# A implementação

Antes mesmo de iniciar o nosso relato sobre a implementação utilizada é importantíssimo frisar que a tecnologia utilizada está constantemente em evolução, sendo assim muitas diferenças da tecnologia atual podem ser verificadas. Relataremos brevemente sobre a tecnologia utilizada afim de guiar o leitor. Para edição de códigos o eclipse Europa versão 3.3.2 foi utilizado. A api javaTV1[1] versão 1.0 deu o suporte ao uso da tecnologia de TV digital para JAVA. Para visualização das xlets usou-se o emulador xletview versão 0.36. Para edição de imagens foi usado o Photoshop CS3.

A príncipio foi estipulado que o desenvolvimento do programa seria do tipo bottom-up, porém como a tecnologia utilizada é muito nova e a maior parte do programa está concentrada na parte gráfica visto que é um programa televisual, passou a ser desenvolvida de forma top-down. Em exceção a parte de persistência, que foi desenvolvido primeiro, pois a parte mais importante do programa é seu banco, contendo as informações necessárias a serem repassadas para o usuário. A tecnologia do emulador não suporta ferramentas como derby e hibernate que facilitariam a criação de bancos, era necessário então criar uma forma de criar um banco antes mesmo de começar a trabalhar com o projeto, esta forma foi lidar com arquivos de entrada e saída. As sessões posteriores são para explicar como cada parte do projeto que foi desenvolvida, a separação em itens facilita a compreensão isolada.

# Do Banco de Dados

O banco de dados do programa é dividido em duas partes: banco local e banco dinâmico. Esta divisão ocorre pois para o nosso problema sempre teremos um conjunto de informações que é fixa e outra que é atualizada a medida que se passa o evento(copa). Podemos pensar que sempre teremos os mesmos jogadores em seus times ao iniciar o programa, mas não podemos garantir que a tabela será a mesma ao iniciar o programa, pois a mesma pode sofrer alterações(resultados de jogos). Daí para o banco fixo, também chamado banco local, usamos arquivos de entrada simples com extensão “sdb”, este é um formato fictício desenvolvido apenas para o programa, com as informações requeridas. Podemos verificar um arquivo de entrada na Figura

