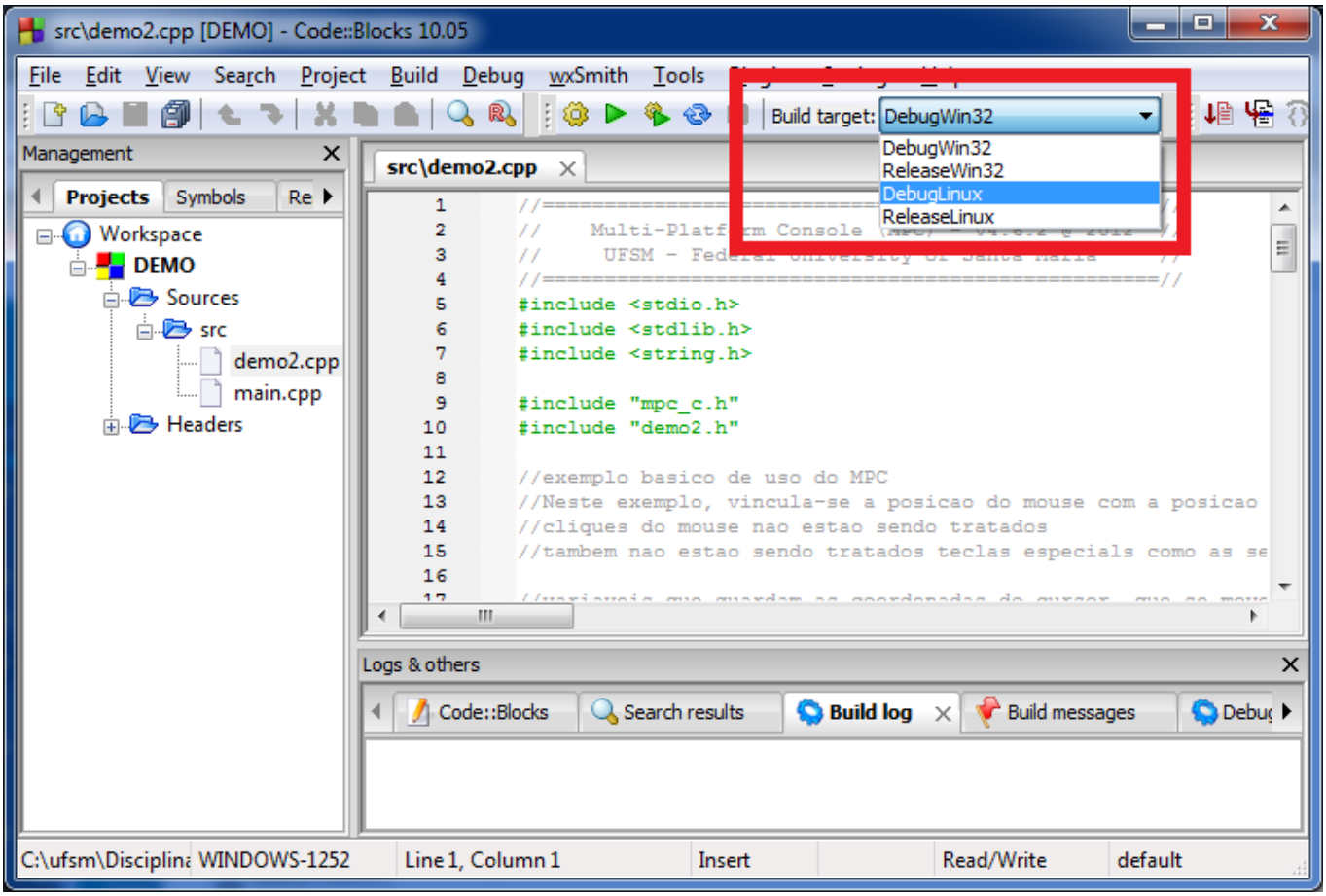
arquivos que fazem parte do projeto devem ter extensão C++. Pode ocorrer erro de link caso for adicionado arquivo com extensão .C.

Caso houver erro de link durante a compilação, certifique-se que está usando o compilador MinGW presente dentro da pasta do code::blocks, como mostrado na seguinte tela:





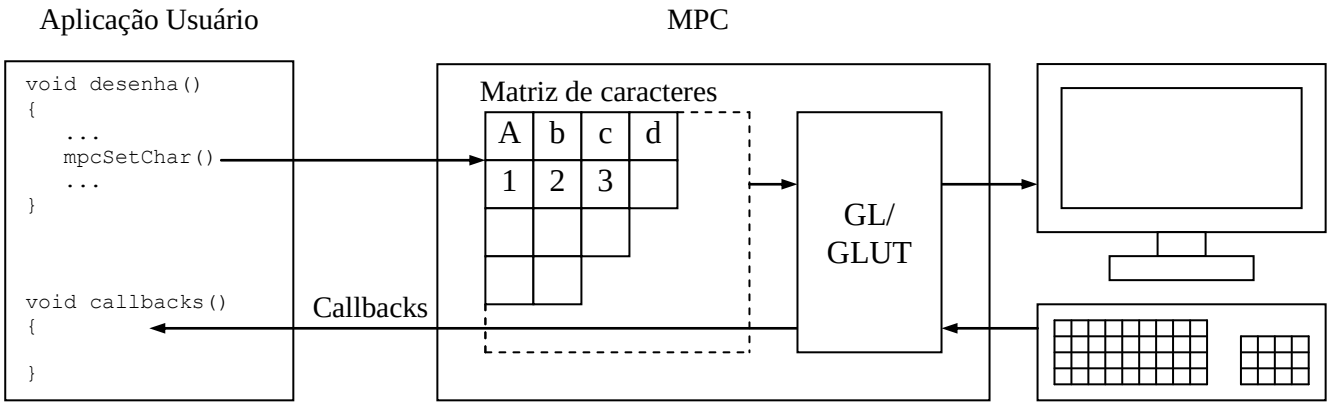
Deve-se selecionar uma das quatro opções de targets: Windows ou Linux, em modo release ou debug, como mostrado na seguinte figura.



O propósito do MPC é fornecer meios para fácil manipulação de caracteres em uma janela retangular. Esta janela é vista como uma grade regular composta por *n* x *m* caracteres (linhas x colunas). Cada caractere possui uma posição pré-estabelecida, com dimensão de 15 por 8 pixels. O MPC suporta uma tela de até 68 x 128 caracteres (linhas e colunas), visto que utiliza uma textura de 1024x1024 pixels para desenho na tela.

