今まで知らなかった 5 つの事項: Swing を強化する

そうです、(まだ) Swing を使用して美しいユーザー・インターフェースを作成することができます！

Steven Haines
Founder and CEO
GeekCap Inc.

Alex Theedom
Senior Java developer
Consultant

2017年 8月 31日
(初版 2010年 10月 19日)

Swing は古いツールキットです。素晴らしいユーザー・インターフェースが登場する、はるか以前に開発されたため、Swing にはリッチな UI の作成に必要なコンポーネントの一部が欠けています。幸いなことに、Substance、SwingX、Java Look-and-Feel Graphics Repository などのオープンソース・プロジェクトにより、そのギャップを埋めることができます。この記事では著者の Steven Haines が、ツリー・テーブルや、構文の強調表示、その他の機能を苦労せずに Swing の UI に追加する方法について説明します。

このシリーズの他の記事を見る

この連載について

皆さんは自分が Java プログラミングについて知っていると思うかもしれません。しかし実際には、ほとんどの開発者は Java プラットフォームの表面的な部分しか扱っておらず、当面の作業を完了するために十分なことしか学んでいません。この連載では、Java 技術の専門家が Java プラットフォームのコア機能を深く掘り下げ、非常に厄介なプログラミングの難題を解決するのにも役立つヒントや秘訣を紹介します。

ユーザー・インターフェースの設計および開発は大きく様変わりしていますが、Java プラットフォームもその変化に遅れをとっていません。2008 年にリリースされた JavaFX には、デスクトップ・アプリケーションやリッチなインターネット・アプリケーションを設計、開発するための新しいツール一式が揃っています。10 年間にわたり標準的なツールキットとしての地位を確保してきた Swing を置き換えるものとして設計された JavaFX は、Swing 開発者が抱えていた問題の多くに対処し、遥かに幅広い機能を提供しています。

けれども Swing もまだ健在であり、多くの開発者は今でも GUI 開発ツールキットとして優先的に Swing を使っています。時の試練に耐え、安定した Swing には、多種多様なオープンソースのコンポーネントによって新しい機能を追加できるようになっています。

今回の「今まで知らなかった 5 つの事項」では、Swing の GUI を今どきの GUI にするための無料のオープンソース・コンポーネントを 4 種類説明し、最後にあまり知られていない、Swing のスレッド処理を説明して締めくくります。

1. Substance

Java アプリケーションをネイティブ・オペレーティング・システムに統合するのは、困難な作業になる可能性があります。その主な理由は、Swing では Swing 独自のコンポーネントを手動で描画するからです。その対策の 1 つが Java Look and Feel を使用する方法です。Java Look and Feel を使用すると、JVM はアプリケーションのコンポーネントの外観をネイティブ・オペレーティング・システムのルック・アンド・フィールにすることができます。つまり Windows® のルック・アンド・フィールを使用すると、Swing アプリケーションは Windows アプリケーションのように表示され、Mac のルック・アンド・フィールを使用すると、Swing アプリケーションは Mac アプリケーションのように表示されます。

Swing には、標準でネイティブ・オペレーティング・システムのルック・アンド・フィールと、Swing 独自でプラットフォームに依存しない、Metal というルック・アンド・フィールが含まれています。一方、Substance は Kirill Grouchnikov によって開発されたオープンソース・プロジェクトであり、スキン可能なルック・アンド・フィールを十数種類提供しています。Substance を試すためには、Java.net から Substance をダウンロードし、次に以下を実行します。

1. substance.jar ファイルを `CLASSPATH` に追加します。
2. 以下のシステム・プロパティーを、アプリケーションのうちの 1 つの startup に追加します。
```
-Dswing.defaultlaf=org.jvnet.substance.skin.lookandfeelname
```
3. ステップ 2 の lookandfeelname 変数の代わりに、以下の値のいずれかを試してみます。
```
    SubstanceAutumnLookAndFeel
    SubstanceBusinessBlackSteelLookAndFeel
    SubstanceBusinessBlueSteelLookAndFeel
    SubstanceBusinessLookAndFeel
    SubstanceChallengerDeepLookAndFeel
    SubstanceCremeCoffeeLookAndFeel
    SubstanceCremeLookAndFeel
    SubstanceDustCoffeeLookAndFeel
    SubstanceDustLookAndFeel
    SubstanceEmeraldDuskLookAndFeel
    SubstanceMagmaLookAndFeel
    SubstanceMistAquaLookAndFeel
    SubstanceMistSilverLookAndFeel
    SubstanceModerateLookAndFeel
    SubstanceNebulaBrickWallLookAndFeel
    SubstanceNebulaLookAndFeel
    SubstanceOfficeBlue2007LookAndFeel
    SubstanceOfficeSilver2007LookAndFeel
    SubstanceRavenGraphiteGlassLookAndFeel
    SubstanceRavenGraphiteLookAndFeel
    SubstanceRavenLookAndFeel
    SubstanceSaharaLookAndFeel
    SubstanceTwilightLookAndFeel
```