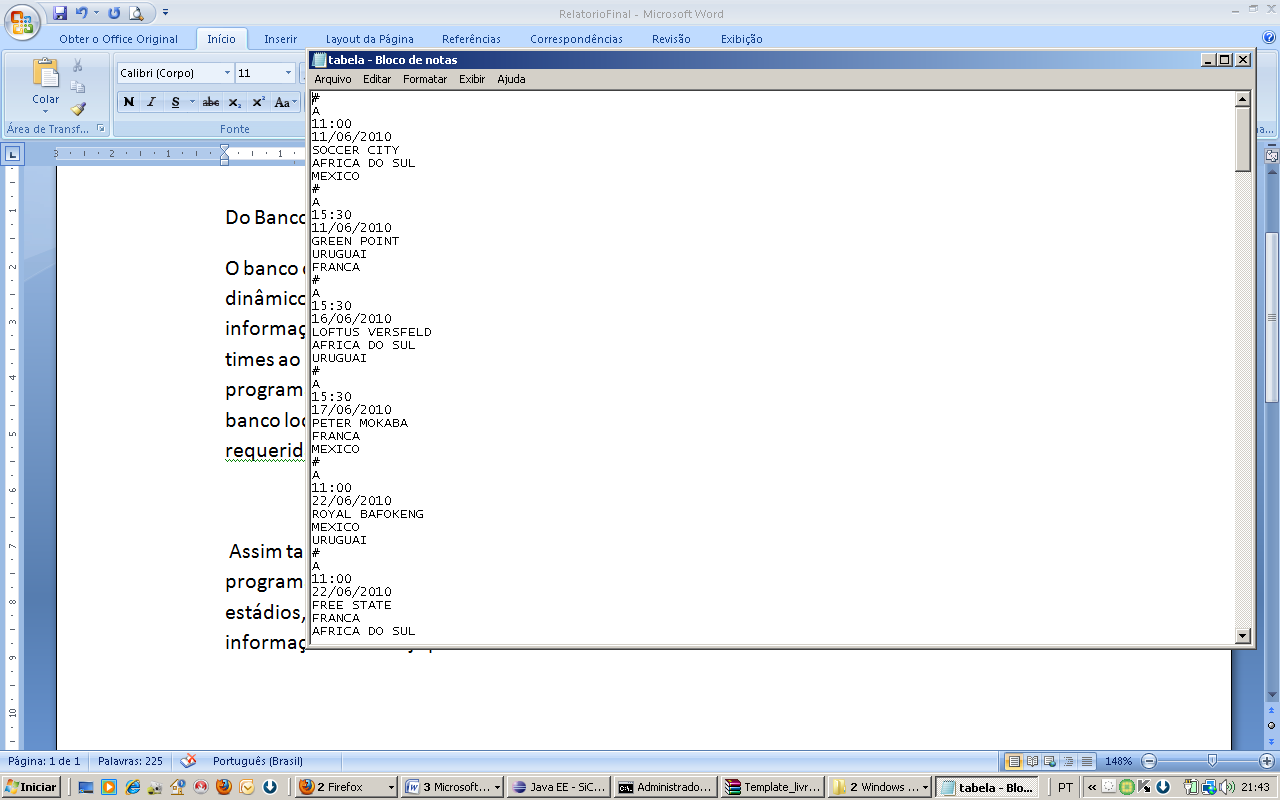
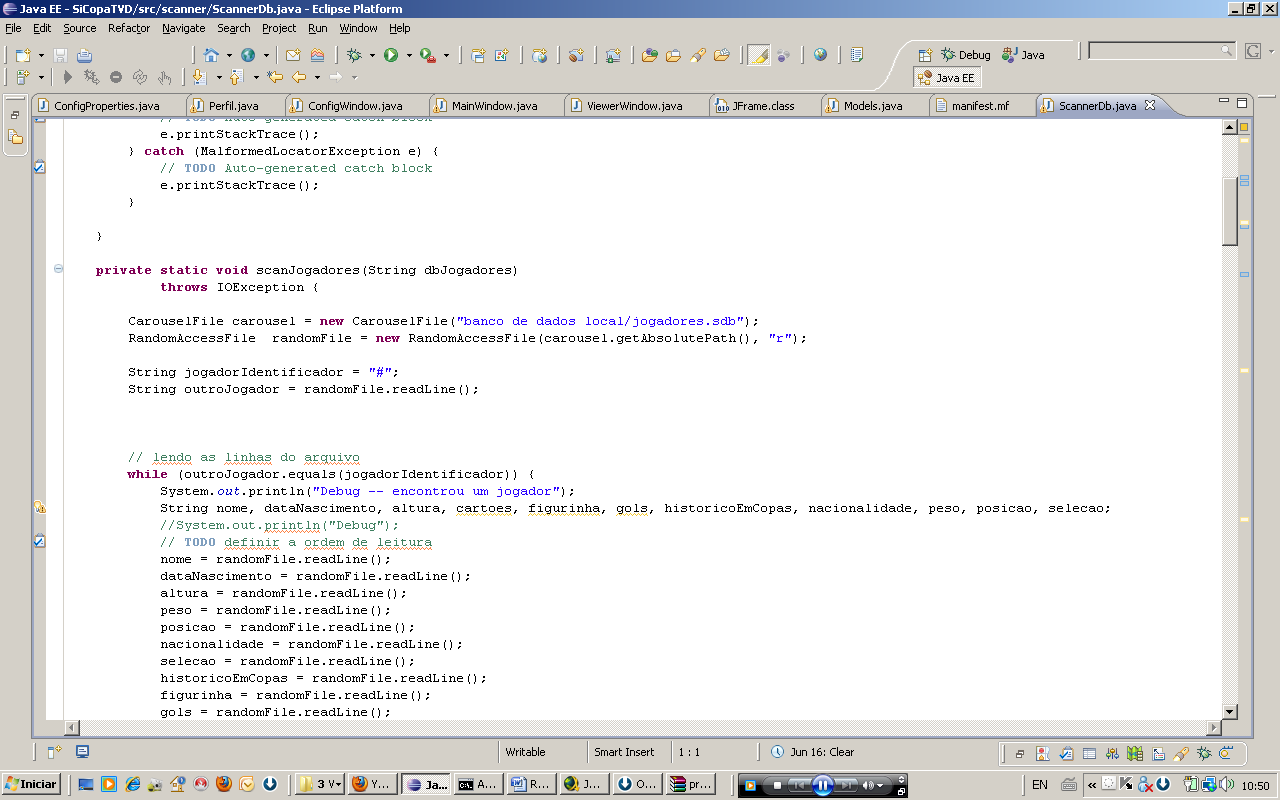


Figura .1 – Figura com parte do arquivo tabela.sdb aberta, contém informação de toda a tabela da fase de grupos, na figura está sendo mostrada apenas o grupo A.

Assim tais informações são lidas do arquivo e preenchidas em objetos específicos. Ou seja, o programa tem um objeto, que segue o padrão singleton, Copa que contem objetos times, estádios, jogadores e tabela. Então, ao iniciar o programa o banco local é preenchido, e as informações básicas já podem ser visualizadas. Se pensarmos em termos de código, o programa para acessar um objeto externo usa a idéia de carroussel [1]. Para explicar melhor, podemos ver na Figura .

Figura .2 – Trecho do método scanJogadores, pertencente a classe ScanDb com uso de objeto carroussel para escanear um arquivo de entrada.