Em um cosole textual, o fluxo convencional do texto, a medida que é digitado, é da esquerda para a direita, de cima para baixo. No MPC, o usuário tem a liberdade de definir caracteres em qualquer posição, em qualquer ordem, de forma rápida e intuitiva. Além disso, permite que os caracteres tenham formatações de cores, fontes, estilos e transparências.

Uma vez que o usuário define as dimensões da janela (em caracteres), por meio da função `mpcSetSize()`, é criado um buffer no MPC para gerenciar esta matriz de caracteres. O único trabalho do programador é informar qual caractere deve ser colocado em cada posição da matriz, por meio da função `mpcSetChar()`, **a cada renderização da tela**, como mostrado na seguinte figura. O papel do MPC é desenhar esta matriz na tela, com os atributos associados.



Como existem na tabela ASCII caracteres especiais para desenho de figuras geométricas simples, pode-se utilizá-los no desenho de janelas, permitindo assim criar interfaces gráficas mais intuitivas e bonitas, como apresentado nas próximas seções. Além de caracteres, a API também suporta a incorporação de linhas (horizontais e verticais), figuras (em formato BMP) ou geradas proceduralmente, e gráficos de barra.

Como usar o MPC

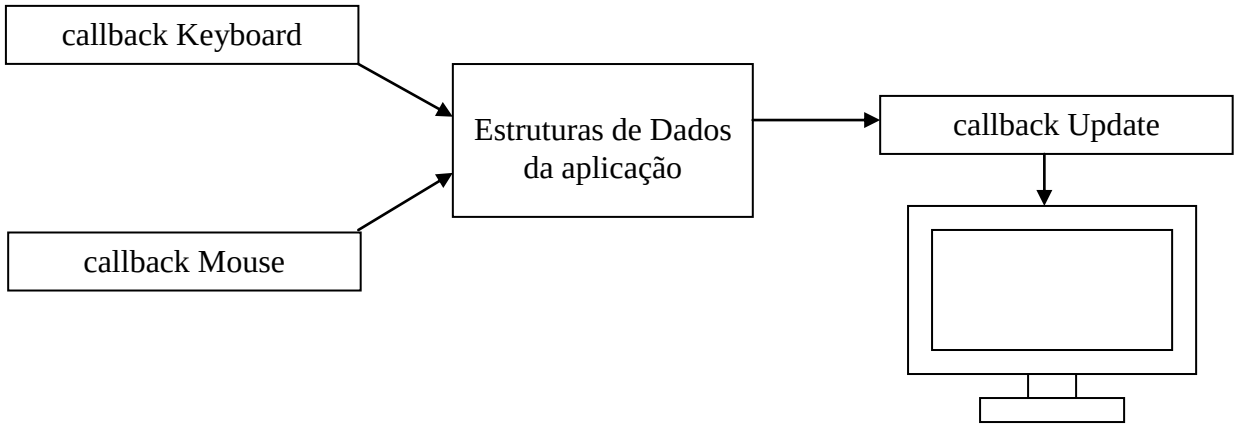
O desenvolvimento de uma aplicação com o MPC é fundamentado em 3 callbacks: Keyboard, Mouse e Update.

```
mpcSetMouseFunc(cbMouse);
mpcSetUpdateFunc(cbUpdate);
mpcSetKeyboardFunc(cbKeyboard);
```

A cada chamada da callback `update()`, definida pela função `mpcSetUpdateFunc()`, deve-se reenviar para o MPC toda informação a ser exibida, incluindo texto, imagens e linhas, caso esta tenha sofrido alguma alteração em relação ao quadro anterior. Esta verificação cabe ao usuário da API. Se nenhuma informação for atualizada, reduz-se o consumo de CPU na reexibição da informação pelo MPC. Internamente, o MPC faz uso de uma textura do OpenGL para agilizar a renderização na tela.

A callback `update()` é a **única** que deve desenhar algo na tela (`mpcHLine()`,`mpcVLine()`, `mpcShowImg()`, `mpcSetChar()`). Ela deve ler os dados das estruturas de dados e enviar para exibição no monitor. Pelas callbacks

`Mouse` e `Keyboard`, deve-se ler dados fornecidos pelo usuário e então alimentar as estruturas de dados da aplicação desenvolvida, como mostrado na seguinte figura.



Propósito da Ferramenta

Essa ferramenta foi desenvolvida com o propósito de servir como base **educacional** para o desenvolvimento de trabalhos acadêmicos, mas especialmente em disciplinas iniciais de programação em computação como Estruturas de Dados. A ferramenta oferece um recurso mínimo de funções para gerenciamento da interface. E esse é o foco principal da ferramenta. Cabe ao programador criar sobre ela interfaces e componentes necessários.

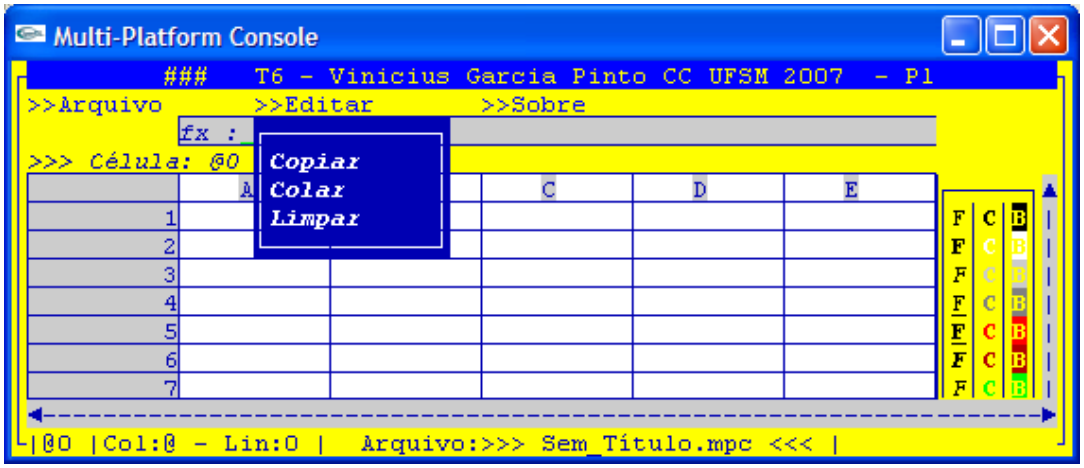
São ilimitados os componentes de interface (widgets) que podem ser criados e gerenciados com a ferramenta. Pode-se implementar, por exemplo, caixas de diálogo, botões, menus, barra de rolagem, tab panes, barras de status, checkbox, combobox, etc.

