Revista Brasileira de Ensino de Física, v. 27, n. 4, p. 613 - 618, (2005)
www.sbfisica.org.br

Easy Java simulations – Modelagem computacional para o ensino de Física

(Easy Java simulations - Computer modelling for Physics teaching)

Jalves S. Figueira[1]

Centro Federal de Educação Tecnológica do Paraná, Pato Branco, PR, Brasil
Recebido em 9/11/2004; Revisado em 21/1/2005; Aceito em 31/8/2005

Neste trabalho é apresentada a ferramenta de *software* Easy Java Simulations-Ejs. Além de citar suas principais características e potencialidades na produção de simulações-Applets dirigidas ao ensino de Física, desenvolvem-se duas aplicações de modelagem em atividades de ensino: um sistema massa-mola e a solução numérica da Equação Schroedinger independente do tempo.
Palavras-chave: Ejs, java, ensino, modelagem, applets, aprendizagem, simulações.

This work presents the software tool Easy Java Simulations-Ejs. Besides the description of its main characteristics and potentialities in the production of Applet simulations directed to physics teaching, it also develops two modelling applications are developed: a mass-spring system and the numeric solution of the time-independent Schroedinger equation.
Keywords: Ejs, java, teaching, modelling, applets, learning, simulations.

1. Introdução

Propostas usando *softwares* de modelagem em atividades didáticas são freqüentes, inclusive nesta revista [1, 2]. Esse interesse, tanto no número de publicações como no número de *sites* envolvendo simulações baseadas em propostas pedagógicas, deve-se, às inúmeras vantagens oferecidas pelo uso dessas ferramentas no ensino, em especial no de ciências, com a utilização de laboratórios virtuais. Por outro lado, de um modo mais amplo, tais atividades só são possíveis devido ao avanço das tecnologias na área da Informática, especialmente pelo uso da linguagem de programação Java e pelo crescimento da rede *www* (*world wide web*).

A linguagem de programação Java tornou-se uma importante ferramenta para as propostas de ensino que se utilizam de ambientes virtuais. Na área das ciências, ela está por detrás dos "Laboratórios Virtuais", ambientes que simulam determinado fenômeno físico e rodam em pequenos programas, conhecidos como Applets (programas executados dentro de uma página *html*). No entanto, apesar das inúmeras vantagens relacionadas ao uso desta linguagem, a sua utilização exige um longo tempo de dedicação e mesmo a construção de um simples Applet é uma tarefa relativamente complexa para muitos.

Tendo isso em vista, foi desenvolvida a ferramenta Easy Java Simulations – Ejs, com o objetivo de simplificar e agilizar a construção de Applets e, mais especificamente, a modelagem de problemas teóricos nas áreas das ciências.

O presente trabalho apresenta o Ejs, suas características e aplicações. Além disso, cita brevemente as possibilidades e as vantagens do uso de simulações no ensino de Física e apresenta, nas últimas seções duas aplicações dessa ferramenta em atividades didáticas. Detalhes sobre a construção da interface gráfica de um Applet constituem o Apêndice.

2. Easy Java simulations - Ejs

A grande parte das páginas com algum conteúdo dinâmico - jogos, animações ou simulações de algum problema físico - são desenvolvidas empregando basicamente a linguagem de programação Java.

A linguagem de programação Java foi apresentada ao público pela Sun Microsystems [3], no ano de 1994. Projetada inicialmente para ser utilizada em dispositivos eletrônicos inteligentes (agendas eletrônicas, televisores, etc.), ganhou aceitação e popularidade ao ser usada em pequenos programas - Applets - executados em navegadores da Internet.

Uma das principais características da linguagem de

[1]Estudante do Mestrado Profissionalizante em Ensino de Física, IF, UFRGS. E-mail: jalves@pb.cefetpr.br.

programação Java, e o que a torna atraente e funcional para a rede, é a independência de plataforma - programas escritos nesta linguagem rodam em qualquer máquina/sistema. Assim, computadores utilizando Windows, Linux ou alguma outra plataforma executam o mesmo aplicativo desenvolvido em Java. Além disso, possui outras características que lhe conferem vantagens sobre as demais linguagens, tais como a simplicidade - apresenta uma estrutura mais limpa, tornando a programação mais simples em relação a outras linguagens, tais como C++ e a orientação a objetos - permite a reutilização do código, aumentando assim a produtividade. Alguns autores [4] acreditam que a evolução das linguagens de programação para o paradigma orientada a objetos representa uma aproximação destas linguagens das características do pensamento humano. A linguagem é elogiada, ainda pelo conjunto bem documentado de bibliotecas. Devido estas características, Java conseguiu adeptos e é uma das poucas linguagens de programação ensinada em cursos universitários.