Ao se criar um objeto do tipo CarouselFile, o mesmo tem a raiz localizada na pasta do emulador xletview e o restante do caminho é repassado como parâmetro do construtor. Ao contrário de imagens, apenas com o uso de objetos carroussel podemos acessar arquivos do tipo texto. Logo após isto, pode-se usar a classe RandomAccessFile, pertence ao pacote utilitário de Java, passando o caminho absoluto da localização do carroussel e no nosso caso “r”(apenas leitura) como parâmetros do construtor para acesso a arquivos de io.

# Da Interface Gráfica

Toda parte gráfica do programa SiCopaTVD foi desenvolvido pelo programa Photoshop CS3 desenvolvido pela Adobe. A idéia foi criar um conjunto de imagens onde o usuário navega entre figuras, que são mostradas em forma de menu e pode acessar a informação que desejar. Uma dada cena pode conter várias imagens diferentes, tornando-a mais dinâmica. O programa tem a noção de qual conjunto de imagens utilizar devido a separação em objetos, onde cada tela tem um objeto característico.

# Tela Menu Principal

Vamos falar um pouco sobre esta tela, ou classe, pois assim pode ser verificada mais claramente a forma utilizada para edição das telas. A Figura mostra a tela do menu principal, a imagem padrão é a exibida na figura.

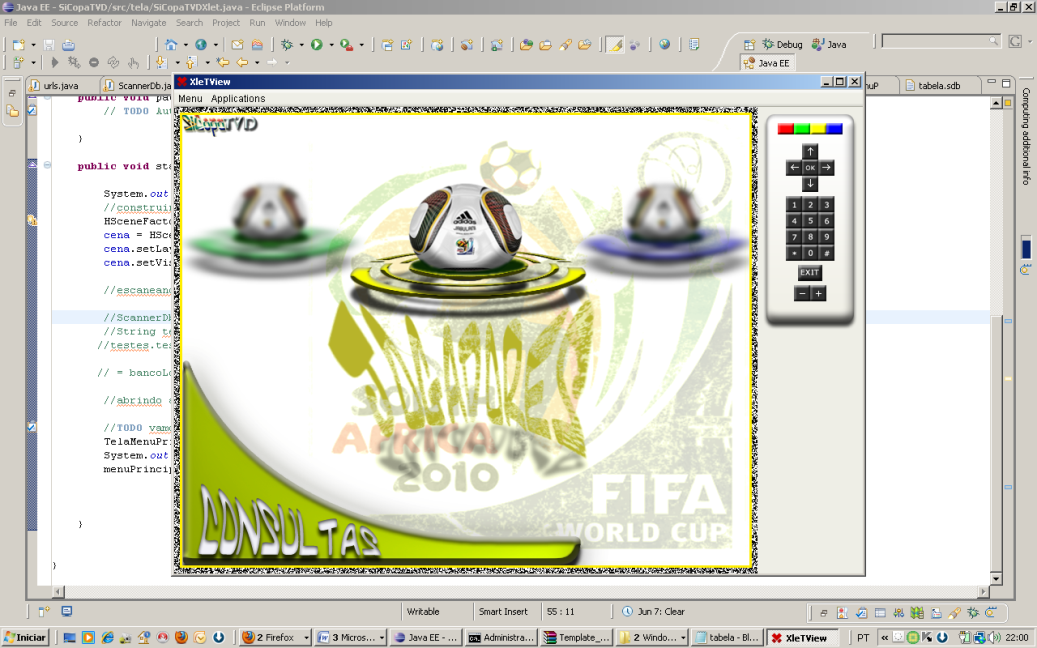


Figura .1 – Emulador XletView com o xlet do programa SiCopaTVD carregado na tela de menu principal.

O usuário pode navegar no controle remoto virtual e explorar esta tela. Abaixo, na Figura, temos o conjunto de imagens possíveis para a cena(ou tela corrente).