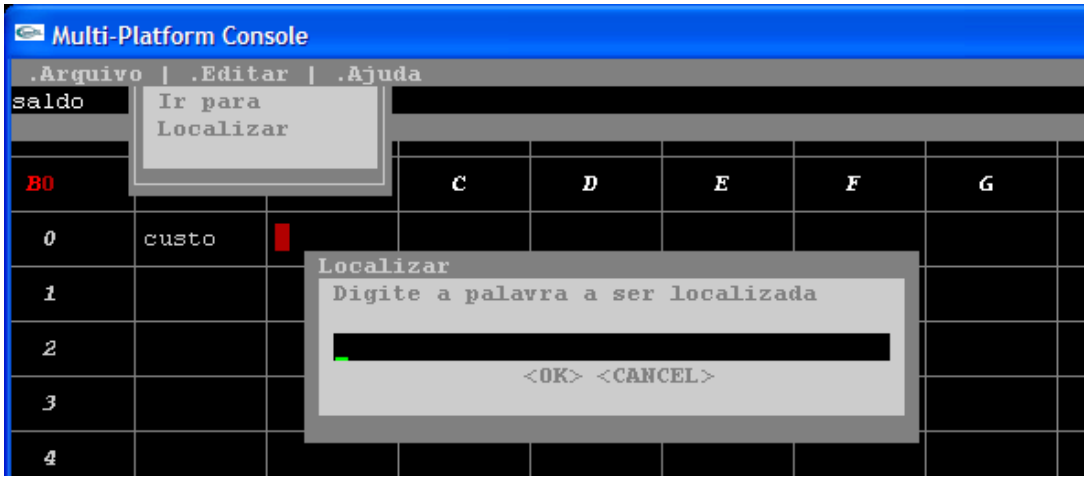
Com a implementação desses componentes, juntamente com algoritmos de gerenciamento de dados (estruturas de dados), o programador pode então implementar aplicativos mais complexos, como editores de texto, planilhas eletrônicas, browsers, sistemas de gerência de informação, navegador de diretórios em disco, programas de desenho e gerenciamento de imagens, dentre outros. Para programadores mais experientes, recomenda-se o uso da linguagem C++ para a criação dos componentes de interface.

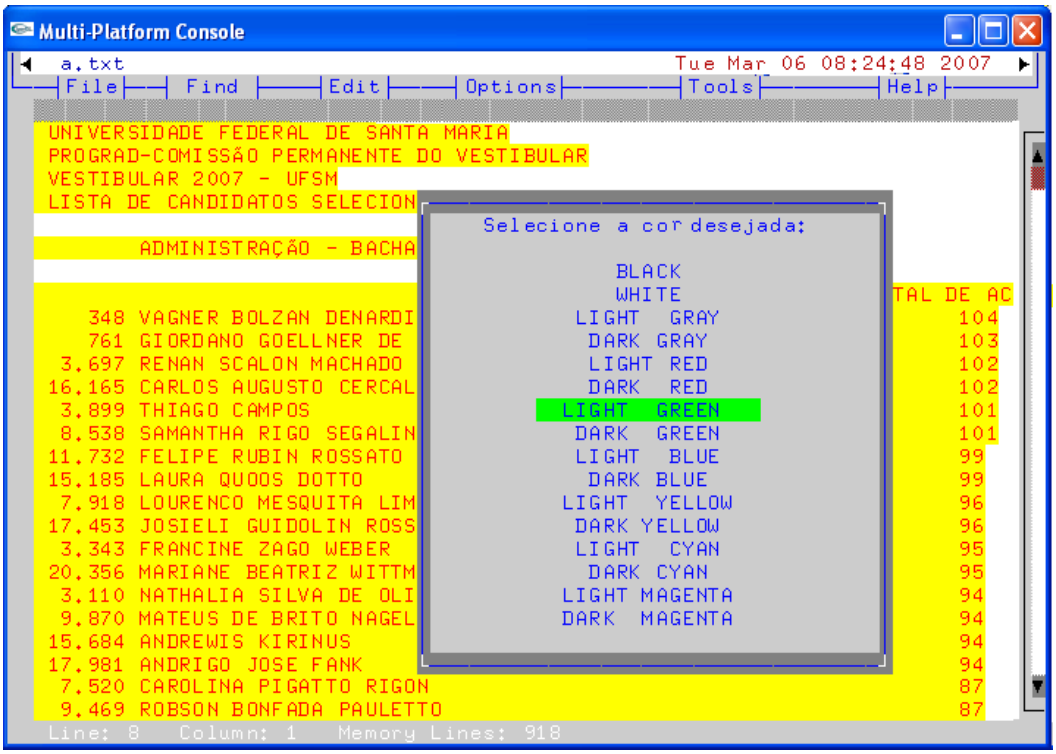
Exemplos de Uso

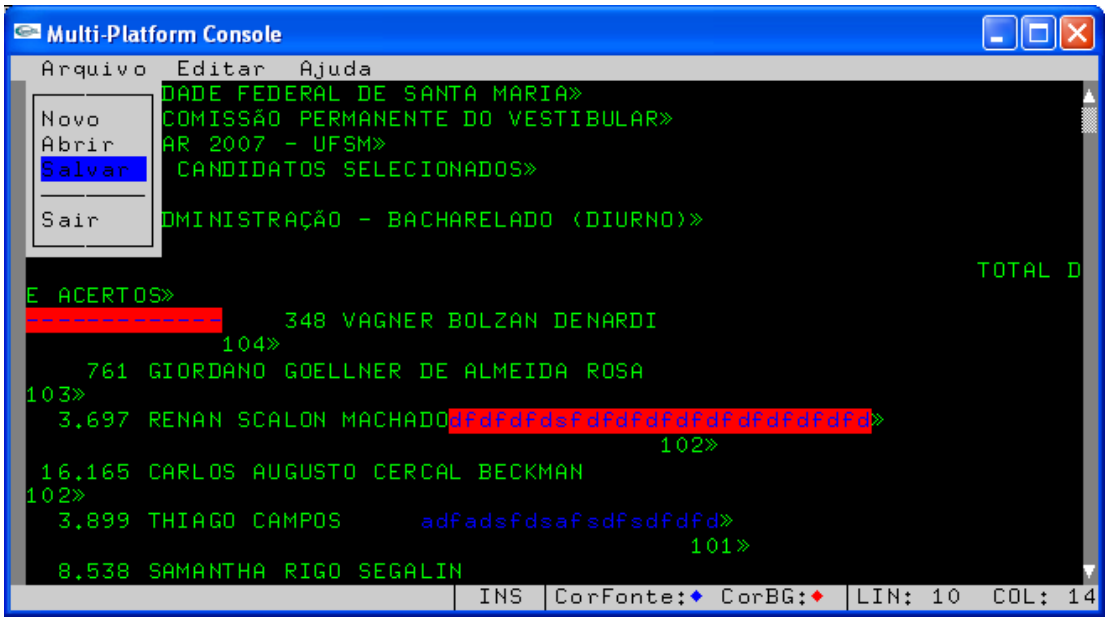
As seguintes figuras ilustram alguns aplicativos (editores de texto, planilhas eletrônicas, gerenciadores de informações, browser e explorer) já desenvolvidos com a API, na disciplina de Estruturas de Dados da UFSM. Estas figuras ilustram as potencialidades da ferramenta na criação de interfaces e *widgets* gráficos.