Como já referido, com o propósito de tornar mais fácil e rápida a construção de simulações utilizando Java, foi desenvolvida a ferramenta Easy Java Simulations - Ejs. O *software* **E**js, já em sua versão 3.3, vem tornando-se cada vez mais popular na rede. Observa-se esta aceitação principalmente pelo grande número de Applets construídos com Ejs. A Fig. 1 apresenta a tela de abertura do Ejs.

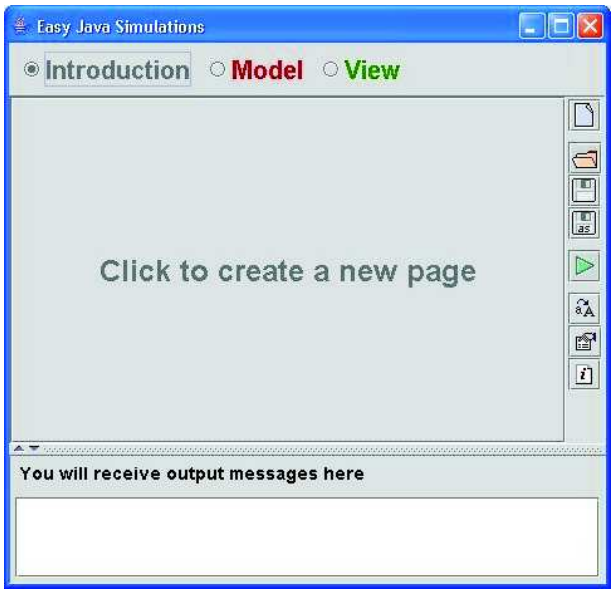

Figura 1 - Tela de abertura do *software* Ejs.

Construído inteiramente na linguagem de programação Java pelo pesquisador e professor Francisco Esquembre [5], pode ser baixado gratuitamente da rede no endereço:

http://fem.um.es/Ejs/

Esquembre relata as vantagens do Ejs:

"Ejs ha sido diseñado para permitir a una persona que quiere crear una simulación, concentrar la mayor parte de su tiempo en escribir y refinar los algoritmos del modelo científico de interes (que es em lo que realmente es experto/a) y dedicar el menor tiempo posible a las técnicas de programación. Y, aún así, obtener un produto final independiente, de altas prestaciones y preparado para su uso em Internet".

Com opções de instalação em espanhol e inglês, a interface gráfica disponibiliza um conjunto de componentes, sendo possível configurar uma simulação, assim como a disposição dos componentes, somente com o uso do *mouse*. Ao final, o *software* gera o resultado final em uma página *html*. Ejs utiliza o pacote JDK (kit de desenvolvimento Java).

O Ejs permite, por exemplo, a modelagem de problemas físicos envolvendo superfícies tridimensionais; a construção de curvas e a adição de gráficos de duas e três dimensões de maneira fácil pelo usuário. O *software* também dispõe de facilidades para a solução numérica de equações diferenciais de primeira ordem. Permitindo que se escreva as equações de forma direta com as seguintes opções de métodos: Euler, Runge-Kutta 4°, Midpoint. A Fig. 2 ilustra uma simulação no campo da mecânica e a Fig. 3 uma simulação mais elaborada, em eletrostática, usando recursos de programação com código Java. Todas disponíveis no *site* do Ejs. Há também tutoriais sobre a interface gráfica, e dezenas de simulações disponíveis para *download*.

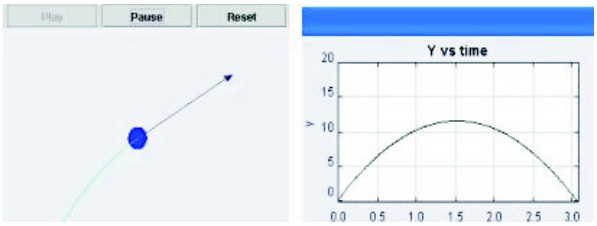

Figura 2 - Applet no campo da mecânica.