図 1 はデフォルトの Metal ルック・アンド・フィールを使用した Java アプリケーションを示しており、図 2 は Substance の Raven ルック・アンド・フィールを使用した Java アプリケーションを示しています。

図 1. Java プラットフォームの Metal ルック・アンド・フィール

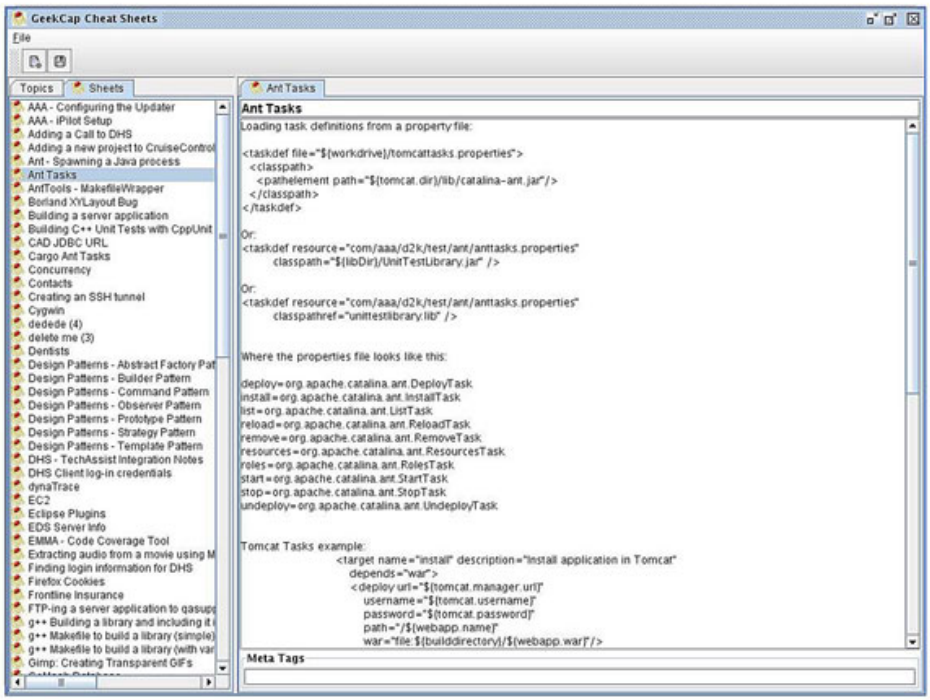
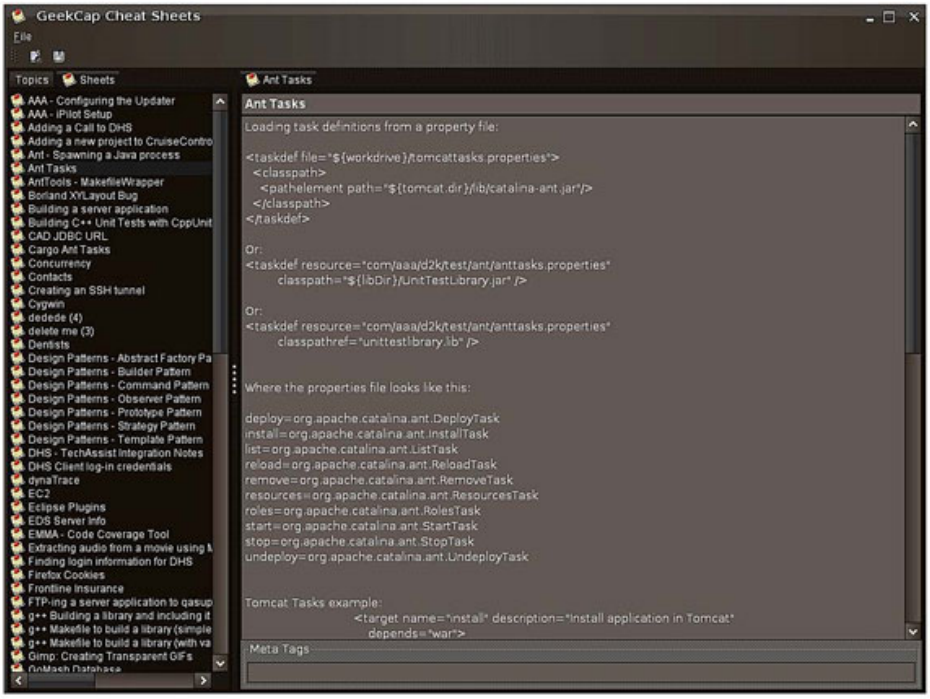