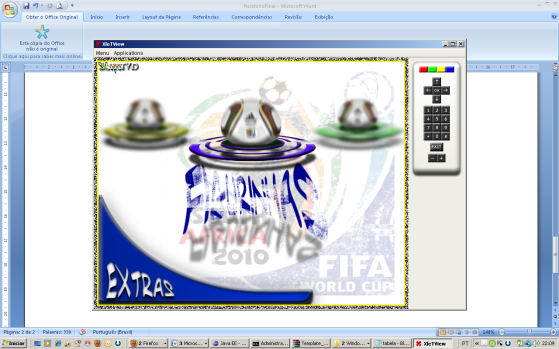
  

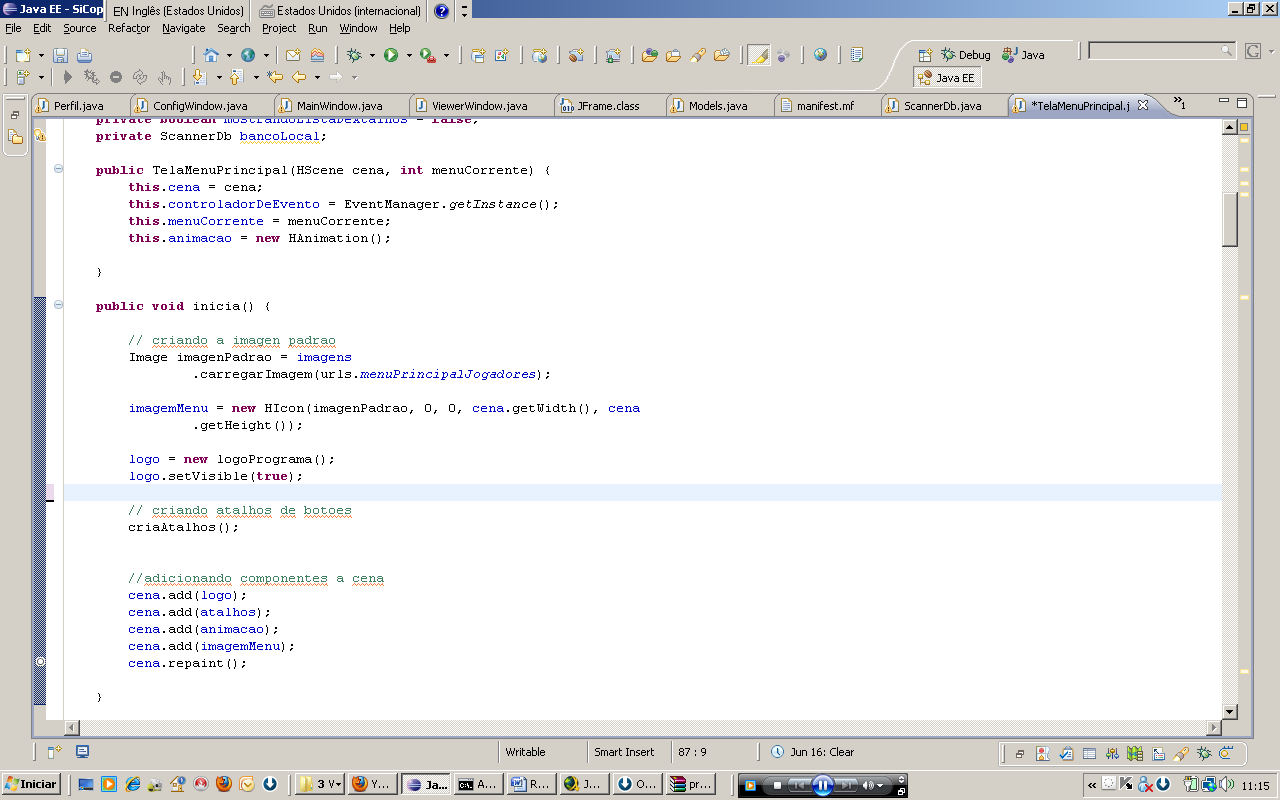
  

Figura .2 – Conjunto de imagens possíveis para a tela de menu principal, vista no emulador XletView

Como já mencionado o menu tela principal é um objeto que contem um conjunto de imagens que podem variar dentro de uma cena, dependendo da navegação do usuário. Ao invocar uma instância da classe, o conjunto de imagens é carregado á medida que é necessário, então a cena é repintada. Pensando em termos de código a cena nada mais é que um container, contendo um conjunto de elementos que constrói a imagem a ser passada. Podemos observar uma estruturação de uma tela na Figura.

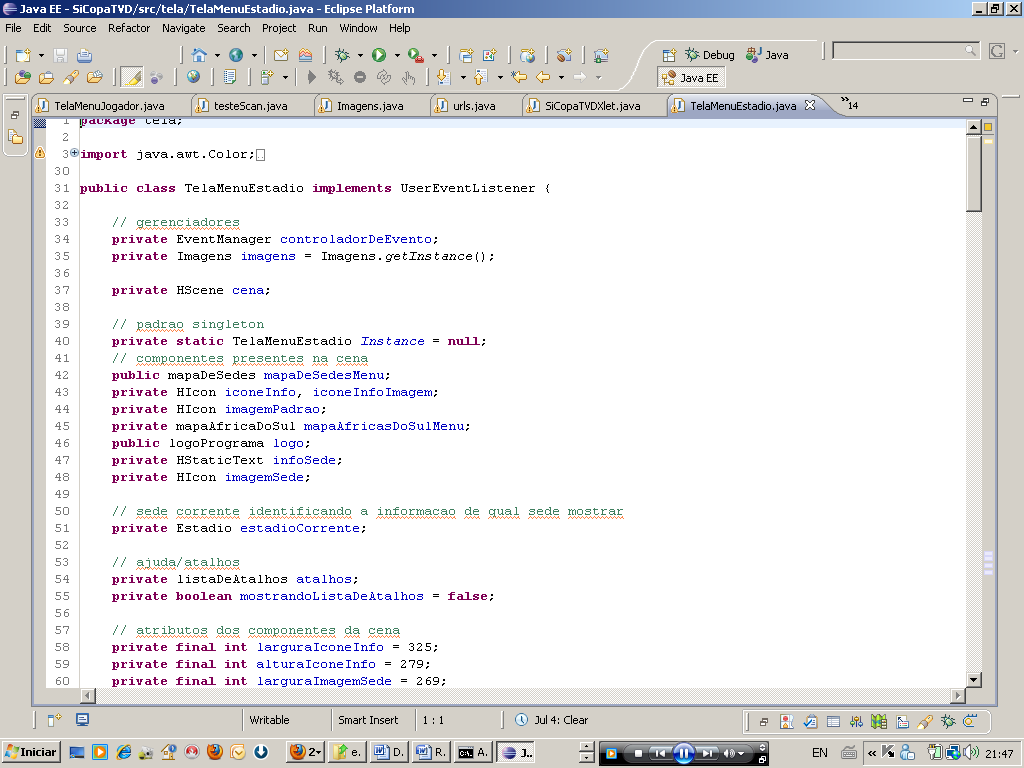
Figura .3 – Figura contendo trecho do código da classe TelaMenuPrincipal, em que é mostrado o contrutor e um método inicia



