目次

1. JDesktop Network Components

JDesktop Network Components (以下、JDNC)は、2004 年の Java One でも紹介された、Sun が進めるリッチクライアントのユーザインターフェイス(以下、UI)、ネットワーク接続機能を提供するコンポーネント群で、

- スタンドアロンアプリケーション
- JavaWebStart
- Applet

での利用を想定しており、今まで難しいといわれていた Swing のコーディングなしで UI を作成することが出来ます

※厳密には UI だけでなく、データベースからのデータ取得などの非 UI の部分も作成できますが、本資料では UI と記述します。

JDNC の開発リーダが、JSF の初代 Spec Lead でもあるため(当初は JSF の一部として実装される予定だった)、JSF との親和性が高く、JSF のビジネスロジックを利用しつつクライアントサイドは JDNC でといった組み合わせも考えられます。

現在開発が進められている途中のため、保存が出来ないなど実用にはまだ遠いですが、リッチクライアントのデファクトスタンダードとなる日が来るかもしれません。

本資料ではサンプルをベースに JDNC を使用した UI 開発の流れを見ていきます。

2005 年 1 月現在の最新バージョンは 0.6 で、今後は以下のスケジュールでリリースされる予定です。

- 2005 年 2 月　0.7 リリース(データ接続関連)
- 2005 年 5 月　0.8 リリース(開発ツール・ドキュメント)
- 2005 年 8 月　0.9 リリース(完成バージョン)
- 2005 年 9 月　1.0 リリース(最終バージョン)

1.1. アーキテクチャ

JDNC は以下の様に 3 つのレイヤーに分かれています。①JDNC Markup Language レイヤーから使用することでXMLから UI を作成できますが、②JDNC API レイヤーから使用することで拡張された Swing コンポーネントを直接利用することも出来ます。

①JDNC Markup Language
②JDNC API
③Swing 拡張

① JDNC Markup Language

UI の定義だけでなく、UI と非 UI コンポーネントのための拡張可能な XML。

② JDNC API

Swing を拡張したコンポーネントを操作する API を提供する。

③ Swing 拡張

Swing のコンポーネントを JavaBeans として扱えるように拡張。

2. JDNC による UI 開発

では、さっそく JDNC による UI 開発の流れをサンプル元に見てみましょう。

2.1. 環境設定

まずは JDNC の実行に必要な環境を以下から入手し、任意のディレクトリに解凍します。

https://JDNC.dev.java.net/servlets/ProjectDocumentList

本資料作成時の環境を以下に示します。

バージョン	OS	WindowsXP XP2
	JavaVM	J2SE 1.4.2_06
	JDNC	JDNC-0_6-2004_12_02
環境変数	JAVA_HOME	C:¥j2sdk1.4.2_06

2.2. サンプルから学ぶ

本章では JDNC についてくるサンプルを実行しながら JDNC ファイルの定義方法を見て行きます。

サンプルを実行するには JDNC を解凍したディレクトリに移動し、以下のコマンドを実行します。

```
run.bat demo¥simpleTable.jdnc info
```

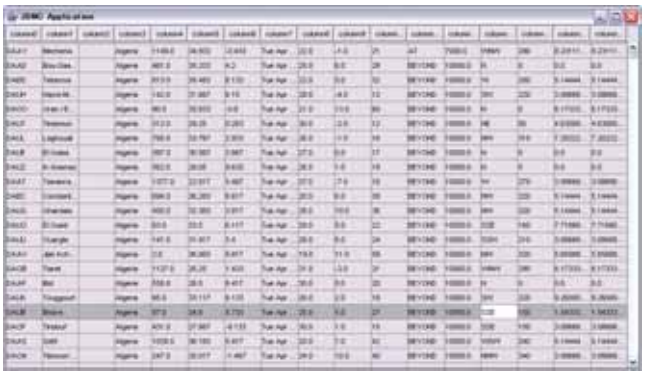
第一引数は読み込む JDNC ファイル、第二引数はログレベルを指定します。(第二引数は省略可能)

