平安京ビュー

As of 2007/06/04
伊藤貴之
お茶の水女子大学 理学部情報科学科

当文書は、階層型データ可視化手法「平安京ビュー」を動かすための、最低限のインストール方法、操作方法、入力データファイル記述方法を規定したものである。質問、バグ報告、機能拡張要求、などは全て伊藤貴之 (itot@computer.org) がメールで対応するものとする。

平安京ビューのインストール方法

(1) JDK (Java Development Kit) および Eclipse をインストールする。伊藤は JDK 1.5.0 および Eclipse 3.1.2 を使っている。インストール方法については各々の文書を参考のこと。

(2) Eclipse の上で平安京ビューをインストールするための準備作業をする。伊藤から平安京ビューのプログラムを受け取ったら、このプログラムのトップディレクトリである src を、あるディレクトリの下に置く。本文書では Windows を使っていることを仮定して、以下 c:¥heiankyoview の下に src ディレクトリを置くと仮定する。

(3) メニューで「File」の「New」の「Project」を選択すると、ダイアログウィンドウが表示されるので、左側「Java」、右側「Java Project」を選択し、Next ボタンを押す。

(4) Project Name 欄に、適当な名前をつける。また Project contents という行の下にある「use defalut」という欄をクリックして、チェックマークがつかない状態にする。また Browse ボタンを押して、(2)で作成したディレクトリ（本文書の例では c:¥heiankyoview）を選択する。以上の操作が終わったら Next ボタンを押す。

(5) Source folder on build path という欄に、(4)で選んだディレクトリの下の src (本文書の例では c:¥heiankyoview¥src) が表示されていることを確認する。また Default output folder という欄に、(4)で選んだディレクトリの下の bin (本文書の例では c:¥heiankyoview¥bin) が表示されていることを確認する。以上を確認したら Finish ボタンを押す。これでインストール終了。

サンプルアプリケーション HeiankyoView の使用方法

(1) パッケージ org.heiankyoview2.applet.heiankyoview の中にある HeiankyoView クラスを Java Applet として起動する。起動方法は以下の通り。

① Eclipse のメニューで「Run」の「Run」を選択する。

② 左側「Java Applet」を選び、「New」ボタンを押す。

③　右側のプロジェクト欄に、1 章の(2)で選んだ名前を選択する。

④　右側の「Search」ボタンを押し、HeiankyoView を選択する。

⑤　下の「Run」ボタンを押す。これで HeiankyoView を起動できる。

(2) メニュー「File - Open」を選択して、入力ファイルを選択する。サンプルファイルの作成方法は 3 章を参照。

(3) 視点を操作するときは、画面右側の各ボタンを押して、画面をドラッグ（＝マウスのボタンを押しながら上下左右に動かす）する。

a. 視点をリセットしたいとき …「Viewing reset」を押す。

b. 回転したいとき … 「Rotate」を押す。

c. 拡大縮小したいとき … 「Scale」を押す。

d. 平行移動したいとき … 「Shift」を押す。

(4) ファイルを読んだ後にメニュー「Appearance – TableAttribute」を選択すると、入力ファイルに記述した属性の名前を表示したダイアログウィンドウが立ち上がる。

- Name 行に表示された任意の属性を押した後に、ある属性を選択して、表示されたノードをクリックすると、そのノードに与えられた属性の名前が表示される。

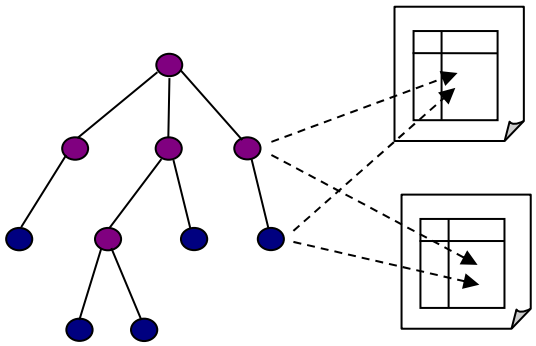
- Color 行に表示された任意の属性を押すと、その属性から算出された色が各ノードに与えられる。

