Java Desktop Integration Components (JDIC)

J.1 Introdução

Os **Java Desktop Integration Components (JDIC)** são parte de um projeto de código-fonte aberto que visa a uma melhor integração entre aplicativos Java e plataformas em que eles executam. Alguns recursos JDIC incluem:

- interagir com a plataforma subjacente para iniciar aplicativos nativos (como navegadores Web e clientes de e-mail);
- exibir uma tela de abertura ou *splash screen* quando um aplicativo inicia a execução para indicar ao usuário que o aplicativo está carregando;
- criar ícones na bandeja de sistema (também chamada área de status da barra de tarefas ou área de notificação) para fornecer acesso a aplicativos Java em execução no segundo plano;
- registrar associações de tipo de arquivo para que os arquivos dos tipos especificados abram automaticamente nos aplicativos Java correspondentes;
- criar pacotes instaladores e mais.

A home page do JDIC (`jdic.dev.java.net/`) inclui uma introdução ao JDIC, downloads, documentação, FAQs, demonstrações, artigos, blogs, comunicados, projetos incubadores, página de um desenvolvedor, fóruns, listas de mala direta etc. O Java SE 6 inclui alguns dos recursos mencionados acima. Discutimos vários desses recursos aqui.

J.2 Telas de splash

Usuários de aplicativos Java muitas vezes percebem um problema de desempenho porque nada aparece na tela quando o aplicativo é inicializado pela primeira vez. Um modo de mostrar a um usuário que seu programa está carregando é exibir uma **tela de splash** — uma janela sem bordas que aparece temporariamente enquanto um aplicativo carrega. O Java SE 6 fornece a nova opção de linha de comando `-splash` para o comando `java` executar essa tarefa. Essa opção permite especificar uma imagem PNG, GIF ou JPG que deve ser exibida quando seu aplicativo começa a carregar. Para demonstrar essa nova opção, criamos um programa (Figura J.1) que permanece em repouso por 5 segundos (para você poder visualizar a tela de splash) e então exibe uma mensagem na linha de comando. O diretório desse exemplo inclui uma imagem no formato PNG para utilizar como a tela de splash. Para exibir a tela de splash quando esse aplicativo carrega, utilize o comando

```
java -splash:DeitelBug.png SplashDemo
```

Depois de iniciar o vídeo da tela de splash, você pode interagir com ele programaticamente via classe `SplashScreen` do pacote `java.awt`. Talvez você faça isso para adicionar algum conteúdo dinâmico à tela de splash. Para obter informações adicionais sobre como trabalhar com telas splash, consulte os seguintes sites:

```
java.sun.com/developer/technicalArticles/J2SE/Desktop/javase6/
   splashscreen/
java.sun.com/javase/6/docs/api/java/awt/SplashScreen.html
```

```
1  // Figura J.1: SplashDemo.java
2  // Demonstração da tela de splash.
3  public class SplashDemo
4  {
5     public static void main( String[] args )
6     {
7        try
8        {
9           Thread.sleep( 5000 );
10       } // fim do try
11       catch ( InterruptedException e )
12       {
13          e.printStackTrace();
14       } // fim do catch
15
16       System.out.println(
17          "This was the splash screen demo." );
18    } // fim do método main
19 } // fim da classe SplashDemo
```

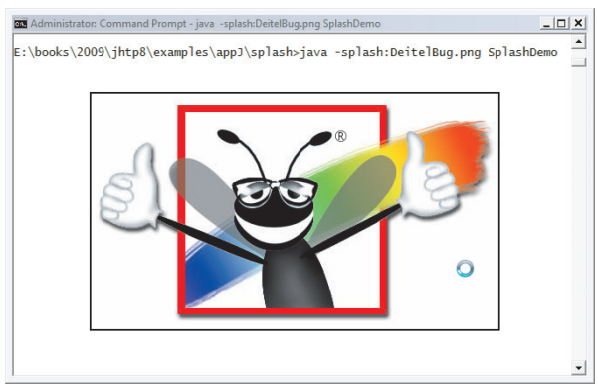

Figura J.1 | Tela de splash exibida com a opção `-splash` para o comando `java`.

J.3 Classe Properties

A nova classe **`Desktop`** do Java 6 SE permite especificar um arquivo ou URI que você quer abrir utilizando o aplicativo apropriado da plataforma subjacente. Por exemplo, se o Firefox for o navegador padrão do seu computador, utilize o método de navegação da classe `Desktop` para abrir um site Web no Firefox. Além disso, você pode abrir uma janela de composição de e-mail no cliente de e-mail padrão do seu sistema, abrir um arquivo no aplicativo associado e imprimir um arquivo utilizando o comando print do aplicativo associado. A Figura J.2 demonstra as três primeiras dessas capacidades.