なお、ここで実行するサンプルは以下のデモサイトの***.jnlp ファイルを選択することで Java Web Start としても実行可能です。

http://javadesktop.org/jdnc/0_5/demo/

2.2.1. simpleTable(simpleTable.jdnc)

このサンプルはタブ区切りのファイル(demo¥data¥weather.txt)の内容を表示するシンプルなものですが、セルをダブルクリックすると編集、ヘッダをクリックすると並び替えが出来ます(保存は出来ません)。



XML としても非常にシンプルになっています(コメントは省略し、行番号を記述してあります)。

```
1:<?xml version='1.0'?>
2:<om:resource xmlns:om="http://www.openmarkup.net/2004/05/om"
3:             xmlns="http://www.jdesktop.org/2004/05/JDNC"
4:             xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
5:             xsi:schemaLocation="http://www.jdesktop.org/2004/05/JDNC schema/JDNC-1_0.xsd">
6:   <table>
7:     <tabularData source="data/weather.txt"/>
8:   </table>
9:</om:resource>
```

この JDNC ファイルのうち、テーブルの表示を行うための定義は 6～8 行目の 3 行のみです。6 行目の<table>タグは JNTable のインスタンスで表示することを意味し、7 行目の<tabularData>タグは JNTable で表示するデータソースを指定しています。

※JNTable は Swing の JTable を拡張したコンポーネントです。

2.2.2. betterTable(betterTable.jdnc)

betterTable では、simpleTable に以下の処理が追加されています。

- ハイライト表示の設定
- カラム名、データ型の設定
- セルへのイメージ設定、リストボックスによる選択(Wind Direction)
- ステータスバー
- ヘッダ

① ハイライト表示の設定

奇数行と偶数行とで行の背景色を変えます。

oddRowBackground で奇数行、evenRowBackground で偶数行の背景色を指定します。"white","red"のように SVG 形式で指定することも出来ます。

```
<highlighters>
    <alternateRowHighlighter oddRowBackground="#EEEEFF" evenRowBackground="#CCCCFF"/>
</highlighters>
```

奇数行の背景色　　偶数行の背景色

② カラム名、データ型の設定

<columnMetaData>タグの name 属性でカラム名、type 属性でデータ型を指定しています。

現在サポートされているデータ型は以下になります。

- string (デフォルト)
- float
- integer
- date
- href

<columnMetaData>タグで定義した name 属性を<column>タグの binding 属性で関連付け、title 属性でタイトルを設定します。<column>タグで定義されたカラムが順にテーブルに表示されます。

enumValues 属性については『③セルへのイメージ設定』を参照してください。prototypeValue では指定した文字列によってカラムの幅を指定します。

```
<tabularData source="data/weather.txt">
    <metaData>
        <columnMetaData name="ICAO"/>
        <columnMetaData name="STATION"/>
        <columnMetaData name="REGION"/>
        <columnMetaData name="COUNTRY"/>
        <columnMetaData name="ELEVATION" type="float"/>
         :
         :
        <columnMetaData name="GUST_SPEED" type="float"/>
    </metaData>
</tabularData>
 :
 :
<columns>
    <column title="Station" binding="STATION"/>
    <column title="Country" binding="COUNTRY" isReadOnly="true" enumValues="#flags" prototypeValue="this is a really really long field"/>      :
     :
    <column title="Gust Speed" binding="GUST_SPEED"  horizontalAlignment="center"/>
</columns>
```

データ型

カラムのタイトル

defs で定義された enumValues に対応

columnMetaData の name 属性に対応

③ セルへのイメージ設定

セルにイメージを設定するには以下の様に<defs>タグ内に<enumValues>タグを定義します。<defs>タグは、JDNC ファイル内で共有される属性を定義する場所です。今回の例では、<column>タグの enumValues から参照されています。<enumeration>タグで、セルの値(value 属性)に対応するテキスト(title 属性)と画像(icon 属性)、およびテキストと画像の表示位置(horizontalAlignment, horizontalTextPosition 属性)を指定します。

```
<om:defs>
    <enumValues xml:id="flags">
        <enumeration value="Afghanistan" title="Afghanistan"
            icon="images/flags/afghanistan.gif"
            horizontalAlignment="trailing"
            horizontalTextPosition="leading"/>
        <enumeration value="Albania" title="Albania"
            icon="images/flags/albania.gif"
            horizontalAlignment="trailing"
            horizontalTextPosition="leading"/>
             :
             :
    </enumValues>
</om:defs>
```

trailing or leading

trailing or leading

④ ステータスバー

ステータスバーはテーブルの下に表示され、行数と列数を表示します。

```
</rootPane>
     :
     :
    <statusBar/>
</rootPane>
```