- Height 行に表示された任意の属性を押すと、その属性から算出された高さが各ノードに与えられる。

(5) ファイルを読んだ後にメニュー「Appearance - Appearance」を選択すると、表示される線分の太さ、背景色、文字表示の ON/OFF、などを設定するダイアログウィンドウが起動する。

データ構造

「平安京ビュー」は Tree というクラスによって階層型データの全体像を格納する。このデータ構造について論じる。



「平安京ビュー」では、上図に示すような階層型データを想定している。

この図において、左側が木構造を表している。紫色で塗った頂点のように、下に頂点がぶら下がっているものを、一般的に「枝ノード」と呼び、「平安京ビュー」でも Branch とい

うクラスでこれを表現する。紺色で塗った頂点のように、下に頂点がぶら下がってないものを、一般的に「葉ノード」と呼び、「平安京ビュー」では Node というクラスでこれを表現する。

また、この図において、右側は枝ノードおよび葉ノードの属性値を記述する表である。この表を「平安京ビュー」では Table というクラスで表現する。

また、Table というクラスとは別に、時系列情報を記述する表を、「平安京ビュー」では Frame というクラスで表現する。

tree ファイルの書式

「平安京ビュー」が前提としている木構造データのファイル書式（tree ファイル）では、現在以下のタグをサポートしている。

属性データに関するタグ

- numtable
  属性を記述する表の数を定義する。
- tablename
  属性を記述する 1 個の表の題名を定義する。
- tabletype
  属性を記述する 1 個の表のデータタイプを定義する。現状では string(文字列) double(倍精度小数) int(整数)の 3 タイプのいずれかを選択する。
- tablenumline
  属性を記述する 1 個の表の行数を定義する。
- tableline (tl と略してもよい)
  属性を記述する 1 個の表の 1 行に相当する内容を定義する。

木構造データに関するタグ

- numbranch
  最初の 1 回は、Branch の総数を定義するために用いる。それ以降は、1 個の Branch に対する子供 Branch の総数を定義するために用いる。(同一タグを 2 つの意味で用いているのは、あまり好ましくない気もするが、特に修正の必要もないのでそのままにしている。)
- branch
  1 個の Branch に関する記述がここから始まるという宣言をする。
- branchtableline (btl と略してもよい)
  1 個の Branch の 1 種類の属性を記述する。
- numnode
  1 個の Branch が持つ Node の総数を記述する。
- nodetableline (ntl と略してよい)
  1 個の Node の 1 種類の属性を記述する。
- childbranch
  Branch がもつ子供 Branch の 1 個がどれであるかを記述する。

時系列データに関するタグ

-framenodeid (fni と略してもよい)

frame ファイルにおける framenode タグの ID がどれであるかを特定する。

-frameimport

読み込むべき frame ファイルが特定されている場合には、そのファイル名を特定する。

「平安京ビュー」が前提としている時系列データのファイル書式（frame ファイル）では、現在以下のタグをサポートしている。

時系列データに関するタグ

-numframe

Frame の総数を示す。

-numnode

Frame に対応する Node の総数を示す。

-numdimension

Frame に対応する各 Node が所有する変数の次元数を示す。

-frame

Frame の ID と時刻を表す。

-framenode(fn と略してもよい)

対応する 1 個の Node における変数を示す。

サンプルファイル(1) Table を有し Frame を有さない例

numtable 3 ←属性を記述する表は 3 個ある。2 行目に必ず記入。

tablename 1 instrumentName ←1 個目の表の題名は instrumentName である。
tabletype 1 string ←1 個目の表のデータ形式は string(文字列型) である。
tablenumline 1 3 ←1 個目の表の行数は 3 である。
tableline 1 trumpet ←1 行目は trumpet である。
tableline 2 trombone ←2 行目は trombone である。
tableline 3 horn ←3 行目は horn である。

tablename 2 price ←2 個目の表の題名は price である。
tabletype 2 int ←2 個目の表のデータ形式は int(整数型) である。
tablenumline 2 2 ←2 個目の表の行数は 2 である。
tableline 1 198000 ←1 行目は 198000 である。
tableline 2 233000 ←2 行目は 233000 である。