O handler de evento nas linhas 86–116 obtém o número de índice da tarefa que o usuário seleciona no `tasksJComboBox` (linha 89) e a `String` que representa o arquivo ou URI a processar (linha 90). A linha 92 utiliza o método **`isDesktopSupported`** `Desktop static` para determinar se os recursos da classe `Desktop` são suportados na plataforma em que esse aplicativo executa. Se forem, a linha 96 utiliza o método **`getDesktop`** `Desktop static`, para obter um objeto `Desktop`. Se o usuário selecionou a opção de abrir o navegador padrão, a linha 101 cria um novo objeto `URI` utilizando a entrada de `String` como o site para exibir no navegador e então passa o objeto `URI` para o método `Desktop` **`browse`** que invoca o navegador padrão do sistema e passa o `URI` para o navegador a fim de exibi-lo. Se o usuário selecionar a opção de abrir um arquivo no programa associado, a linha 104 cria um novo objeto `File` utilizando a entrada de `String` como o arquivo a ser aberto e, então, passa o objeto `File` para o método `Desktop` **`open`**, que passa o arquivo para o aplicativo apropriado a fim de abrir o arquivo. Por fim, se o usuário seleciona a opção de compor um e-mail, a linha 107 cria um novo objeto `URI` utilizando a entrada de `String` como o endereço de e-mail ao qual o e-mail será enviado, passa então o objeto `URI` para o método `Desktop` **`mail`** que invoca o cliente de e-mail padrão do sistema e passa o `URI` para o cliente de e-mail como o destinatário do e-mail. Aprender mais sobre a classe `Desktop` em

```
java.sun.com/javase/6/docs/api/java/awt/Desktop.html
```

```
1   // Figura J.2: DesktopDemo.java
2   // Usa Desktop para iniciar o navegador padrão, abrir um arquivo
3   // no seu aplicativo associado e um e-mail no cliente de e-mail padrão.
4   import java.awt.Desktop;
5   import java.io.File;
6   import java.io.IOException;
7   import java.net.URI;
8
9   public class DesktopDemo extends javax.swing.JFrame
10  {
11     // construtor
12     public DesktopDemo()
13     {
14        initComponents();
15     } // fim do construtor DesktopDemo
16
17     // Para economizar espaço, as linhas 20-84 do código de GUI autogeradas pelo
18     // NetBeans elas não são mostradas aqui. O código completo desse exemplo
19     // está localizado no arquivo DesktopDemo.java no diretório desse exemplo.
20
85     // determina a tarefa selecionada e executa a tarefa
86     private void doTaskJButtonActionPerformed(
87        java.awt.event.ActionEvent evt)
88     {
89        int index = tasksJComboBox.getSelectedIndex();
90        String input = inputJTextField.getText();
91
92        if (Desktop.isDesktopSupported())
93        {
94           try
95           {
96              Desktop desktop = Desktop.getDesktop();
97
98              switch ( index )
99              {
100                case 0: // abre o navegador
101                   desktop.browse( new URI( input ) );
102                   break;
103                case 1: // abre o arquivo
104                   desktop.open( new File( input ) );
105                   break;
106                case 2: // abre a janela de composição de e-mail
107                   desktop.mail( new URI( input ) );
108                   break;
109             } // fim do switch
110          } // fim do try
111          catch ( Exception e )
112          {
113             e.printStackTrace();
114          } // fim do catch
115       } // fim do if
116    } // fim do método doTaskJButtonActionPerformed
117
118    public static void main(String[] args)
119    {
120       java.awt.EventQueue.invokeLater(
121          new Runnable()
122          {
123             public void run()
124             {
125                new DesktopDemo().setVisible(true);
126             }
127          }
128       );
129    } // fim do método main
130
131    // Declaração de variáveis - não modifique
132    private javax.swing.JButton doTaskJButton;
```

```
133       private javax.swing.JLabel inputJLabel;
134       private javax.swing.JTextField inputJTextField;
135       private javax.swing.JLabel instructionLabel;
136       private javax.swing.JComboBox tasksJComboBox;
137       // Fim da declaração de variáveis
138    }
```