⑤ ヘッダ

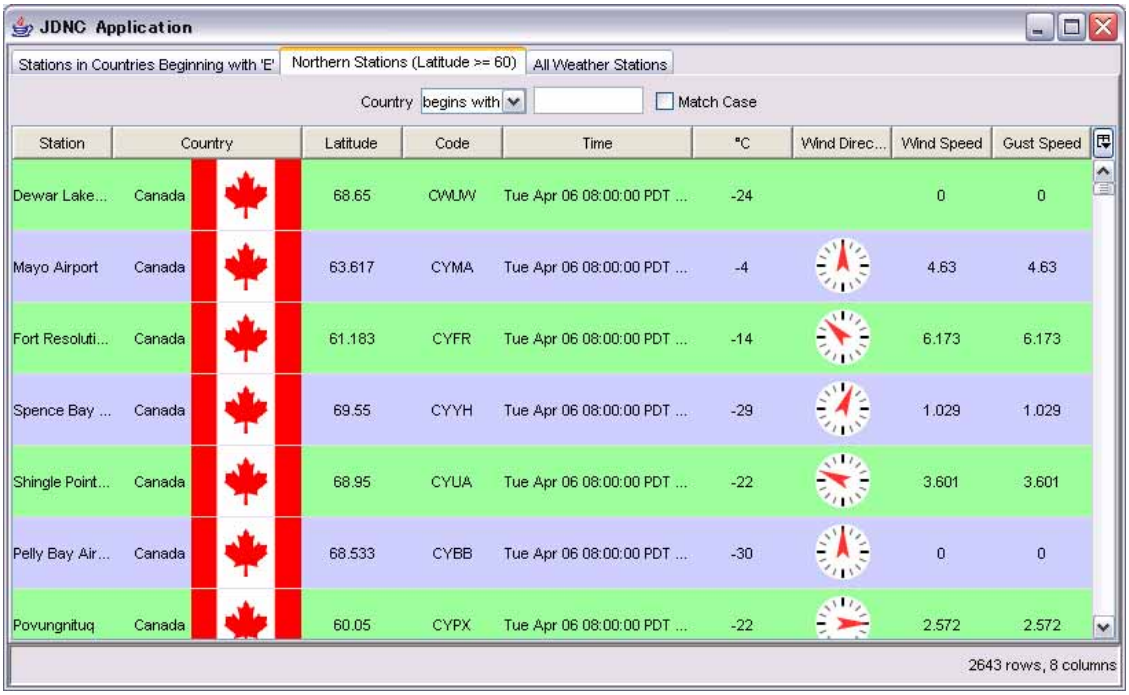
<header>タグではテーブルのヘッダに関する定義を行います。

```
    <table>
         :
        <header background="#E0E0D0"/>
         :
    </table>
```

ヘッダの背景色

2.2.3. tabTable(tabTable.jdnc)

tabTable ではタブで切り替えることにより、表示形式の異なる 3 パターンのテーブルを表示します。また"Northern Stations(Latitude >= 60"、"All Weather Stations"タグのテーブルでは動的にフィルタリングを行うことが出来ます。