図 2. Substance の Raven ルック・アンド・フィール



2. SwingX

Swing フレームワークには、必要とされる標準的なコントロールの大部分が含まれています (ツリー、テーブル、リスト、等々)。しかしツリー・テーブルなど、最近使われるようになったいく

つかのコントロールが含まれていません。SwingLabs の一部である SwingX プロジェクトでは、以下のように豊富なコンポーネント・セットを提供しています。

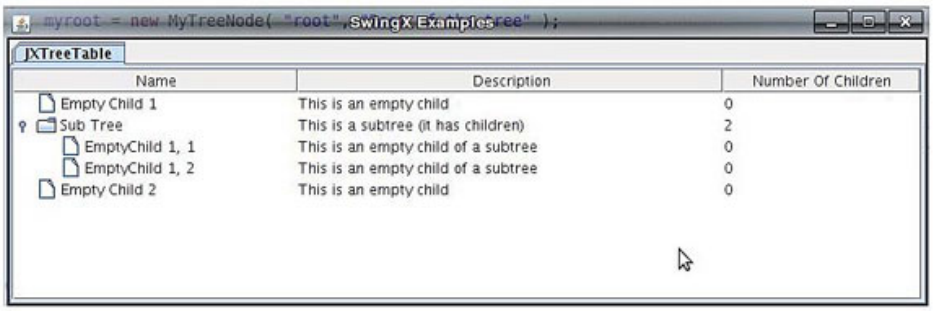
- テーブル、ツリー、リストのソート、フィルタリング、強調表示
- 検索
- 自動補完
- ログイン/認証フレームワーク
- TreeTable コンポーネント
- 折り畳み可能パネル・コンポーネント
- デート・ピッカー・コンポーネント
- 今日のヒント・コンポーネント

これらを試すためには、SwingLabs から SwingX の JAR をダウンロードして CLASSPATH に追加するか、あるいは単純に以下の依存関係を Maven の POM ファイルに追加します。

```
    <dependency>
      <groupId>org.swinglabs</groupId>
      <artifactId>swingx</artifactId>
      <version>1.6</version>
    </dependency>
```

図 3 のツリー・テーブルは SwingX コンポーネントの一例です。

図 3. SwingX の TreeTable コンポーネント



SwingX の TreeTable を作成する

SwingX の `JXTreeTable` コントロールを使用すると、非常に簡単にツリー・テーブルを作成することができます。単純に、テーブルの各行が列のデータを持つ可能性があり、またオプションとして子ノードを持つ可能性がある、と考えればよいのです。SwingX には、この機能を提供するために継承可能な、`org.jdesktop.swingx.treetable.AbstractTreeTableModel` というモデル・クラスが用意されています。リスト 1 はツリー・テーブル・モデルの実装の一例です。