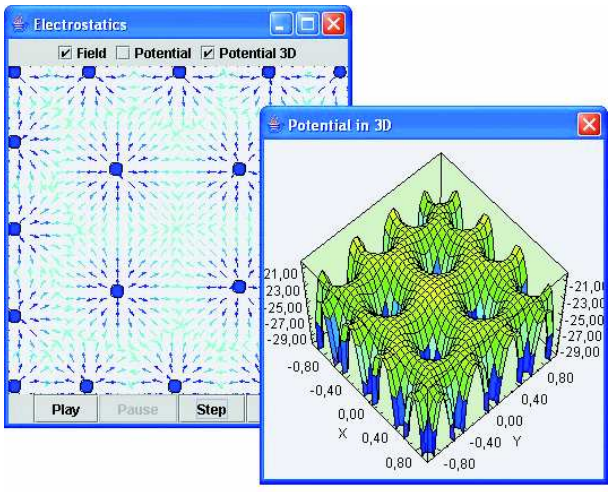

Figura 3 - Applet disponível na página do Ejs, simula o potencial eletrostático em três dimensões.

Na versão 3.3, foram incluídos novos recursos. O *software* permite que sejam importadas funções do MATLAB e, destacam-se também novos componentes da interface gráfica.

As principais operações no desenvolvimento de uma modelagem são feitas usando uma das seguintes opções: *Introduction, Model e View,* disponíveis na tela de abertura do *software*.

- Janela *Introduction*: neste ambiente gráfico é disponibilizado um editor *html*, com ferramentas de formatação de texto e inserção de figuras. Assim, o trabalho final pode ser bem documentado. O conjunto final, simulação e documentação são salvas em uma página de *html*.

- Janela *Model*: Principal janela para o desenvolvimento da modelagem. Permite as seguintes opções para o usuário: *Variables* - neste ambiente são incluídos os nomes, tipos e valores das variáveis usadas na simulação; *Initialization* - permite que as variáveis sejam inicializadas por meio de linhas de código Java; *Evolution* - nesta janela temos três painéis: o painel à esquerda, permite escolher o número de quadros por segundo em que será apresentada a evolução das equações e, à direita dividido em duas partes: permite que se escolha um editor de equações diferenciais de primeira ordem (Runge-Kutta 4°, Euler, Midpoint), ou digitar algoritmo em código Java; *Constraints* - semelhante à opção *Inicialization*, permite que se incluam vínculos às equações; *Custom* – neste painel é possível escrever rotinas mais elaboradas em código Java.

- Janela *View*: neste ambiente gráfico é possível configurar a apresentação final ao usuário (Interface Gráfica) sem digitar nenhuma linha de comando. O Ejs utiliza o conjunto de bibliotecas *Swing* do Java2. A disposição dos componentes gráficos, tais como: painéis, botões, caixa de textos e janelas determinam também o nível de interatividade com o usuário. Detalhes sobre a configuração dos componentes são apresentados no Apêndice.

As opções *Constraints* e *Custom*, descritas acima e usadas na modelagem com código Java, somente são necessárias em simulações com nível de complexidade maior.

O Easy Java Simulations é parte do projeto *Open-Source Physics Education* [6], e, portanto, utiliza um conjunto de bibliotecas com código aberto. Diversos projetos vem sendo desenvolvidos com o objetivo de formar um conjunto de bibliotecas usando *Open-Source*. Dentre esses destacam-se os trabalhos do pesquisador Wolfgang Christian [7] e o projeto Physlets [8], todos destinados à melhoria da qualidade do ensino de ciências.

3. Simulações

Dentre os diversos usos do computador como recurso didático, a simulação ou modelagem é uma atividade que permite uma maior interatividade dos alunos com um determinado modelo físico. No ensino de Física tem, dentre outros, o objetivo de ilustrar e questionar o aluno sobre conceitos e modelos físicos. Assim, a modelagem computacional constitui um recurso didático no ensino de Física de atualização e enriquecimento as atividades de ensino.

Algumas das vantagens que as atividades de modelagem realizadas em um computador podem oferecer no processo de ensino e aprendizagem são as seguintes:

- Permite a visualização gráfica de elementos sutis do modelo teórico.

- Possibilita a participação ativa dos alunos: sistemas interativos exigem respostas e tomadas de decisões, fazendo com que o aluno construa seu próprio conhecimento.