① tabbedPane

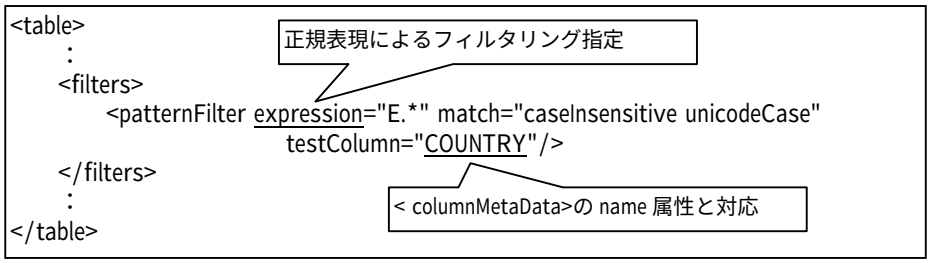
<tabbedPane>タグ内に定義したコンポーネントをタブで表示します。この例では<table>タグを 3 つ定義しているためタブを切り替えることによりそれぞれのテーブルが表示されます。

```
<rootPane>
    <tabbedPane>
        <table title="Stations in Countries Beginning with 'E'"...>
         :
        </table>
        <table title="Northern Stations (Latitude >= 60)"...>
         :
        </table>
        <table title="All Weather Stations"...>
         :
        </table>
    </tabbedPane>
</rootPane>
```

② 静的データフィルタ

テーブルに表示するデータを静的にフィルタリングします。match には以下の値が一つ以上指定可能です。各値の詳細は java.util.regex.Pattern の APIドキュメントを参照してください。

- caseInsensitive
- multilane
- dotAll
- unicodeCase
- canonEq



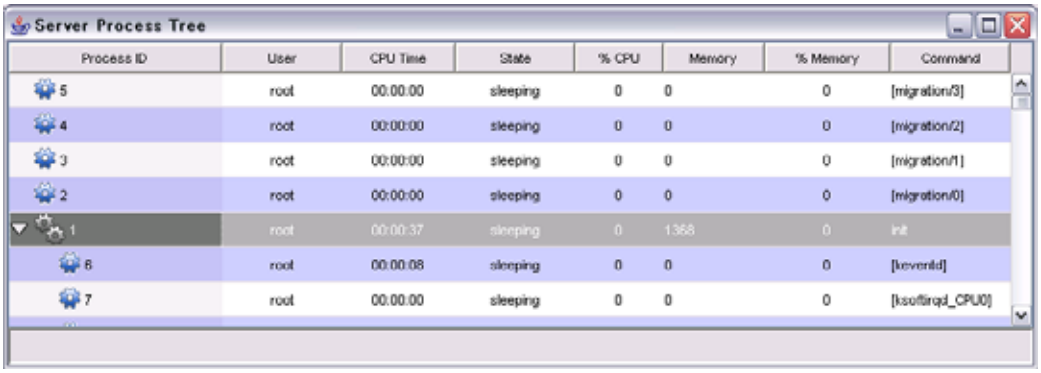
③ 動的データフィルタ

ユーザの入力によって、動的にパターンフィルタを切り替えます。<toolBar>タグ内の<searchPanel>によってテーブルの上にフィルタリング用のコンボボックス、入力フィールド等が表示されます。

```
<table>
    :
    <filters>
        <patternFilter xml:id="countryFilter" testColumn="COUNTRY"/>
    </filters>
    <toolBar>
        <searchPanel patternFilter="#countryFilter"/>
    </toolBar>
    :
</table>
```

2.2.4. treeTable(treeTable.jdnc)

ここまでは JNTable についてのサンプルでしたが、今回のサンプルは JNTreeTable を使用したものです。JNTreeTable は JNTable のサブクラスであり、内部的には JNTable 同様に Swing の JTable を使用しています。



① タイトル表示

<app>タグ内でタイトル、バージョンを指定し、<rootPane>タグの app 属性で<app>タグを参照しています。

```
<om:defs>
    <app xml:id="pstree" title="Server Process Tree" versionString="1.0"/>
    :
</om:defs>
<rootPane app="#pstree">
```

② ツリーテーブルの表示

<treeTable>タグでツリーテーブルに関する定義を行います。

```
<rootPane app="#pstree">
    <treeTable preferredSize="704 528" columnMargin="0"
        rowHeight="28" source="data/pstree.txt"
        expandedIcon="images/close.png" collapsedIcon="images/open.gif"
        containerIcon="images/gears.png" leafIcon="images/gear.gif">
    :
    </treeTable>
</rootPane>
```