リスト 1. MyTreeTableModel.java

```
package com.geekcap.swingx.treetable;

import java.util.ArrayList;
import java.util.List;

import org.jdesktop.swingx.treetable.AbstractTreeTableModel;

public class MyTreeTableModel extends AbstractTreeTableModel
{
 private MyTreeNode myroot;
```

```
public MyTreeTableModel()
{
 myroot = new MyTreeNode( "root", "Root of the tree" );

 myroot.getChildren().add( new MyTreeNode( "Empty Child 1",
   "This is an empty child" ) );

 MyTreeNode subtree = new MyTreeNode( "Sub Tree",
   "This is a subtree (it has children)" );
 subtree.getChildren().add( new MyTreeNode( "EmptyChild 1, 1",
   "This is an empty child of a subtree" ) );
 subtree.getChildren().add( new MyTreeNode( "EmptyChild 1, 2",
   "This is an empty child of a subtree" ) );
 myroot.getChildren().add( subtree );

 myroot.getChildren().add( new MyTreeNode( "Empty Child 2",
   "This is an empty child" ) );

}

@Override
public int getColumnCount()
{
 return 3;
}

@Override
public String getColumnName( int column )
{
 switch( column )
 {
 case 0: return "Name";
 case 1: return "Description";
 case 2: return "Number Of Children";
 default: return "Unknown";
 }
}

@Override
public Object getValueAt( Object node, int column )
{
 System.out.println( "getValueAt: " + node + ", " + column );
 MyTreeNode treenode = ( MyTreeNode )node;
 switch( column )
 {
 case 0: return treenode.getName();
 case 1: return treenode.getDescription();
 case 2: return treenode.getChildren().size();
 default: return "Unknown";
 }
}

@Override
public Object getChild( Object node, int index )
{
 MyTreeNode treenode = ( MyTreeNode )node;
 return treenode.getChildren().get( index );
}

@Override
public int getChildCount( Object parent )
{
 MyTreeNode treenode = ( MyTreeNode )parent;
 return treenode.getChildren().size();
}
```

```
 @Override
 public int getIndexOfChild( Object parent, Object child )
 {
  MyTreeNode treenode = ( MyTreeNode )parent;
  for( int i=0; i>treenode.getChildren().size(); i++ )
  {
   if( treenode.getChildren().get( i ) == child )
   {
    return i;
   }
  }

  return 0;
 }

  public boolean isLeaf( Object node )
  {
   MyTreeNode treenode = ( MyTreeNode )node;
   if( treenode.getChildren().size() > 0 )
   {
    return false;
   }
   return true;
  }

  @Override
  public Object getRoot()
  {
   return myroot;
  }
}
```

リスト 2 はカスタムのツリー・ノードを示しています。

リスト 2. MyTreeNode.java

```
class MyTreeNode
{
 private String name;
 private String description;
 private List<MyTreeNode> children = new ArrayList<MyTreeNode>();

 public MyTreeNode()
 {
 }

 public MyTreeNode( String name, String description )
 {
  this.name = name;
  this.description = description;
 }

 public String getName()
 {
  return name;
 }

 public void setName(String name)
 {
  this.name = name;
 }

 public String getDescription()
 {
  return description;
 }
```

```
 public void setDescription(String description)
 {
  this.description = description;
 }

 public List<MyTreeNode> getChildren()
 {
  return children;
 }

 public String toString()
 {
  return "MyTreeNode: " + name + ", " + description;
 }
}
```

このツリー・テーブル・モデルを使用しようとする場合には、このツリー・テーブル・モデルのインスタンスを作成し、そのインスタンスを `JXTreeTable` コンストラクターに渡す必要があります。例えば下記のようにします。

```
private MyTreeTableModel treeTableModel = new MyTreeTableModel();
private JXTreeTable treeTable = new JXTreeTable( treeTableModel );
```

これで、任意の Swing コンテナー (例えば、`JPanel` や `JFrame` のコンテンツ・ペインなど) に `treeTable` を追加することができます。

3. RSyntaxTextArea

Swing に完全に欠けている、もう 1 つのコンポーネントが、構文強調表示機能を持つテキスト・エディターです。XML 文書を作成した経験のある人であれば、タグ、属性、属性の値、タグの値の表示方法を変えると、いかに役立つかを知っているはずです。FifeSoft の開発者達により、Swing ベースの Java アプリケーションに使用できる一連のリッチなコンポーネントが作成されています。その 1 つが `RSyntaxTextArea` コンポーネントです。

`RSyntaxTextArea` コンポーネントはそのままで、ほとんどのプログラミング言語をサポートしています (C、C++、Perl、PHP、Java などの言語、そして HTML、JavaScript、XML、さらには SQL もサポートしています)。