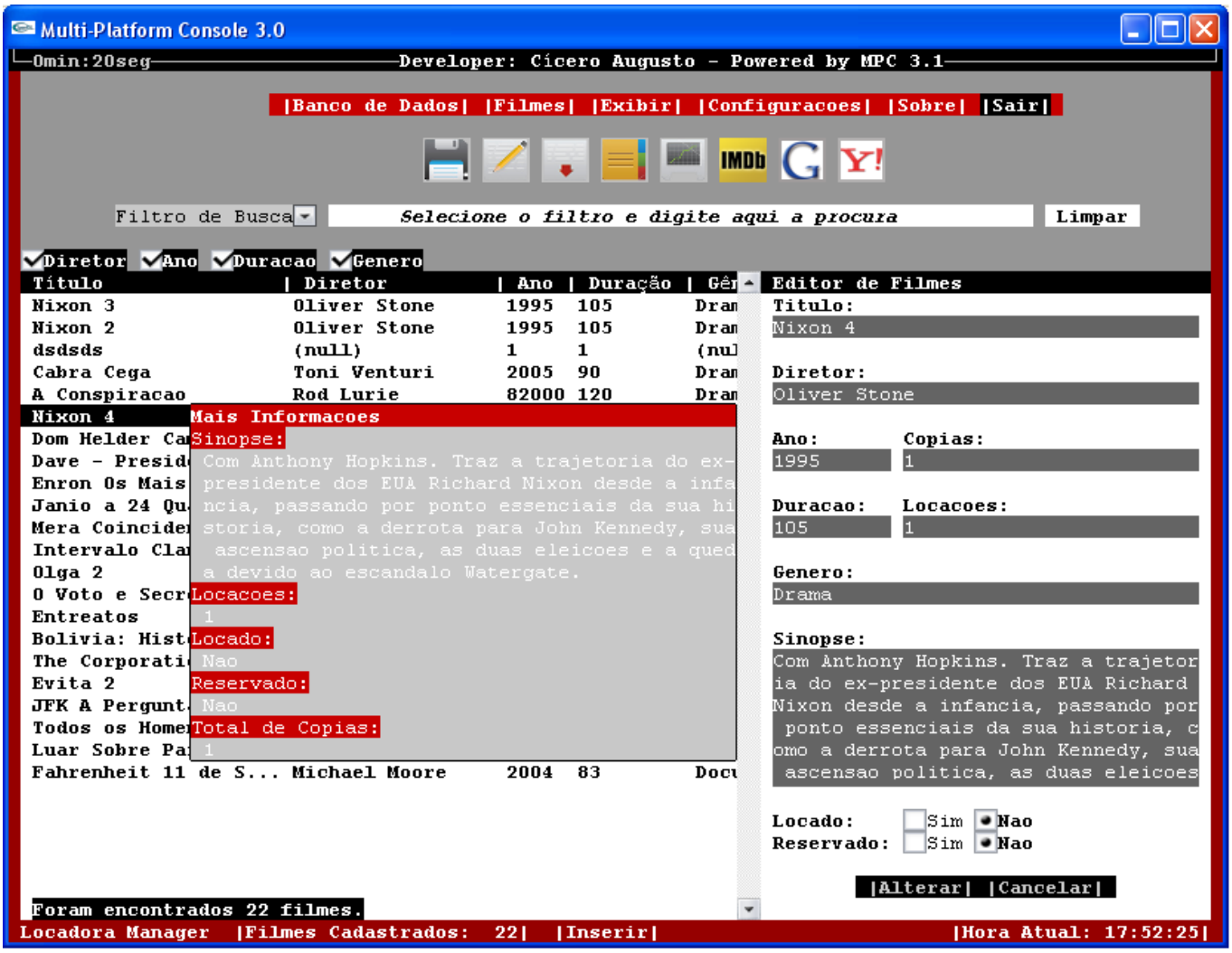

Multi-Platform Console
Arquivo Editar
New
2000.5
*sem titulo
A B C D E F
1
2 custo Saldo
3 90
4 23.9
5 2000.5
6
7
8
9
10
11