O funcionamento de toda tela, ou melhor de todo menu, é sempre criar o objeto e logo após iniciar os componentes,atributos, desse menu. Na parte de inicialização é que se faz a inserção dos componentes que participaram da cena naquele dado estado.

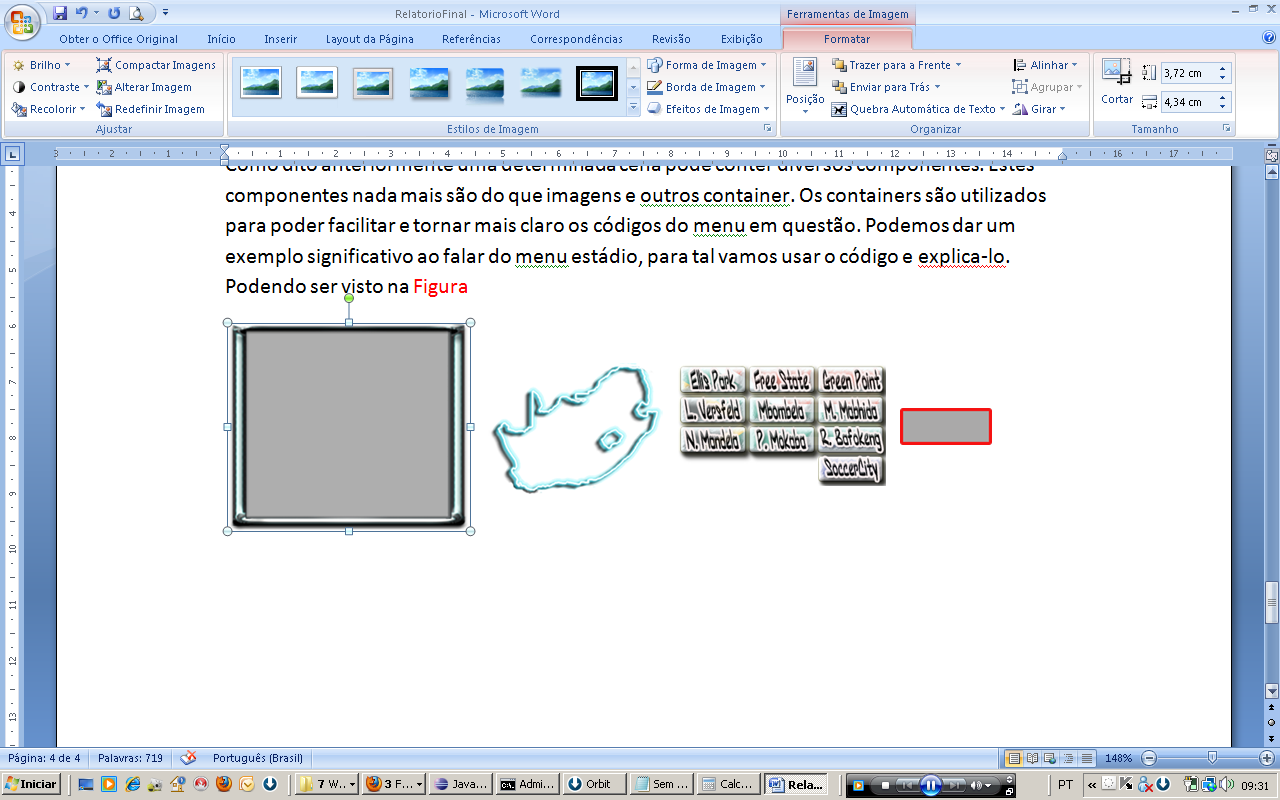
Dos componentes de uma cena

Como a cena é um container ela pode conter outros componentes(containers).Para facilitar a compreensão e a edição do código algumas cenas contém containers com funcionalidades especificas. Podemos dar um exemplo significativo ao falar do menu estádio, para tal daremos uma breve olhado em seu codigo. O mesmo pode ser visto na Figura