- Interpretação de modelos físicos: ao utilizar laboratórios virtuais e testar hipóteses, obtendo previsões sobre esses sistemas, o aluno é capaz de refletir sobre diferentes modelos teóricos.

É importante que se faça a distinção, com nossos alunos, entre simulações e experimentos físicos. As ciências são construções humanas com base em observações e conjecturas. Assim, o experimento físico é crucial em todo o desenvolvimento teórico (da ciência). A modelagem ou simulação são modelos teóricos da realidade, com base nessas observações do mundo real. Assim, entende-se que simulações não podem substituir a atividade experimental, e o uso de laboratórios, no ensino, é uma necessidade para que nossos alunos compreendam a atividade científica.

Nas próximas seções serão descritas duas aplicações didáticas na modelagem computacional com o Ejs. A primeira explorando o sistema massa-mola e a segunda mais elaborada explorando um tópico de Mecânica Quântica. Não é o objetivo deste trabalho esgotar o assunto, e sim despertar nos leitores desta revista o interesse pelo *software.*

3.1. Sistema massa-mola

Neste trabalho foi aplicada a modelagem em um sistema massa-mola ideal sem atrito. Isto significa que a mola é desprovida de massa e foram desprezadas as forças de contato entre superfície e bloco.

A mola de constante elástica K encontra-se fixa a uma parede. Uma partícula de massa m é presa à extremidade livre da mola. A força F exercida pela extremidade livre da mola que oscila uma distância x é dada pela função:

$$F(x) = -Kx, \tag{1}$$

onde F é uma força restauradora, levando o sistema a sua configuração original. O sistema encontra-se em equilíbrio na coordenada $x = 0$.

Considerando a segunda lei do movimento de Newton, força igual a massa vezes a aceleração, dada por:

$$F = m\,\frac{d^2x}{dt^2}. \qquad (2)$$

Tem-se da igualdade das Eqs. (1) e (2),

$$\frac{d^2x}{dt^2} = -\frac{K}{m}\,x. \qquad (3)$$

A partir da equação diferencial (3), conhecendo as condições iniciais $x(0) = x_0$ *e* $V(0) = V_0$ do sistema, pode-se determinar a evolução em um instante de tempo futuro t.

No planejamento da modelagem, teve-se a preocupação com a interatividade do modelo. Deste modo, as grandezas: massa (m), posição (x) e constante elástica (K) foram escolhidas de forma a serem alterados seus parâmetros pelo aluno. Na Fig. 4 observa-se o resultado final.

Na implementação da modelagem com o Ejs, em termos gerais, procurou-se seguir os seguintes passos.

- Passo 1: Compor o conjunto de componentes da interface gráfica: botões, janela gráfica, painéis, caixa de texto, blocos, partículas, molas. Detalhes da construção são descritos no Apêndice.

- Passo 2: Determinar o conjunto de variáveis do modelo envolvido. Na janela *Model* opção *Variables*, insere-se o nome e valor das variáveis. No sistema proposto, foram utilizados um conjunto de sete variáveis: tempo (t), incremento de tempo $(\mathrm{d}t)$, comprimento da mola (l), velocidade (V), massa (m), constante elástica (K) e posição (x).

- Passo 3: Especificar as expressões que determinam a evolução do sistema. Na janela *Evolution*, clique com o botão direito do *mouse* e escolha Adicionar página de eq. diferencial. E, escreve-se a Eq. (3) de segunda ordem na forma de um sistema de equações de primeira ordem: $d\,x/d\,t = V$ e $d\,V/d\,t = -K/m\,(x)$.

Observa-se na Fig. 4, à direita na parte superior o painel gráfico e, em baixo, à esquerda, as caixas de textos para as entradas das grandezas: massa, constante elástica, distensão da mola.

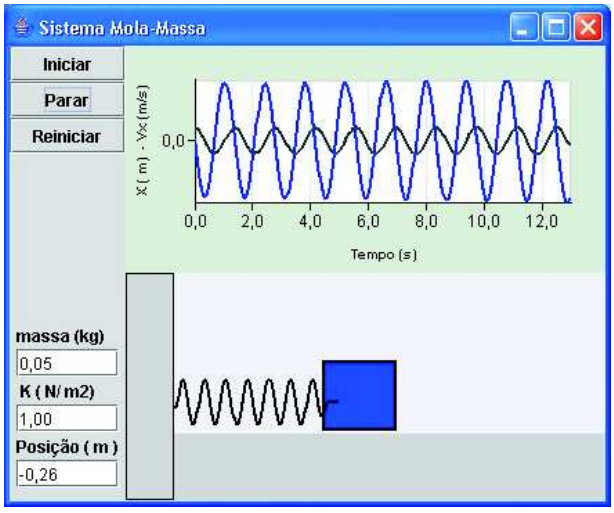

Figura 4 - Simulação usando o Ejs para um sistema massa-mola.