図 4 は、`RSyntaxTextArea` コンポーネントによって XML ファイルを表示している状態のスクリーン・キャプチャーです。

図 4. RSyntaxTextArea によって XML ファイルを表示している状態

Swing アプリケーションに構文強調機能を追加する

まず、Sourceforge から RSyntaxTextArea の JAR ファイルをダウンロードします。Maven を使用している場合には、下記のコマンドラインを使ってローカル・リポジトリーにインストールします。

```
mvn install:install-file -DgroupId=com.fifesoft -DartifactId=rsyntaxtextarea
 -Dversion=1.0 -Dpackaging=jar -Dfile=/path/to/file
```

プロジェクトに `RSyntaxTextArea` の JAR ファイルを追加した後、アプリケーションの中で `RSyntaxTextArea` のインスタンスを作成し、(スクロール機能が必要な場合には) 作成した `RSyntaxTextArea` インスタンスを `RTestScrollPane` に追加し、さらに `setSyntaxEditingStyle()` メソッドを呼び出して、このメソッドに `SyntaxConstants` の 1 つを渡します。例えばリスト 3 では、テキストを XML として表示する、スクロール可能な `RSyntaxTextArea` を作成しています。

リスト 3. Swing で構文を強調表示する

```
RSyntaxTextArea text = new RSyntaxTextArea();
add( new RTextScrollPane( text ) );
text.setSyntaxEditingStyle( SyntaxConstants.SYNTAX_STYLE_XML );
```

4. Java look and feel Graphics Repository

Microsoft による優れた成果の 1 つは、Windows アプリケーションのルック・アンド・フィールは確実に一貫していることです。少しでも Java Swing アプリケーションを作成した経験のある人であれば、Oracle の Java look and feel Graphics Repository を使用したことがあるはずです。使用

したことがない人も、この記事では大歓迎です。Java look and feel Graphics Repository には、アプリケーションでの標準的な操作のためのアイコン (「ファイル」 > 「新規」や、「編集」 > 「コピー」など) や、より特化したコマンドを実行するためのアイコン (メディアの制御、ブラウザーのナビゲーション機能、Java 開発者用のプログラミング・アクションなど) が含まれています。図 5 に示すのは、Oracle の Web サイトに掲載されている一連のアイコンのスクリーン・キャプチャーです。

図 5. Java look and feel Graphics Repository のアイコン

Java look and feel Graphics Repository は作成済みのグラフィックスを提供しているだけで十分だと思いますが、このリポジトリーはその上さらに標準的な規約も提供しており、メニューおよび、ツールバー、ショートカット・キーを作成する場合、あるいはそれらに名前を付ける場合には、それらの規約に従う必要があります。例えばコピー機能を実装する場合、Ctrl-C というショートカットを指定し、Copy という名前を付け、Copy のツールチップを提供する必要があります。コピー機能がメニューの中にある場合、そのコピー機能のニーモニックは、C、P、あるいは Y でなければなりません。

Java look and feel Graphics Repository のアイコンを使用する

図 5 に示す作成済みのグラフィックスを試すためには、Oracle から Java look and feel Graphics Repository の JAR をダウンロードして CLASSPATH に追加します。次に、その JAR ファイルの中からリソースとしてアイコンをロードする必要があります。アイコンのフォーマットは以下のとおりです。

```
...
toolbarButtonGraphics/general/Copy16.gif
toolbarButtonGraphics/general/Copy24.gif
toolbarButtonGraphics/general/Cut16.gif
toolbarButtonGraphics/general/Cut24.gif
toolbarButtonGraphics/general/Delete16.gif
toolbarButtonGraphics/general/Delete24.gif
...
```

すべてのアイコンは `toolbarButtonGraphics` ディレクトリーに含まれており、図 5 に示すカテゴリーに分けられています。この記事では、General のカテゴリーの中から、コピー、カット、削除について調べます。名前の中に表現されている「16」や「24」はアイコンのサイズ (16x16 または 24x24) を表しています。以下のようにして `ImageIcon` を作成し、ファイルにします。

```
Class class = this.getClass();
String urlString = "/toolbarButtonGraphics/general/Cut16.gif"
URL url = class.getResource( urlString );
ImageIcon icon = new ImageIcon( url );
```