tablename 3 tone ←3 個目の表の題名は tone である。
tabletype 3 double ←3 個目の表のデータ形式は double(倍精度小数型) である。
tablenumline 3 4 ←3 個目の表の行数は 4 である。
tableline 1 100.34 ←1 行目は 100.34 である。
tableline 2 302.31 ←2 行目は 302.31 である。
tableline 3 543.03 ←3 行目は 543.03 である。
tableline 4 1198.12 ←4 行目は 1198.12 である。

numbranch 3 ←Branch の総数は 3 である。

branch 1 ←ここから 1 番目の Branch に関する記述を開始する。
numnode 1 1 ←1 番目の Branch の Node の総数は 1 である。
numbranch 1 2 ←1 番目の Branch の子 Branch の総数は 2 である。
branchtableline 1 1 1 ←1 個目の Branch の 1 個目の表の属性は 1(trumpet)である。

nodetableline 1 1 1 ←1 個目の Node の 1 個目の表の属性は 1(trumpet)である。
nodetableline 1 2 2 ←1 個目の Node の 2 個目の表の属性は 2(233000)である。
nodetableline 1 3 2 ←1 個目の Node の 3 個目の表の属性は 2(302.31)である。

childbranch 1 2 ←1 個目の子 Branch は 2 番目の Branch である。
childbranch 2 3 ←2 個目の子 Branch は 3 番目の Branch である

branch 2　←ここから 2 番目の Branch に関する記述を開始する。
numnode 2 4　←2 番目の Branch の Node の総数は 4 である。
numbranch 2 0　←2 番目の Branch の子 Branch の総数は 0 である。

nodetableline 1 1 1 ←1 個目の Node の 1 個目の表の属性は 1(trumpet)である。
nodetableline 1 2 1 ←1 個目の Node の 2 個目の表の属性は 1(198000)である。
nodetableline 1 3 1 ←1 個目の Node の 3 個目の表の属性は 1(100.34)である。

nodetableline 2 1 1 ←2 個目の Node の 1 個目の表の属性は 1(trumpet)である。
nodetableline 2 2 1 ←2 個目の Node の 2 個目の表の属性は 1(198000)である。
nodetableline 2 3 2 ←2 個目の Node の 3 個目の表の属性は 2(302.31)である。

nodetableline 3 1 2 ←3 個目の Node の 1 個目の表の属性は 2(trombone)である。
nodetableline 3 2 1 ←3 個目の Node の 2 個目の表の属性は 1(198000)である。
nodetableline 3 3 3 ←3 個目の Node の 3 個目の表の属性は 3(543.03)である。

nodetableline 4 1 3 ←4 個目の Node の 1 個目の表の属性は 3(horn)である。
nodetableline 4 2 2 ←4 個目の Node の 2 個目の表の属性は 2(233000)である。
nodetableline 4 3 4 ←4 個目の Node の 3 個目の表の属性は 4(1198.12)である。

branch 3　←ここから 3 番目の Branch に関する記述を開始する。
numnode 2 1　←3 番目の Branch の Node の総数は 1 である。
numbranch 2 0　←3 番目の Branch の子 Branch の総数は 0 である。
branchtableline 3 1 3 ←1 個目の Branch の 1 個目の表の属性は 3(horn)である。

nodetableline 1 1 3 ←1 個目の Node の 1 個目の表の属性は 3(horn)である。
nodetableline 1 2 1 ←1 個目の Node の 2 個目の表の属性は 1(198000)である。
nodetableline 1 3 4 ←1 個目の Node の 3 個目の表の属性は 4(1198.12)である。

サンプルファイル(2) Table を有さず Frame を有する例

numtable 0 ←属性を記述する表はない。

numbranch 3　←Branch の総数は 3 である。

branch 1　←ここから 1 番目の Branch に関する記述を開始する。
numnode 1 1 ←1 番目の Branch の Node の総数は 1 である。
numbranch 1 2 ←1 番目の Branch の子 Branch の総数は 2 である。

framenodeid 1 1 ←1 個目の Node の Frame ID は 1 である。

childbranch 1 2 ←1 個目の子 Branch は 2 番目の Branch である。
childbranch 2 3 ←2 個目の子 Branch は 3 番目の Branch である

branch 2　←ここから 2 番目の Branch に関する記述を開始する。
numnode 2 4　←2 番目の Branch の Node の総数は 4 である。
numbranch 2 0　←2 番目の Branch の子 Branch の総数は 0 である。