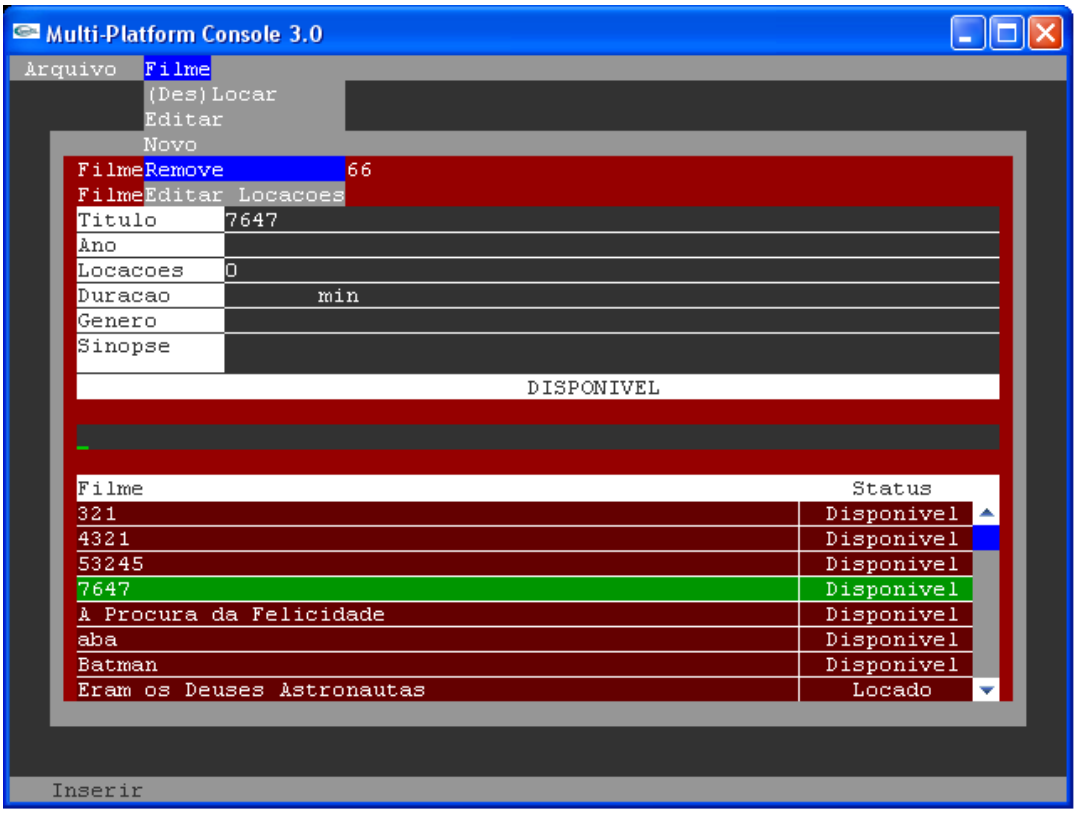

Multi-Platform Console
T6 - Vinicius Garcia Pinto CC UFSM 2007 - P1
>>Arquivo >>Editar >>Sobre
fx :
>>> Célula: @0
Copiar
Colar
Limpar
A C D E
1
2
3
4
5
6
7
F C B
|@0 |Col:@ - Lin:0 | Arquivo:>>> Sem_Título.mpc <<< |

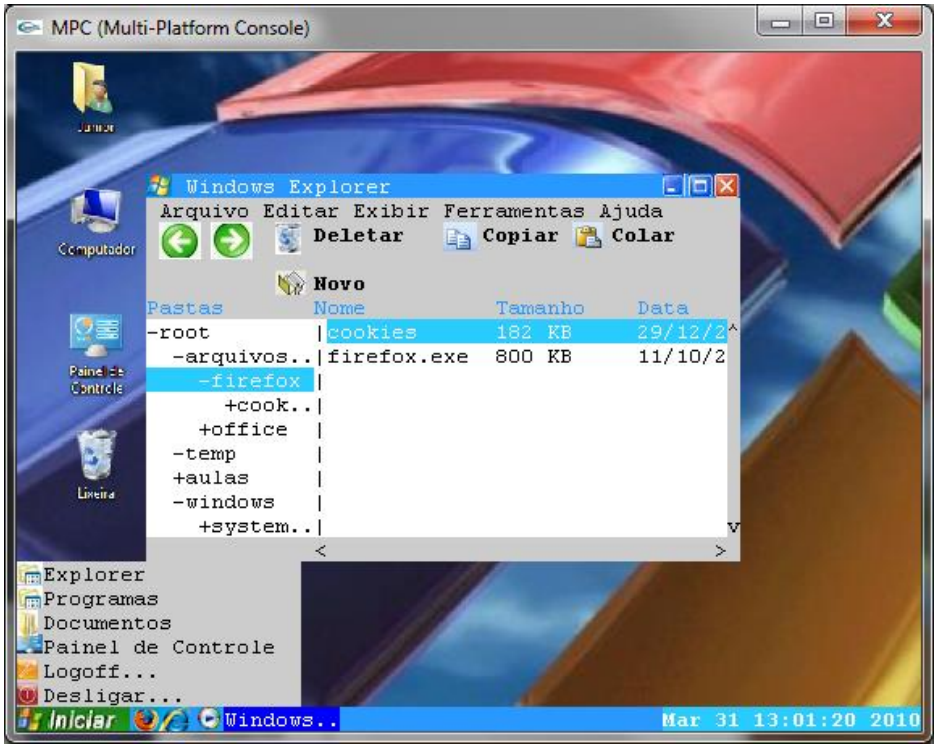

Multi-Platform Console
.Arquivo | .Editar | .Ajuda
saldo
Ir para
Localizar
B0 C D E F G
0 custo
1
2
3
4
Localizar
Digite a palavra a ser localizada
<OK> <CANCEL>