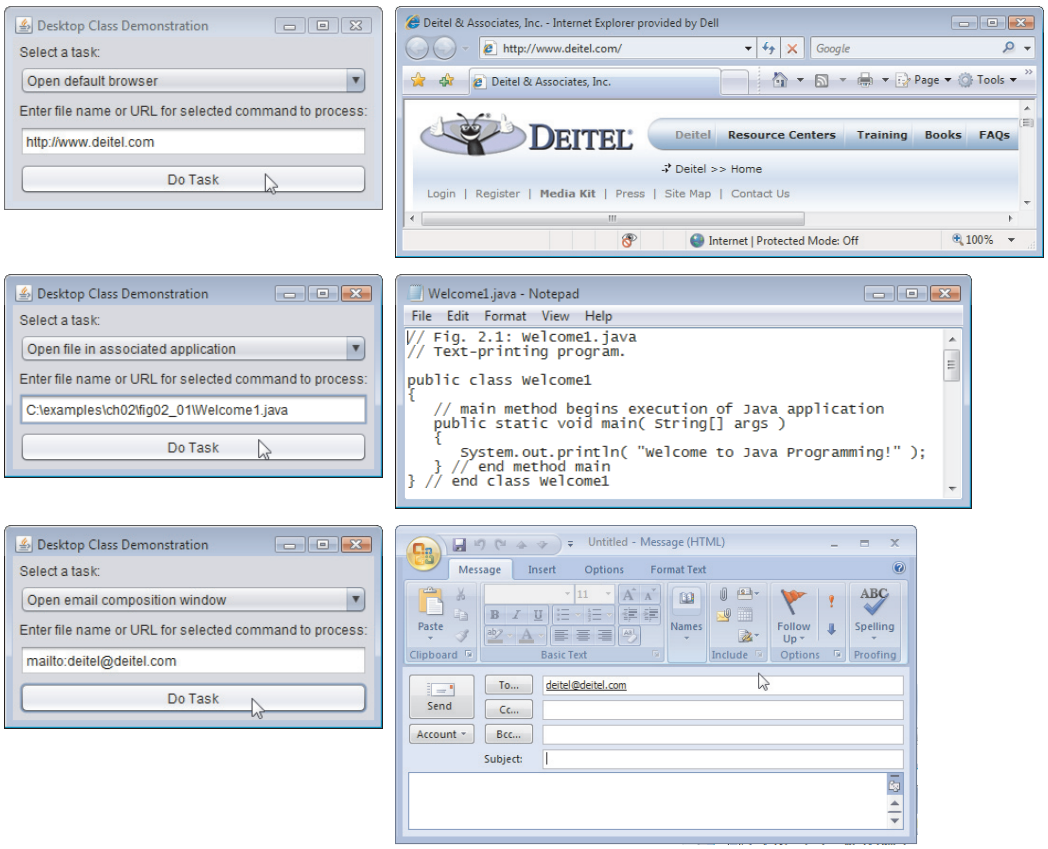

Figura J.2 | Use `Desktop` para iniciar o navegador padrão, abrir um arquivo no seu aplicativo associado e compor um e-mail no cliente de e-mail padrão.

J.4 Ícones da área de status do sistema

Ícones da área de status (*system tray*) geralmente aparecem na área de status do sistema, na área de status da barra de tarefas ou na área de notificação. Eles costumam fornecer acesso rápido a aplicativos que estão em execução no segundo plano no seu sistema. Ao posicionar o mouse sobre um desses ícones, uma dica de ferramenta aparece indicando o aplicativo que o ícone representa. Se você clicar no ícone, um menu pop-up aparece com opções para esse aplicativo.

As classes `SystemTray` e `TrayIcon` (ambos do pacote `java.awt`) permitem criar e gerenciar seus próprios ícones da área de status do sistema de uma maneira independente de plataforma. A classe **`SystemTray`** fornece acesso à área de status do sistema da plataforma subjacente — a classe consiste em três métodos:

- o método `static` **`getDefaultSystemTray`** retorna a área de status do sistema;
- o método **`addTrayIcon`** adiciona um novo `TrayIcon` à área de status do sistema;
- o método **`removeTrayIcon`** remove um ícone da área de status do sistema.

A classe **`TrayIcon`** consiste em vários métodos que permitem aos usuários especificar um ícone, uma dica de ferramenta e um menu pop-up para o ícone. Além disso, os ícones de área de status suportam `ActionListeners`, `MouseListeners` e `MouseMotionListeners`. Você pode aprender mais sobre classes `SystemTray` e `TrayIcon` em

```
java.sun.com/javase/6/docs/api/java/awt/SystemTray.html
java.sun.com/javase/6/docs/api/java/awt/TrayIcon.html
```

J.5 JDIC Incubator Projects

Os JDIC Incubator Projects são desenvolvidos, mantidos e possuídos pelos membros da comunidade Java. Esses projetos estão associados, mas não são distribuídos, com o JDIC. Os Incubator Projects podem acabar se tornando parte do projeto JDIC depois que eles forem totalmente desenvolvidos e atenderem certos critérios. Para informações adicionais sobre os Incubator Projects e aprender como você pode configurar um Incubator Project, visite

```
jdic.dev.java.net/#incubator
```

J.6 Demos JDIC

O site JDIC inclui demonstrações para o FileExplorer, o pacote de navegador, o pacote TrayIcon, a classe Floating Dock e a Wallpaper API (`jdic.dev.java.net/#demos`). O código-fonte para essas demonstrações está incluído no download do JDIC (`jdic.dev.java.net/servlets/ProjectDocumentList`). Para mais demonstrações, verifique alguns projetos de incubadora.