framenodeid 1 2 ←1 個目の Node の Frame ID は 2 である。
framenodeid 2 3 ←2 個目の Node の Frame ID は 3 である。
framenodeid 3 4 ←3 個目の Node の Frame ID は 4 である。
framenodeid 4 5 ←4 個目の Node の Frame ID は 5 である。

branch 3　←ここから 3 番目の Branch に関する記述を開始する。
numnode 2 1　←3 番目の Branch の Node の総数は 1 である。
numbranch 2 0　←3 番目の Branch の子 Branch の総数は 0 である。

framenodeid 1 6 ←1 個目の Node の Frame ID は 6 である。

--------(ここから下、frame ファイルとして切り分けることができる)-------

numframe 2 ←Frame の総数は 2 である。
numnodes 6 ←Node の総数は 2 である。
numdimension 3 ←Node が所有する変数の次元数は 3 である。

frame 1 0.0 ←1 番目の Frame の時刻は 0.0 である。
framenode 1 0.0 1.0 0.0 ←1 番目の Node の変数値は(0.0, 1.0, 0.0)である。
framenode 2 0.0 2.0 0.0 ←2 番目の Node の変数値は(0.0, 2.0, 0.0)である。
framenode 3 1.0 1.0 0.0 ←3 番目の Node の変数値は(1.0, 1.0, 0.0)である。
framenode 4 1.0 2.0 0.0 ←4 番目の Node の変数値は(1.0, 2.0, 0.0)である。
framenode 5 1.0 1.0 1.0 ←5 番目の Node の変数値は(1.0, 1.0, 1.0)である。
framenode 6 1.0 2.0 1.0 ←6 番目の Node の変数値は(1.0, 2.0, 1.0)である。

frame 2 1.0 ←2 番目の Frame の時刻は 1.0 である。
framenode 1 3.0 1.0 0.0 ←1 番目の Node の変数値は(3.0, 1.0, 0.0)である。
framenode 2 3.0 2.0 0.0 ←2 番目の Node の変数値は(3.0, 2.0, 0.0)である。
framenode 3 4.0 1.0 0.0 ←3 番目の Node の変数値は(4.0, 1.0, 0.0)である。
framenode 4 4.0 2.0 0.0 ←4 番目の Node の変数値は(4.0, 2.0, 0.0)である。
framenode 5 5.0 1.0 1.0 ←5 番目の Node の変数値は(5.0, 1.0, 1.0)である。
framenode 6 5.0 2.0 1.0 ←6 番目の Node の変数値は(5.0, 2.0, 1.0)である。

サンプルファイルを自分でつくってみる

「平安京ビュー」では tree ファイルというファイル形式を読み込むことができる。ここでは一例として、自分のファイルシステムの中身を視覚化するためのデータファイルを作成するサンプルプログラムを紹介する。

(1) あるディレクトリ上で例えば、「tar cvf ../xxx.tar .」というコマンドを実行し、そのディレクトリの下にあるファイルを全部まとめて 1 ファイルに格納する xxx.tar ファイルを作成する。

(2) xxx.tar ファイルの置いてあるディレクトリ上で例えば、「tar tvf xxx.tar > xxx.txt」というコマンドを実行し、xxx.tar ファイルの中に格納されたファイルの一覧表示結果を xxx.txt というテキストファイルに保存する。

(3) 平安京ビューに付属している Tar2Tree というサンプルプログラムを起動し、xxx.txt ファイルを tree ファイルに変換する。方法は以下の通り。

① Eclipse のメニューで「Run」の「Run」を選択する。
② 左側「Java Application」を選び、「New」ボタンを押す。
③ 右側のプロジェクト欄に、1 章の(2)で選んだ名前を選択する。
④ 右側の「Search」ボタンを押し、Tar2Tree を選択する。
⑤ Arguments というタブを選択し、Program Arguments という欄に、(2)で作成した txt ファイル、これから作成したい tree ファイル、txt ファイルの行数より大きい正整数、の順に、空白をいれて記述する。このとき例えば、「c:¥heiankyoview¥xxx.txt c:¥heiankyoview¥xxx.tree 10000」というように、txt ファイルと net ファイルは絶対パスで記述したほうが確実である。
⑥ 下の「Run」ボタンを押す。これで Tar2Tree を起動できる。