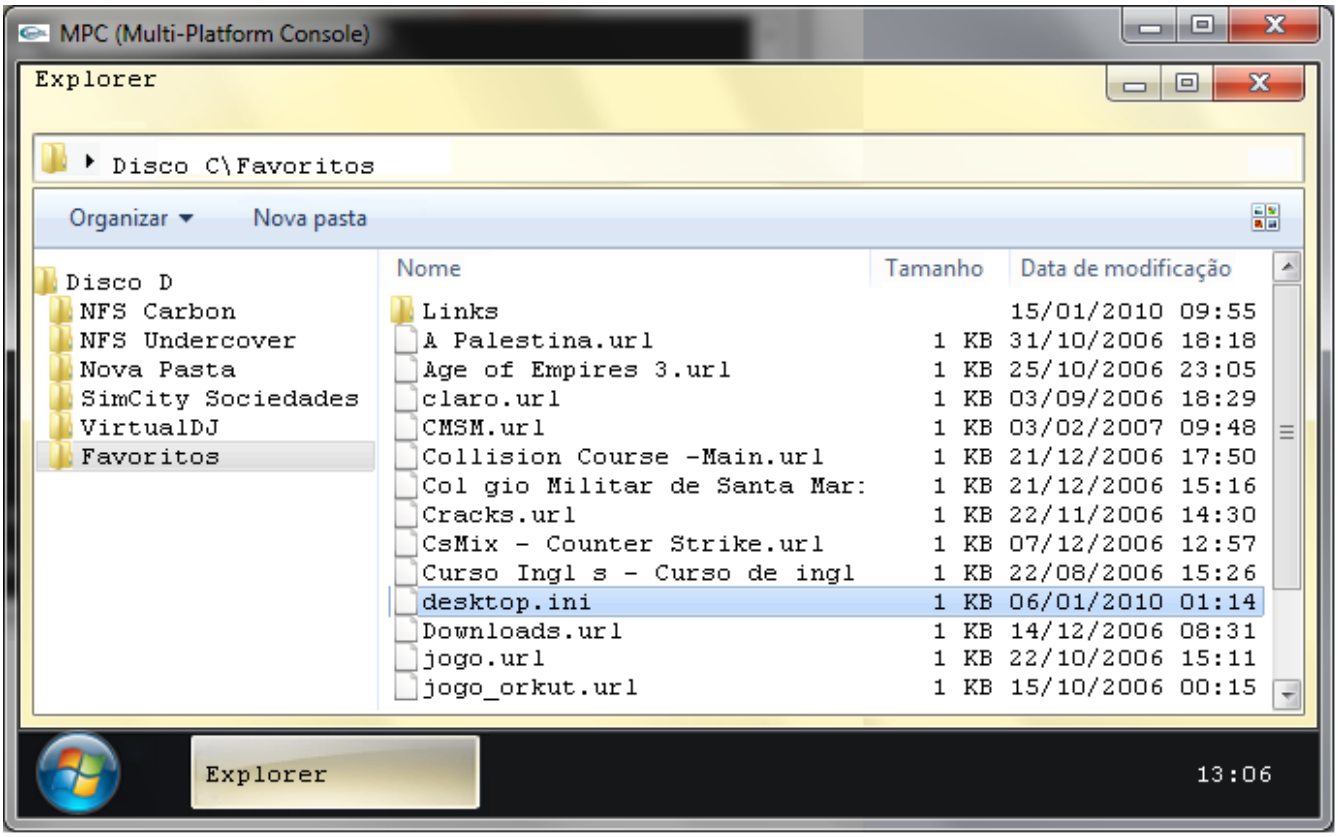

Multi-Platform Console
a.txt
Tue Mar 06 08:24:48 2007
File
Find
Edit
Options
Tools
Help
UNIVERSIDADE FEDERAL DE SANTA MARIA
PROGRAD-COMISSÃO PERMANENTE DO VESTIBULAR
VESTIBULAR 2007 - UFSM
LISTA DE CANDIDATOS SELECION
ADMINISTRAÇÃO - BACHA
TAL DE AC
348 VAGNER BOLZAN DENARDI 104
761 GIORDANO GOELLNER DE 103
3.697 RENAN SCALON MACHADO 102
16.165 CARLOS AUGUSTO CERCAL 102
3.899 THIAGO CAMPOS 101
8.538 SAMANTHA RIGO SEGALIN 101
11.732 FELIPE RUBIN ROSSATO 99
15.185 LAURA QUOOS DOTTO 99
7.918 LOURENCO MESQUITA LIM 96
17.453 JOSIELI GUIDOLIN ROSS 96
3.343 FRANCINE ZAGO WEBER 95
20.356 MARIANE BEATRIZ WITTM 95
3.110 NATHALIA SILVA DE OLI 94
9.870 MATEUS DE BRITO NAGEL 94
15.684 ANDREWIS KIRINUS 94
17.981 ANDRIGO JOSE FANK 94
7.520 CAROLINA PIGATTO RIGON 87
9.469 ROBSON BONFADA PAULETTO 87
Line: 8 Column: 1 Memory Lines: 918
Selecione a cor desejada:
BLACK
WHITE
LIGHT GRAY
DARK GRAY
LIGHT RED
DARK RED
LIGHT GREEN
DARK GREEN
LIGHT BLUE
DARK BLUE
LIGHT YELLOW
DARK YELLOW
LIGHT CYAN
DARK CYAN
LIGHT MAGENTA
DARK MAGENTA


Multi-Platform Console
Arquivo Editar Ajuda
Novo
Abrir
Salvar
Sair
DADE FEDERAL DE SANTA MARIA»
COMISSÃO PERMANENTE DO VESTIBULAR»
AR 2007 - UFSM»
CANDIDATOS SELECIONADOS»
DMINISTRAÇÃO - BACHARELADO (DIURNO)»
TOTAL D
E ACERTOS»
348 VAGNER BOLZAN DENARDI
104»
761 GIORDANO GOELLNER DE ALMEIDA ROSA
103»
3.697 RENAN SCALON MACHADOdfdfdfdsfdfdfdfdfdfdfdfdfd»
102»
16.165 CARLOS AUGUSTO CERCAL BECKMAN
102»
3.899 THIAGO CAMPOS adfadsfdsafsdfsdfdfd»
101»
8.538 SAMANTHA RIGO SEGALIN
INS
CorFonte: CorBG:
LIN: 10 COL: 14


Multi-Platform Console 3.0
0min:20seg
Developer: Cícero Augusto - Powered by MPC 3.1
|Banco de Dados| |Filmes| |Exibir| |Configuracoes| |Sobre| |Sair|
IMDb
Filtro de Busca
Selecione o filtro e digite aqui a procura
Limpar
Diretor Ano Duracao Genero
Titulo | Diretor | Ano | Duração | Gêr
Nixon 3 Oliver Stone 1995 105 Dran
Nixon 2 Oliver Stone 1995 105 Dran
dsdsds (null) 1 1 (nul
Cabra Cega Toni Venturi 2005 90 Dran
A Conspiracao Rod Lurie 82000 120 Dran
Nixon 4
Dom Helder Ca
Dave - Presid
Enron Os Mais
Janio a 24 Qu
Mera Coincide
Intervalo Cla
Olga 2
O Voto e Secr
Entreatos
Bolivia: Hist
The Corporati
Evita 2
JFK A Pergunt
Todos os Home
Luar Sobre Pa
Fahrenheit 11 de S... Michael Moore 2004 83 Docu
Mais Informacoes
Sinopse:
Com Anthony Hopkins. Traz a trajetoria do ex-
presidente dos EUA Richard Nixon desde a infa
ncia, passando por ponto essenciais da sua hi
storia, como a derrota para John Kennedy, sua
ascensao politica, as duas eleicoes e a qued
a devido ao escandalo Watergate.
Locacoes:
1
Locado:
Nao
Reservado:
Nao
Total de Copias:
1
Editor de Filmes
Titulo:
Nixon 4
Diretor:
Oliver Stone
Ano:
1995
Copias:
1
Duracao:
105
Locacoes:
1
Genero:
Drama
Sinopse:
Com Anthony Hopkins. Traz a trajetor
ia do ex-presidente dos EUA Richard
Nixon desde a infancia, passando por
ponto essenciais da sua historia, c
omo a derrota para John Kennedy, sua
ascensao politica, as duas eleicoes
Locado: Sim Nao
Reservado: Sim Nao
|Alterar| |Cancelar|
Foram encontrados 22 filmes.
Locadora Manager |Filmes Cadastrados: 22| |Inserir| |Hora Atual: 17:52:25|


Multi-Platform Console 3.0
Arquivo
Filme
(Des)Locar
Editar
Novo
FilmeRemove
66
FilmeEditar Locacoes
Titulo 7647
Ano
Locacoes 0
Duracao min
Genero
Sinopse
DISPONIVEL
Filme Status
321 Disponivel
4321 Disponivel
53245 Disponivel
7647 Disponivel
A Procura da Felicidade Disponivel
aba Disponivel
Batman Disponivel
Eram os Deuses Astronautas Locado
Inserir


MPC (Multi-Platform Console)
Windows Explorer
Arquivo Editar Exibir Ferramentas Ajuda
Deletar
Copiar
Colar
Novo
Pastas
Nome
Tamanho
Data
-root
-arquivos..
-firefox
+cook..
+office
-temp
+aulas
-windows
+system..
cookies 182 KB 29/12/2
firefox.exe 800 KB 11/10/2
Computador
Painel de Controle
Lixeira
Explorer
Programas
Documentos
Painel de Controle
Logoff...
Desligar...
Iniciar
Windows..
Mar 31 13:01:20 2010