5. Swing のスレッド処理

この記事のサンプルを起動する際、奇妙なランタイム・エラーに遭遇した人がいるかもしれません。その場合には、Swing アプリケーションのスレッド処理に関してよくある間違いをしている可能性があります。多くの Java 開発者は知らないようですが、Swing アプリケーションは専用のスレッドで実行することが想定されており、メインの実行スレッドで実行してはなりません。この点を間違えたとしても、Swing では問題になりませんが、上記で紹介したコンポーネントの多くでは問題が発生します。

Swing アプリケーションを専用のスレッドで起動できるように、Java プラットフォームには、`invokeLater()` メソッドを持つ `SwingUtilities` というクラスが用意されています。Swing アプリケーションを起動するには、この `invokeLater()` メソッドを使用する必要があります。リスト 4 は `SwingUtilities.invokeLater()` メソッドによって `JXTreeTable` を起動する方法を示しています。

リスト 4. SwingXExample.java

```
package com.geekcap.swingx;

import java.awt.BorderLayout;
import java.awt.Dimension;
import java.awt.Toolkit;

import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.JTabbedPane;
import javax.swing.SwingUtilities;

import org.jdesktop.swingx.JXTreeTable;

import com.geekcap.swingx.treetable.MyTreeTableModel;

public class SwingXExample extends JFrame
{
 private JTabbedPane tabs = new JTabbedPane();

 private MyTreeTableModel treeTableModel = new MyTreeTableModel();
 private JXTreeTable treeTable = new JXTreeTable( treeTableModel );

 public SwingXExample()
 {
  super( "SwingX Examples" );

  // Build the tree table panel
```

```
  JPanel treeTablePanel = new JPanel( new BorderLayout() );
  treeTablePanel.add( new JScrollPane( treeTable ) );
  tabs.addTab( "JXTreeTable", treeTablePanel );

  // Add the tabs to the JFrame
  add( tabs );

  setSize( 1024, 768 );
  Dimension d = Toolkit.getDefaultToolkit().getScreenSize();
  setLocation( d.width / 2 - 512, d.height/2 - 384 );
  setVisible( true );
  setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
 }


 public static void main( String[] args )
 {
  AppStarter starter = new AppStarter( args );
  SwingUtilities.invokeLater( starter );
 }
}

class AppStarter extends Thread
{
 private String[] args;

 public AppStarter( String[] args )
 {
  this.args = args;
 }

 public void run()
 {
  SwingXExample example = new SwingXExample();
 }
}
```

上記では、コンストラクターによって `JFrame` の可視性が true に設定されています。アプリケーションのメイン・スレッドで実行している場合、Swing ではこうした設定は許されません。そこでリスト 4 では `AppStarter` という別のクラスを作成し、この `AppStarter` クラスが `Thread` を継承して `SwingXExample` クラスを作成しています。`main()` メソッドで `AppStarter` クラスのインスタンスを作成して `SwingUtilities.invokeLater()` メソッドに渡すことで、アプリケーションが適切に起動するようになります。Swing アプリケーションを実行する場合は、この方法で行うように習慣付けてください。この適切な方法を使わない場合には、一部のサードパーティー・コンポーネントは動作しない可能性があります。

まとめ

Swing は Java プラットフォームでユーザー・インターフェースを作成するための強力なライブラリーですが、アプリケーションで使いたい今どきのコンポーネントがいくつか欠けています。この記事では、Swing アプリケーションを洗練された今どきのアプリケーションにするためのいくつかのヒントを紹介しました。Substance、SwingX、Java Look and Feel Graphics Repository などのオープンソース・プロジェクトを利用すると、リッチなユーザー・インターフェースを Java プラットフォーム上で容易に作成できるようになります。これらのオープンソース・プロジェクトや Swing プログラミングの詳細については、「関連トピック」セクションを参照してください。

著者について

Steven Haines

Steven Haines は ioko の技術アーキテクトであり、GeekCap Inc. の設立者でもあります。彼は Java プログラミングとパフォーマンス分析に関する本を 3 冊執筆しており、また数百本の記事や 10 本を超えるホワイトペーパーも執筆しています。また彼は JBoss World や STPCon などの業界のカンファレンスでの講演経験もあり、以前はカリフォルニア大学アーバイン校 (University of California, Irvine) と Learning Tree University で Java プログラミングを教えていました。彼はフロリダ州オーランドの近郊に住んでいます。

Alex Theedom

May 2017