データソースの指定

ツリーが開いているきのアイコン

ツリーが子要素を持たない場合のアイコン

ツリーが閉じられているときのアイコン

ツリーが子要素を持つ場合のアイコン

③ ハイライト表示

<hierarchicalColumnHighlighter>タグは階層化を強調するために使用します。今回の例では[Process ID]の列の背景色が暗くなっています。

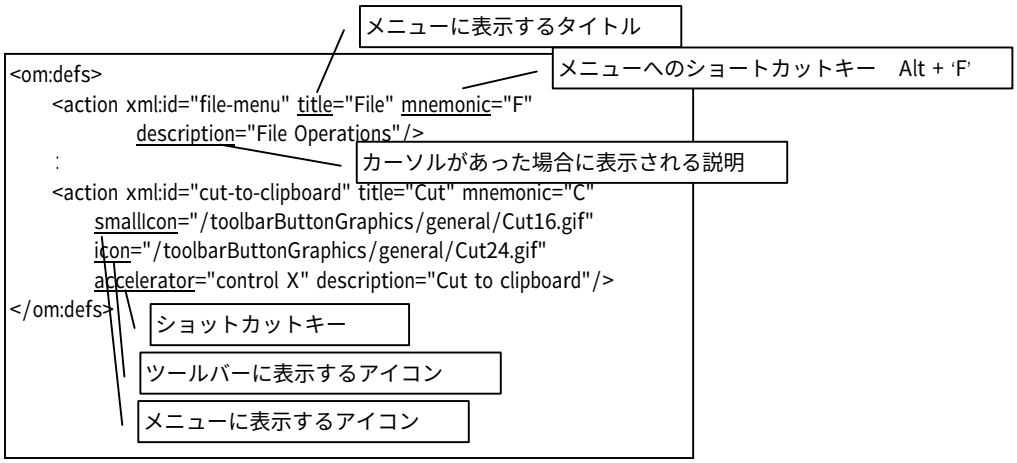
```
<highlighters>
    <alternateRowHighlighter evenRowBackground="#CCCCFF"/>
    <hierarchicalColumnHighlighter/>
</highlighters>
```

(2) editorDemo(editorDemo.jdnc)

editorDemo は JDNC を使用したテキストエディタのサンプルで、Swing の JEditorPane を拡張したコンポーネントを使用しています。ここでも新たに出てきたタグを中心に説明します。

① アクション定義

<action>タグで使用するアクションを定義します。以下には出てきませんが group 属性を指定すると同一グループ内での複数選択を出来ないようにします。editrDemo では group="paragraph-align"を使用してテキストの左寄せ、中央寄せ、右寄せアクションの排他制御を行っています。



標準で以下のアクションに対応しています。

- cut-to-clipboard　　切り取り
- copy-to-clipboard　　コピー
- paste-from-clipboard　　貼り付け
- undo　　操作の取り消し
- redo　　undo の取り消し
- find　　検索
- print　　印刷
- about　　説明の表示

定義したアクションはメニューバー、ツールバー、ポップアップメニューから参照される形で利用します。

メニューバー

```
<menuBar>
     :
    <menu title="Format" mnemonic="O" description="Text Formatting Commands">
         <action actionRef="#font-bold"/>
          :
         <separator/>
         <action actionRef="#left-justify"/>
          :
    </menu>
</menuBar>
```

セパレータを設定

ツールバー

```
<toolBar>
     <action actionRef="#font-bold"/>
      :
     <separator/>
     <action actionRef="#cut-to-clipboard"/>
      :
</toolBar>
```