Na figura acima pode ser verificado que um dado objeto TelaMenuEstadio terá atributos do tipo mapaDeSedes, mapaAfricaDoSul entre outros, que são objetos container que compõe a cena do objeto TelaMenuEstadio. As partes do objeto(containers) terão um conjunto de funcionalidades que no fim deixam o código mais limpo e tomam pra si o dever sobre uma dada função. Para ser mais claro, podemos citar o caso do objeto mapaDeSedes que a medida que o usuário caminha sobre o menu ele tem o dever de mudar o cursor de lugar, ou no caso de mapaAfricaDoSul mudar o local geográfico, entre outras funções. Podemos verificar na Figura a separação do menu Estádio.

Figura .4 – Figura contendo partes(imagens) do menu tela Estadio. Tela de info, mapa da Africa do Sul, mapa das Sedes, cursor.



A montagem da cena do menu Estádio pode ser verificada na Figura.

Figura .5 – Emulador xletview o programa SiCopaTVD rodando sobre a tela de menu de estádios.

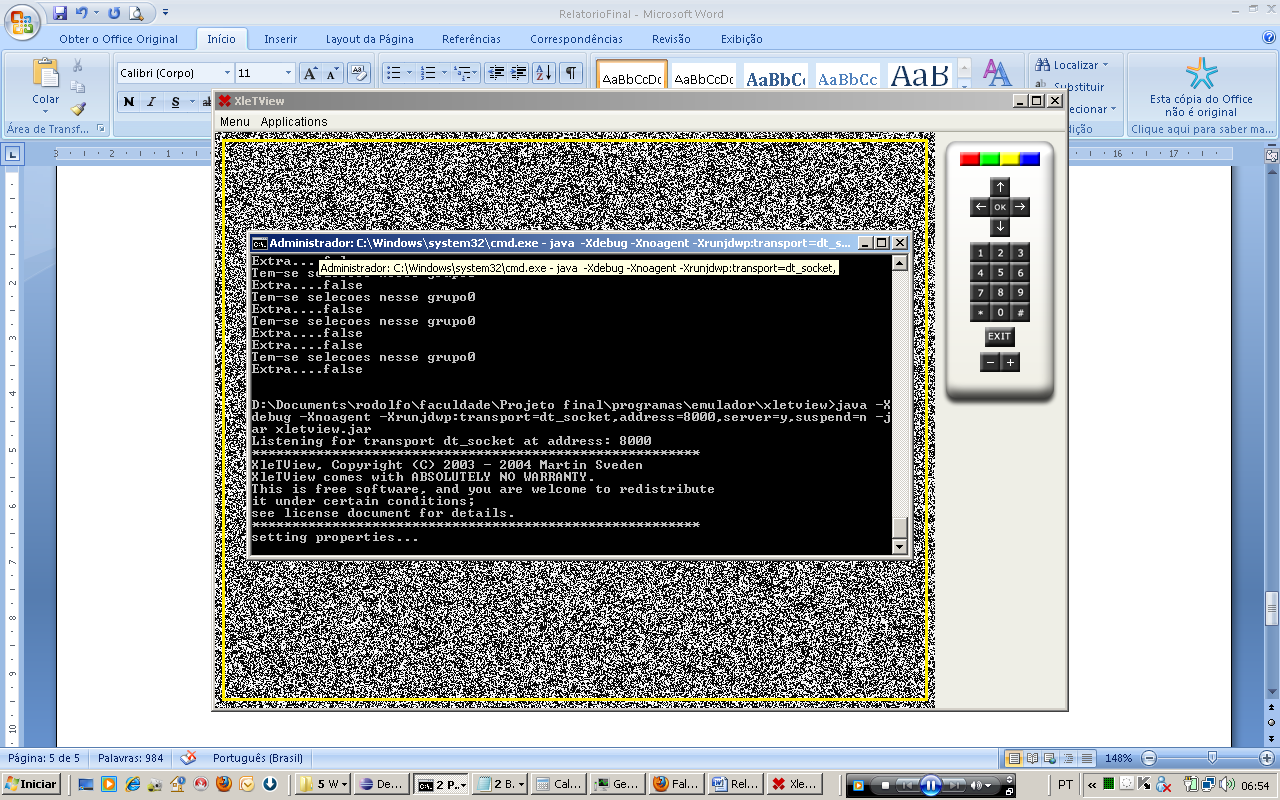


Adotando esta técnica as classes de menus tornam-se mais claras e fácil a reutilização de um de seus componentes.