Nesta aplicação não foram utilizadas as janelas *Initialization*, *Constraints e Custom.*

Conceitos importantes do sistema massa-mola podem ser explorados neste simples Applet. Na simulação, a partícula move-se ao longo da superfície; no painel superior, é possível verificar os gráficos da posição e velocidade em função do tempo.

Este aplicativo exige a participação ativa dos alunos e constitui um exemplo de laboratório virtual.

3.2. Solução numérica da equação de Schroedinger independente do tempo

A equação diferencial (3) da seção 3.1 descreve o comportamento de um oscilador harmônico simples para um sistema macroscópico. Sistemas macroscópicos são descritos pelas leis de Newton da mecânica clássica.

Nesta seção, desenvolve-se uma modelagem para um sistema microscópico. Partículas microscópicas tem seu movimento governado por uma função de onda associada, cujo comportamento é descrito pela teoria de Schroedinger da mecânica quântica.

Para tal, considera-se uma partícula de massa m aprisionada entre os pontos $-a/2 < x < a/2$, com energia total E e função energia potencial $V(x)$ idealizada da forma:

$$V\,(\,x\,) = \begin{cases} V_o,\, x < -a/2 \quad \text{ou} \quad x > a/2 \\ 0\,,\; -a/2 < x < a/2, \end{cases}$$

onde a representa a largura do poço.

A equação de Schroedinger independente do tempo unidimensional (4) descreve o comportamento da dependência espacial da função de onda associada à partícula:

$$\frac{d^2\psi}{d\,x^2} = \frac{2\,m}{\hbar^2}\,[\,V(x) - E\,]\;\psi. \qquad (4)$$

Eisberg [9] descreve a solução numérica para este exemplo e, usando argumentos qualitativos referentes as curvaturas e inclinações da solução ψ, determina os valores da energia total E para as quais a equação de Schroedinger independente do tempo tem solução.

Esta técnica possibilita explorar as propriedades da equação diferencial (4), e também permite apresentar, de uma forma qualitativa, a necessidade da quantização da energia em sistemas microscópicos com partículas ligadas.

Em uma solução numérica, é preferível substituir a coordenada x,na Eq. (4), pela coordenada adimensional u, sendo $u = x\,/\,a$. Da mesma forma, escreve-se $\hbar = 1$, $m = 1$ *e* $a = 1$. Por fim, tem-se a Eq. (4) na forma de um sistema de equações de primeira ordem.

$$\frac{d\,\psi}{d\,u} = a\,\varphi, \quad \frac{d\,\varphi}{d\,u} = K\,\psi. \qquad (5)$$

Na Eq. (5), K tem o seguinte valor: $K = 2ma/\hbar^2\,[\,V(\,u\,) - E\,]$.

Na implementação da modelagem foi determinado as soluções pares de um poço simétrico, em que $\psi(-x) = \psi(x)$.

Finalmente, na construção do modelo, em termos gerais, deve-se seguir os seguintes passos:

- Passo 1: Determinar a interface gráfica (*layout* final): conjunto de componentes, botões, gráficos, painel de desenho, paredes do poço. O poço deve ser simétrico na coordenada ***u.*** A construção da interface gráfica é descrita no Apêndice deste trabalho.

- Passo 2: Escolher o conjunto de variáveis necessárias para a modelagem. Nesta aplicação foi utilizado um conjunto de sete variáveis: energia total (E), potencial (V), variável independente (u), constante (K), variável dependente (ψ), largura do poço (a) e função (φ). Sendo as condições iniciais: $\psi(0) = 1$ e $\varphi(0) = 0$.

- Passo 3: Determinar a evolução do sistema. Na janela *Evolution,* clique com o botão direito do *mouse* e escolha Adicionar página de equação diferencial. Nesta janela inserem-se as seguintes equações: $d\psi/du = a\varphi$ e $d\varphi/du = K\psi$.

- Passo 4: Especificar na janela *Constraints* os limites do potencial $V(u)$.