ポップアップメニュー（エディタを右クリックした際に表示）

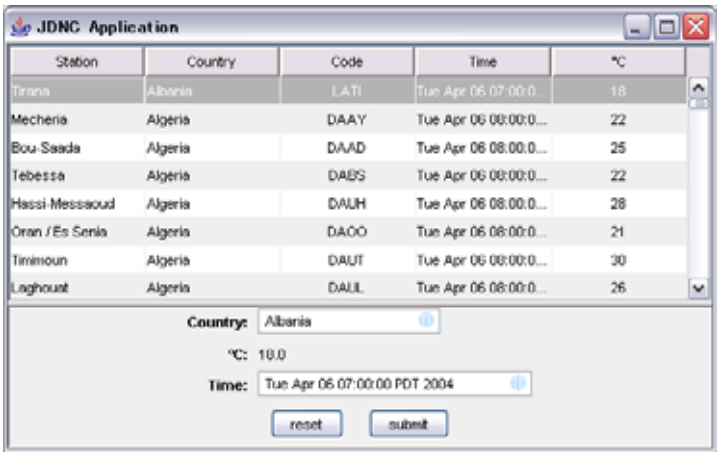
```
<popupMenu>
     <action actionRef="#cut-to-clipboard"/>
      :
</popupMenu>
```

2.2.5. formDemo(formDemo.jdnc)

今までの例はデータを表示するのみでしたが、formDemo ではボタン、リストボックスが配置されています。より実用に近いサンプルになっていますが、こちらは今後大きく変るようですので参考程度としてみてください。

なお、formDemo はそのままでは日付データの変換に失敗し、テーブルにデータが表示されません。以下の様に"TIMESTAMP"の type を"date"から"string"に変更するとデータが表示されます。

```
<columnMetaData name="TIMESTAMP" type="string" title="Time">
```



<splitPane>でテーブルとフォームを上下に分割し、フォーム内には<components>タグを使用してコンポーネントを配置しています。reset、submit ボタンは<form>タグにより自動で配置されています

```
<splitPane orientation="vertical" background="#EEEEEE">
    <table xml:id="weathertable" data="#weatherdata" title="Weather Stations"
        showsHorizontalLines="false" showsVerticalLines="false" selectionMode="single"
        preferredSize="550 200" background="#EEEEEE">
    :
    </table>
    <form data="#weatherdata" tracks="#weathertable" background="#EEEEEE">
        <components>
            <component binding="COUNTRY"/>
            <component binding="TEMPERATURE"/>
            <component binding="TIMESTAMP"/>
        </components>
    </form>
</splitPane>
```

2.2.6. その他

① パターン指定によるハイライト表示

以下の<patternHighlighter>タグではカラム"COUNTRY"のデータのうち、expression 属性で指定した正規表現と一致する(BorM で始まる)セルの背景色を赤にします。ここでの正規表現は J2SE に含まれる java.util.regex で利用可能な正規表現です。

```
<highlighters>
    <alternateRowHighlighter oddRowBackground="#EEEEFF"
                             evenRowBackground="#CCCCFF"/>
    <patternHighlighter expression="[B,M].*" testColumn="COUNTRY"
            foreground="red"/>
</highlighters>
```

3. 参考資料

- Sun Enterprise News[JavaOne 2004 レポート](http://sdc.sun.co.jp/news/2004/08/feature040801.html)
- JDNC プロジェクトページ(https://JDNC.dev.java.net/)
- JDNC スキーマ(http://javadesktop.org/jdnc/0_5/docs/schema/index.html)
- JDNC API ドキュメント(http://javadesktop.org/jdnc/0_5/docs/api/index.html)

4. 最後に

サンプルを中心に JDNC の UI 開発を見てきましたが、いかがでしたでしょうか。このように UI 開発に頻繁に使用される機能を共通コンポーネントとして用意し、XML から生成することで UI 開発を簡単にしようというのが JDNC の目指すゴールです。

ユーザにとってメリットの多いリッチクライアントですが、UI 部分の開発コストがネックとなり、導入を見合わせている事例が多く存在します。オープンな技術である JDNC が普及すれば、UI 開発のコストダウンが実現され、リッチクライアント導入もより加速することが予想されます。

実用にはもうしばらく時間がかかりそうな JDNC ですが、今後も状況を見つつ本資料も更新して行こうと思います。

記述内容に何かお気づきの点、質問等ありましたら下記までご連絡ください。

開発部
横井 朗
yokoi@bbreak.co.jp