MPC (Multi-Platform Console)
Explorer
Disco C\Favoritos
Organizar
Nova pasta
Disco D
NFS Carbon
NFS Undercover
Nova Pasta
SimCity Sociedades
VirtualDJ
Favoritos
Nome
Tamanho
Data de modificação
Links 15/01/2010 09:55
A Palestina.url 1 KB 31/10/2006 18:18
Age of Empires 3.url 1 KB 25/10/2006 23:05
claro.url 1 KB 03/09/2006 18:29
CMSM.url 1 KB 03/02/2007 09:48
Collision Course -Main.url 1 KB 21/12/2006 17:50
Col gio Militar de Santa Mar: 1 KB 21/12/2006 15:16
Cracks.url 1 KB 22/11/2006 14:30
CsMix - Counter Strike.url 1 KB 07/12/2006 12:57
Curso Ingl s - Curso de ingl 1 KB 22/08/2006 15:26
desktop.ini 1 KB 06/01/2010 01:14
Downloads.url 1 KB 14/12/2006 08:31
jogo.url 1 KB 22/10/2006 15:11
jogo_orkut.url 1 KB 15/10/2006 00:15
Explorer
13:06

Funções Básicas do MPC

A função `main()` da aplicação que faz uso da API MPC deve realizar duas tarefas: inicializar o MPC/Aplicação e chamar a função `mpcRun()`. Na inicialização do MPC deve-se primeiramente (e obrigatoriamente) informar o tamanho da janela a ser criada, em caracteres, com a função `mpcSetSize()`. Esta função pode ser chamada uma única vez, a aceita como parâmetros mínimos [10x50] e máximos [68x128]. Caso a aplicação usar gráficos ou figuras, deve-se obrigatoriamente chamar a funcão `mpcSetClippingArea()`. Pode-se também definir callbacks e parâmetros de cursor. Após inicializado o MPC, deve-se chamar a função `mpcRun()`, que passa o controle da aplicação para o MPC. Todas as funções do MPC iniciam com o prefixo `mpc`.

```
int main()
{
   init_Mpc();

   mpcRun(30); //frames por segundo

   return 0;
}

void init_Mpc()
{
   mpcSetSize(APP_LINES, APP_COLUMNS);
   mpcSetCursorPos(cursorL, cursorC);
   mpcSetCursorColor(GREEN_4);
   mpcSetCursorVisible(1);

   //MPC callbacks
   mpcSetMouseFunc(cb_mouse);       //callback de mouse
   mpcSetUpdateFunc(cb_update);     //callback de desenho
   mpcSetKeyboardFunc(cb_keyboard); //callback de teclado
}
```

A função `mpcSetChar()` é responsável por setar um determinado caractere numa determinada área do console. Os parâmetros são: posição do caractere (linha e coluna), o tipo de fonte (normal, negrito, itálico, sublinhado e combinações), cor do texto, cor do background do caractere e transparência.

```
void mpcSetSize(int lin, int col);

void mpcSetChar(int lin, int col, int charac, int font,
                int r_t, int g_t, int b_t,  //RGB text color
                int r_b, int g_b, int b_b,  //RGB background color
                double alpha);              //transparencia [0,1]

void mpcRun(30); //frames por segundo
```

Para a especificação da cor pode-se utilizar combinações de RGB ou utilizar as constantes definidas no arquivo definitions.h.

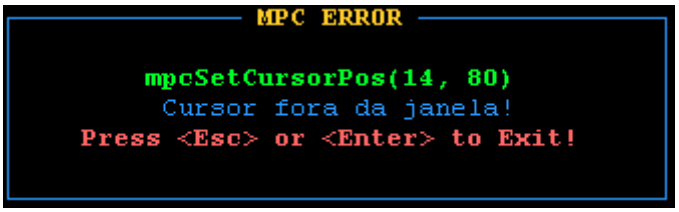
```
void mpcSetChar(11,10, 'b', F_STD, CYAN_2,  GREEN_1,    0.5); //50% transparente
void mpcSetChar(12,10, 'c', F_IS,  0,0,200, 200,10,100,   1); //sem transparencia
```

Funções de Cursor do MPC

O cursor é um recurso que pode ser usado para auxiliar o usuário na digitação de textos, por exemplo. Existem 3 funções para especificação do cursor. Pode-se definir a sua posição, sua cor e se deve ser exibido ou não. O controle de piscada (*blink*) fica a cargo do programador. Para isso, use a callback `update()`.

```
void mpcSetCursorColor(int r, int g, int b);
void mpcSetCursorPos  (int lin, int col);
void mpcSetCursorVisible(bool boolean);
```

Caso o cursor for posicionado fora da janela do MPC (coordenada maior que as dimensões em caracteres da janela), um erro é indicado e o aplicativo deve ser reinicializado. Cabe ao programador fazer o controle da posição do cursor.



Funções de Callbacks do MPC

As callbacks são a forma como o usuário interage com uma aplicação. Na funcão `init_App_and_Mpc()` o programador deve registrar no MPC as callbacks que forem necessárias. São oferecidas callbacks de teclado, de mouse e uma de atualização de tela. Toda vez que uma tecla for pressionada ou liberada, a callback `keyboard()` é chamada. Todos os eventos de mouse são trados pela callback `mouse()`. A callback `update()` é chamada continuamente, a cada atualização de tela, independente de eventos ocorrerem ou não.

```
void mpcSetKeyboardFunc(void(*func)(int key, int modifier, bool special, bool up));
void mpcSetMouseFunc(void(*func)(int lin, int col, int button, int state));
void mpcSetUpdateFunc(void(*func)());
```

Funções de Linhas do MPC

Adicionalmente ao recurso de exibição de caracteres, o MPC oferece recursos para desenho de linhas verticais (`mpcVLine()`) ou horizontais (`mpcHLine()`). Elas podem ser úteis na montagem da interface gráfica, como pode ser visto nos exemplos de uso. Para desenhar uma linha, deve-se informar as coordenadas cartesianas iniciais e finais em pixels, além da cor.

```
void mpcVLine(int x, int yini, int yfim, int r, int g, int b, double alpha);
void mpcHLine(int y, int xini, int xfim, int r, int g, int b, double alpha);
```

Funções de Figuras e Imagens do MPC

O MPC permite a manipulação de imagens/gráficos. O MPC oferece recursos para carregamento de imagens em formato BMP, de 24 bits/pixel, sem compressão.