A Fig. 5 representa o resultado final obtido. Pode-se verificar o comportamento da solução (ψ), para dois valores arbitrários da energia total E e potencial V_0.

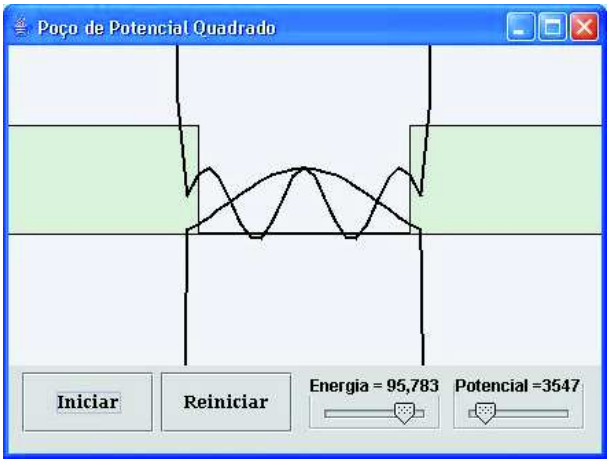

Figura 5 - Solução numérica da equação de Schroedinger para um poço de potencial quadrado.

4. Considerações finais

A necessidade de inserção de linhas de código Java, nas janelas: *Inicialization, Constraints* e *Custom* dependem do nível de sofisticação da modelagem. Contudo, o Ejs é extremamente funcional, é um *software* livre e apresenta facilidades nas construções de Applets: possui rotinas prontas para soluções numéricas de equações diferenciais, com uma interface de fácil manuseio, apresentando janelas, menus e ícones. Com o *software* Ejs é possível configurar a disposição de componentes de um Applet com simples toques do *mouse*, obtendo o trabalho final já configurado para uma página de *html*.

Tendo isso em vista, o Ejs constitui uma importante ferramenta computacional de apoio à pesquisa e ao ensino de Física nas atividades didáticas de modelagem.

5. Agradecimentos

Aos colegas pelas sugestões, ao Centro de Referência para o Ensino de Física-IF/UFRGS e especialmente ao árbitro pelas inúmeras observações e sugestões.

Apêndice

Construindo a interface gráfica de uma simulação

A interface gráfica, número e disposição dos componentes gráficos de uma simulação têm uma importância fundamental em uma apresentação, esta determina a interatividade com o usuário.

Com o Ejs aberto, clique na opção *View.* Assim, teremos a seguinte visualização, Fig. 6.

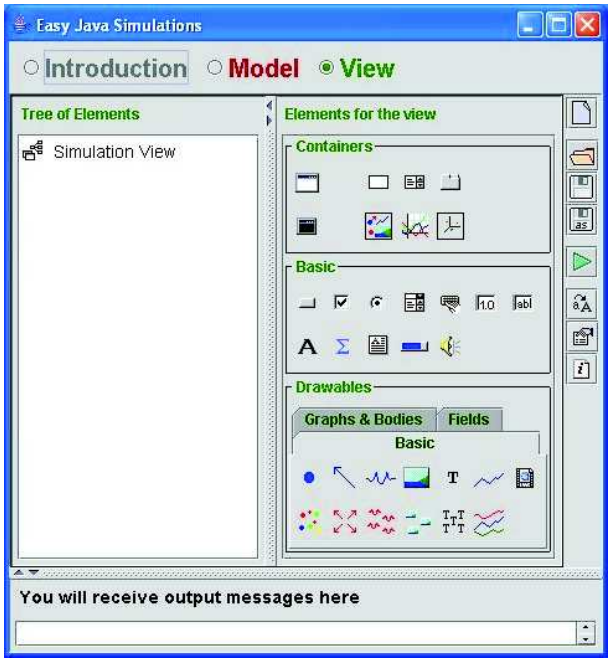

Figura 6 - Visualização da janela *View.*

Pode-se observar nesta figura, dois painéis: à direita, os componentes gráficos e, à esquerda, o ambiente de trabalho. Cada elemento gráfico (componente) possui determinadas propriedades, os quais podem ser modificadas com o uso do *mouse.*

Os principais componentes de um Applet são: janela, painel, gráficos, linhas, painel de desenho, botões, caixa de texto, barra de rolagem, caixa de seleção, blocos, molas, etc. Para adicioná-los a área de trabalho, clique sobre o componente e sobre o elemento a ser adicionado.