# Dos testes e verificação de erros

É necessária tal seção, pois, como a plataforma utilizada para edição do código, o Eclipse, não é a mesma usada para a emulação da xlet; ou melhor, o programa eclipse não consegue rodar internamente xltes, para isto temos o emulador xletview. Para o uso de debug deve ser configurado uma porta de comunicação com o eclipse para acesso externo ao mesmo. A porta que mais foi utilizada foi a 8000. A técnica de usar o debug externo ao eclipse[2] foi muito utilizada visto a pouco conhecimento a alguns aspectos da tecnologia utilizada o que facilita muito a correção de erros. Podemos verficar na Figura o trecho de chamada do emulador pelo prompt usando o debug.

Figura .1 – Prompt de comando fazendo a chamada do emulador xletview com o uso da porta 8000 para debug



Quanto aos testes, tirando o desenvolvimento do banco, todo o resto foi testado dentro do emulador parte a parte, a medida que era construído era testado, afim de evitar uma possível dispersão numa de erros. Em especial para o banco de dados foi criado um modelo de testes que testava internamente ao eclipse o uso do banco(se tal estava sendo acessado e construído de forma correta). Esta técnica por hora foi enganosa, ao usá-la a leitura dos arquivos de entrada estava correta, porém ao transpor para o emulador não obtínhamos o resultado esperado, como dito anteriormente, o emulador não consegue acessar um arquivo io, daí a necessidade do uso de carroussel para o acesso de arquivos io. À medida que tal foi usado, os problemas de acesso foram selecionados.

# Resultados

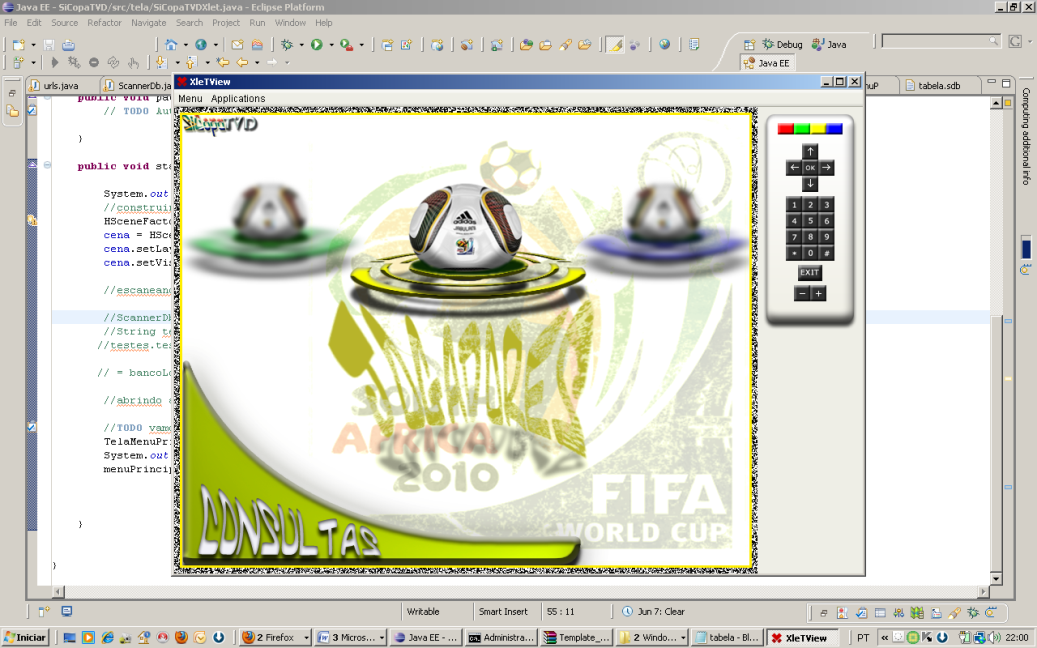
Esta seção trará os resultados sobre o programa SiCopaTVD. Os resultados não serão mostrados em forma de gráficos, ou comparativos, visto que o mesmo é apenas um programa de consulta. Por isso, para tornar o processo de compreensão dos resultados mais claro, separamos em itens. A seguir vamos falar do resultado sobre as telas, sobre o banco de dados, e as regras de negócio figurinha e moeda Lair.

# Telas

Como o programa tem um conjunto de telas o resultado sobre as mesmas está separado perante esta divisão. Nos itens será falado um pouco sobre o menu em questão e as opções do controle remoto do usuário. O importante frisar que toda tela tem uma subtela de atalhos no qual mostra quais as teclas e suas funções ativas para a dada tela, definida pela tecla \*, e a tecla vermelha também presente em muitas retorna a tela anterior.

# Tela Principal

Esta é a tela principal do programa, nesta tela o usuário escolhe qual o sub-menu deseja entrar. Pode haver navegação na mesma trocando apenas as imagens que designam qual o submenu o usuário acessará(caso aperte ok).