Pode-se também gerar gráficos/tabelas. Tanto gráficos como imagens são tratados como buffers gráficos, que podem ser manipulados livremente. O buffer é representado como um vetor do tipo `unsigned char`, onde sequências de 3 bytes consecutivos definem a cor RGB de cada pixel. Cada buffer tem atributos como largura, altura, ID sequencial e espaço ocupado em memória. Várias funções fazem uso do ID para se referenciarem a figuras específicas. Cabe ao programador gerenciar os IDs. Se for passado como argumento um ID inválido para qualquer função de imagem será gerada uma mensagem de erro.

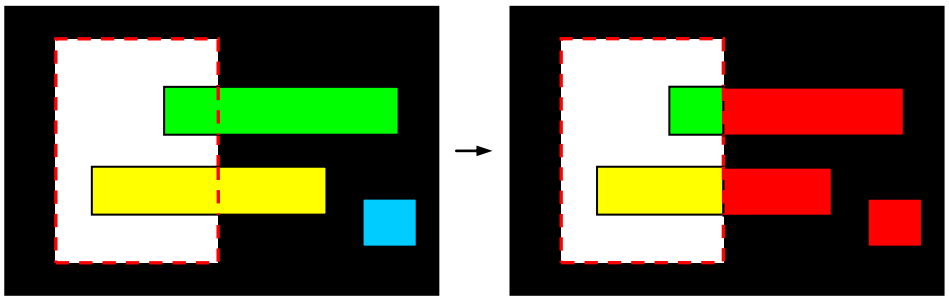
Ao contrário das linhas, cuja posição é dada em coordenadas cartesianas, as figuras são posicionadas em relação aos caracteres, pela função `mpcShowImg()`. Com esta função pode-se exibir a mesma imagem em vários locais da tela simultaneamente. Esse recurso pode ser usado para geração de interfaces gráficas mais avançadas ou na implementação de browsers, por exemplo.

```
void   mpcSetClippingArea(int initLin, int initCol, int numLin, int numCol );
int    mpcLoadBmp(char *path);
int    mpcLoadBmp char *path, int r, int g, int b);
void   mpcShowImg(int lin, int col, int id, double alpha);

void   mpcSetImgIdVisible(int boolean);
void   mpcDestroyImg(int id );
int    mpcBuildGraph(int *vet, int num, const char *titulo, const char* eixoY,
                     const char*legenda[], int size);
int    mpcCreateImg(unsigned char *p, int width, int height);
uchar* mpcGetImg(int id);
int    mpcGetImgHeight(int id);
int    mpcGetImgWidth(int id);
int    mpcGetImgSize(int id);
```

Para melhor tratar texto e imagens simultaneamente, o MPC tem a função `mpcSetClippingArea()`, que permite definir uma área de recorte na qual a figura pode ser exibida. Desta forma, toda figura deve ser inserida dentro da área de recorte, pois caso contrário não será visualizada. As coordenas da área de recorte são especificadas em coordenadas de caracteres.

No seguinte exemplo ilustra-se o funcionamento da área de recorte (definida em branco). Todas as figuras ou partes que estiverem fora são eleminados ou recortados durante a exibição (partes vermelhas). Com este recurso, pode-se, por exemplo, evitar que uma figura inserida em um editor de texto seja desenhada sobre a barra de menus.



No comando `mpcSetClippingArea(2,3,40,75)` define-se que a área de recorte começa a partir da 2ª linha e 3ª coluna. A área de recorte avança 40 linhas para baixo, atingindo a linha 42 como limite e 75 colunas da direita para a esquerda, atingindo a coluna 78 como limite.

As funções `mpcGetImgHeight()` e `mpcGetImgWidth()` são responsáveis por retornar um inteiro representando a altura/largura da imagem em pixels. Deve-se passar como parâmetro um inteiro que representa o ID da figura. Podem ser usadas para descobrir o tamanho de uma figura carregada do arquivo, para ajustar a sua posição dentro de um documento, como por exemplo, dentro de um editor de texto ou browser.

A função `mpcGetImgSize()` é responsável por retornar um inteiro representando o tamanho total do buffer da imagem, que é dado pela fórmula altura x largura x 3.

A função `mpcGetImg()` retorna o buffer de dados de uma imagem (array de pixels em formato RGB no padrão 888 – 8 bits para cada componente). Pode ser usada para fazer a replicação de uma imagem para posterior edição. A função `mpcCreateImg()` cria uma imagem dentro do MPC (função avançada – o programador pode necessitar de conhecimento em computação gráfica). Deve-se fornecer as dimensões e o array de pixels em formato RGB. Com essa função pode-se criar gráficos personalizados (ver função `mpcBuildGraph`) ou qualquer outro tipo de figura. A função retorna o ID da figura.

Com a função `mpcLoadBmp()` pode-se carregar uma imagem de um aquivo no formato BMP para a memória do MPC. Deve se passar como parâmetro uma string que corresponde ao nome e path da figura. Opcionalmente pode-se especificar a cor da imagem que será tratada como transparente. O ID retornado deve ser usado para se referenciar a esta figura. Se a imagem não puder ser carregada, a função retorna um ID com valor -1.

Para posicionar uma imgem na tela do MPC deve-se utilizar a função `mpcShowImg()`. Deve-se informar a linha e coluna iniciais, além do ID da figura. Pode-se exibir além da figura o seu respectivo ID, com a função `mpcSetImgIdVisible()`. Esse recurso é importante para o usuário gerenciar figuras em seu documento, especialmente quando existem várias figuras abertas simultaneamente. Sabendo o ID de uma figura, ele pode invocar uma caixa de diálogo por onde possa digitar o ID de uma figura que deseja interagir. Tendo-se o ID, pode-se mover a figura, ocultá-la ou até mesmo removê-la da memória do MPC (`mpcDestroyImg()`).

A função `mpcBuildGraph()` é responsável por criar um gráfico e alocá-lo na memória. Esse gráfico é criado na forma de uma imagem BMP. Para que o gráfico possa ser exibido ele deve ser tratado como uma imagem comum, ou seja, só poderá ser exibido caso esteja contido dentro da área de recorte. Deve-se passar como parâmetros um ponteiro para um vetor que representa o valor de cada barra a ser desenhada, um inteiro que representa o número total de barras, uma string que representa o título que ficará superior ao gráfico, uma string que representa o eixo Y, um vetor que armazena n ponteiros para strings que representam o texto contido na legenda e por fim, o tamanho do gráfico. Esta função retorna um inteiro referente ao id da imagem carregada para a memória. O número de casas correspondentes ao vetor vet, o número da variável num e o de n de ponteiros para strings da legenda devem ser iguais para que não haja conflitos. A função retorna o ID da figura.