Os componentes são divididos nos seguintes grupos: C*ontainers, Basic e Drawables.*

Containers são aqueles componentes que hospedam outros componentes. A principal propriedade dos *Containers* é o *layout*, esta determina a posição do componente inserido. É possível escolher os seguintes gerenciadores de *layout*: *Flow, Border, Grid, Horizontal Box, Vertical Box.*

Vamos examinar a disposição dos componentes do Applet desenvolvido na seção 3.2 Fig. 5.

Os principais passos na construção da interface gráfica da seção 3.2 são:

- Passo 1: Adicione uma janela principal no ambiente de trabalho. Com o botão esquerdo do *mouse* clique sobre o *containers Frame* e, em seguida na área de trabalho. Após, com o botão direito, clique sobre este elemento e escolha a opção *Rename*. Assim é possível alterar o nome do componente para "JanelaPrincipal".

- Passo 2: Repita o procedimento com o ícone painel de desenho. Escolha a posição centro para este componente. Na Fig. 7 pode-se observar a disposição dos componentes na área de trabalho.

- Passo 3: Adicione um painel no componente "JanelaPrincipal". Clique sobre o ícone painel e em seguida sobre o componente "JanelaPrincipal" na área de trabalho. Escolha a posição *Dow* e, semelhante ao passo 1 altere o nome para "Painel". Com o botão direito do *mouse* altere a propriedade *layout* para: *GridLayout Rows 1, Columns 4*. Desta forma os novos componentes adicionados a este painel estarão distribuídos ao longo de uma tabela.

- Passo 4: Para adicionar botões e barras de rolagem no componente Painel, repita o procedimento no passo 1. As propriedades desses componentes são alteradas usando o botão esquerdo *mouse*.

Seguindo os procedimentos acima, adicione também no painel de desenho dois traços (traçoA e traçoB) e figuras de desenho (parede A e paredeB). As propriedades desses componentes são facilmente alteradas com o *mouse*.

O resultado final dos componentes na área de trabalho é ilustrado na Fig. 7.

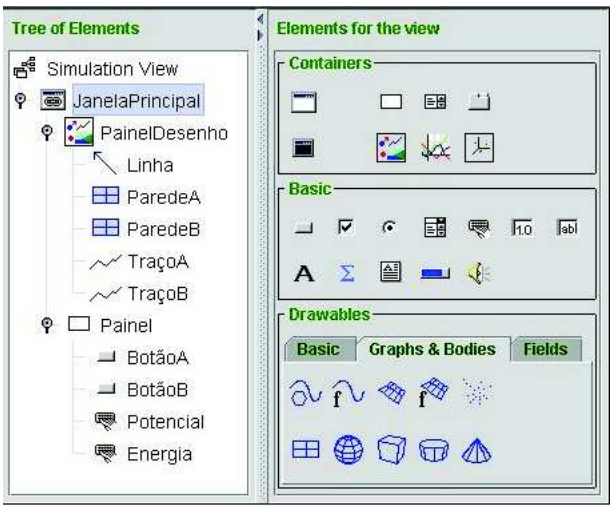

Figura 7 - Detalhes da distribuição dos componentes gráficos.

Referências

[1] I. Yamamoto e V.B. Barbeta, Rev. Bras. Ens. Fís. **23**, 215 (2001).

[2] I.S. Araújo, E.A. Veit e M.A. Moreira, Rev. Bras. Ens. Fís. **26**, 179 (2004).

[3] Sun microsystem, disponível em http://java.sun.com/. Acesso em 26 jan. 2005.

[4] J.S. Nogueira, C. Rinaldi, J.M. Ferreira e S.R. Paulo, Rev. Bras. Ens. Fís. **22**, 517 (2000).

[5] Francisco Esquembre, página pessoal, disponível em http://fem.um.es/fem/. Acesso em 26 jan. 2005.

[6] Open-Source Physics Education, disponível em: http://www.opensourcephysics.org/. Acesso em 26 jan. 2005.

[7] Wolfgang Christian, disponível em http://webphysics.davidson.edu/Faculty/wc/Welcome.htm. Acesso em 26 jan. 2005.

[8] Physics Applets. Disponível em http://webphysics.davidson.edu/Applets/Applets.html Acesso em 26 jan. 2005.

[9] Robert Martin Eisberg, *Física Quântica* (Ed. Campus, Rio de Janeiro, 1979).