Atalhos:

←,→ Troca de menu

↑,↓ Troca o submenu

OK acessa o submenu

# Tela Jogadores

Tela onde contém informações dos jogadores de uma seleção escolhida na tela de seleções(tela anterior que dá acesso a esta). Se a seleção tiver mais do que onze jogadores haverá páginas de lista de jogadores.



←,→ Troca a pagina da lista de jogadores

↑,↓ Troca o jogador

# Tela de times

Tela onde contém informações sobre um dado time escolhido na tela de seleções(tela anterior). Esta tela é simples e não tem funcionalidades do controle.



# Tela de estádios

Tela que contem informações sobre os estádios presentes na copa do mundo. Nesta tela tem-se um mapa de estádios onde o usuário navega e define qual estádio visualizar as informações. Tem-se também um mapa da Africa do Sul com um cursor localizando geograficamente a posição de cada sede selecionada.



←,→ ,↑,↓ Navega entre as sedes respeitando os limites do mapa de sedes.

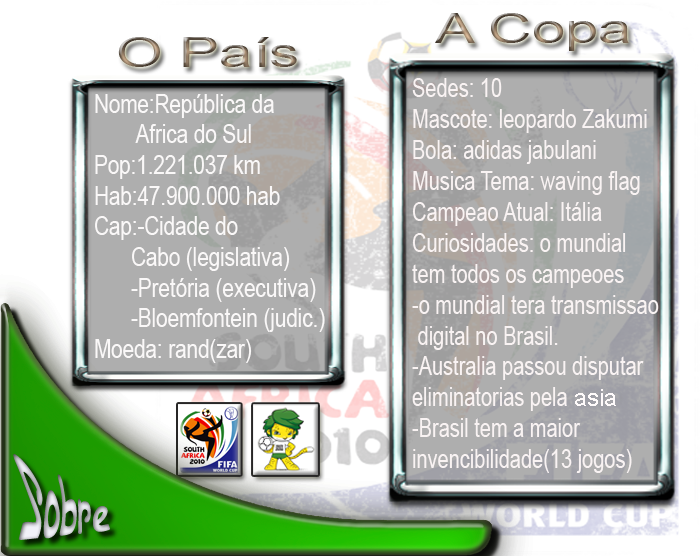
# Tela de figurinhas

Tela que exibe as figurinhas adquiridas.



# Tela de sobre

Tela que exibe informações sobre o país sede e sobre a Copa. Não há navegação no controle.



# Tela de tabela

Tela que contém informação sobre a tabela do campeonato. Nesta tela pode-se acessar a fase de grupos ou a fase eliminatória. Como o objeto carrossel não foi implementado para acesso dinâmico, a atualização dos placares também não é feita de forma dinâmica através de um certo canal.



Atalhos:

←,→ Troca o grupo quando ativo.

↑,↓ Troca a fase eliminatória quando ativa.

0 alterna entre fases grupos e eliminatórias.

# Tela de classificação

Tela e funcionalidades não implementadas.

# Tela de compra

Tela e funcionalidades não implementadas.

# Tela de usuário

Tela e funcionalidades não implementadas.

# Banco de dados

O banco de dados não foi preenchido completamente. Estádios, seleções e copa estão com todas as informações completas. Agora a parte de jogadores, que tem mais de 600 jogadores, não foi preenchido totalmente. Desses apenas quatorze do Brasil, um da Alemanha, um da África do Sul e um da Alemanha foram mapeados. Algumas informações deveriam atualizar-se dinamicamente como é o caso dos placares e da parte das figurinhas. Como não foi tratado o objeto carrossel para tal finalidade está funciona a atualização está desabilitada para essa versão, logo para o arquivo de tabela a atualização é feita manualmente e direto no arquivo.

# Figurinhas

A idéia do compartilhamento das figurinhas não está contida nesta versão, portando não tem como trocar com outros usuários qualquer tipo de figurinha. As figurinhas que são contidas nesta versão foram inseridas diretamente(manualmente) no local base do arquivo carrossel, não foram conquistadas perante a um certo número de pontos, como foi proposto.

# Moeda Lair e Usuário

O programa teria diversos usuários com uma quantidade de moedas cada um, porém devido a escassez do tempo tal funcionalidade não se encontra presente nesta versão. A parte de moedas não foi implementada e, por conseguinte nem a parte de usuários.

# Requisitos funcionais

Foi implementado os requisitos funcionais RF1, RF2, RF3, RF4, RF5 sendo que o requisito RF3 foi desmembrado em dois submenus.

# Bibliografia

[1]

[1] <http://www.interactivetvweb.org/tutorials/mhp/file_access>

[2] <http://b4dtv.blogspot.com/2009/07/tutorial-debug-no-xletview-usando-o.html>

flagpedia.net

http://www.fifa.com/worldcup/index.html

www.wikipedia.org