なお伊藤は、Windows 上で動く cygwin という環境上にある tar コマンドを使っている。これ以外の環境での tar コマンドでの動作確認はしていない。

XML による「平安京ビュー」対応データの作成

「平安京ビュー」は以下のタグによって記述された XML 文書を入力できる。

<tree>

表示対象となる木構造全体を括るためのタグ。このタグの内部に直接内包できるタグは<branch><tables>を 1 個ずつのみであり、他のタグを内包してはいけない。

<branch>

「平安京ビュー」の画面上で 1 個の枠に相当する、階層型データの一階層の情報を括るためのタグ。このタグの内部に直接内包できるタグは、1 個以上の<branch>、1 個以上の<node>、1 個以上の<tablepointer>、1 個の<framepointer>だけであり、他のタグを内包してはいけない。

<node>

「平安京ビュー」の画面上で 1 個の棒グラフに相当する、階層型データの情報の最小単位を括るためのタグ。このタグの内部に直接内包できるタグは、1 個以上の<tablepointer>だけであり、他のタグを内包してはいけない。

<tables>

属性表の集合を括るためのタグ。このタグの内部に直接内包できるタグは、1 個以上の<table>だけであり、他のタグを内包してはいけない。

<table>

1 個の属性表を括るためのタグ。このタグには attribute として name と type を含めなければならない。name の値は、複数の table に対して同一であってはならない。type の値は、string, double, int のいずれかでなければならない。このタグに直接内包できるタグは、1 個以上の<tableline>だけであり、他のタグを内包してはいけない。

<tableline>

1 個の属性表の中の 1 行に相当する情報を括るためのタグ。このタグには attribute として id と value を含めなければならない。id の値は、現状では 1 からの通し番号でなければならない。value の値は、table タグの type で指定される string(文字列)、double(倍精度浮動小数)、int(整数)、のいずれかに整合していなければならない。このタグに別のタグを内包してはならない。

<tablepointer>

branch タグおよび node タグから、属性表の 1 行を指定するためのタグ。このタグには attribute として tablename, tablelineid を含めなければならない。tablename の値は、table タグの name の値に指定されているものと同一でなければならない。tablelineid の値は、tablename の値によって特定される table タグに内包される tableline タグのいずれかの id の値と同一でなければならない。このタグに別のタグを内包してはならない。

<framepointer>

node タグから、時系列情報の 1 行を指定するためのタグ。このタグには attribute として

id を含めなければならない。id の値は、framenode タグのいずれかの id の値と同一でなければならない。このタグに別のタグを内包してはならない。

<frames>

表示対象となる時系列情報全体を括るためのタグ。このタグには attribute として numnode と numdimension を含めなければならない。numnode および numdimension の値は正の整数でなければならない。このタグの内部に直接内包できるタグは 1 個以上の<frame>だけであり、他のタグを内包してはいけない。

<frame>

1 時刻の情報を括るためのタグ。このタグには attribute として name と time を含めなければならない。name の値は、複数の frame に対して同一であってはならない。time の値は、実数でなければならない。このタグに直接内包できるタグは、1 個以上の<framenode>だけであり、他のタグを内包してはいけない。

<framenode>

1 時刻における 1 個の Node の情報を括るためのタグ。このタグには attribute として id と value を含めなければならない。id の値は、現状では 1 からの通し番号でなければならない。value の値は、frames タグの numdimension 値で指定された個数の実数であり、実数と実数の間はカンマで区切らなければならない。このタグに別のタグを内包してはならない。

サンプルファイル(1) Table を有し Frame を有さない例

```
<tree>
 <tables>
  <table name=”instrumentName” type=”string”>
   <tableline id=”1” value=”trumpet” />
   <tableline id=”2” value=”trombone” />
   <tableline id=”3” value=”horn” />
  </table>
  <table name=”price” type=”int”>
   <tableline id=”1” value=”198000” />
   <tableline id=”2” value=”233000” />
  </table>
  <table name=”tone” type=”double”>
   <tableline id=”1” value=”100.34” />
   <tableline id=”2” value=”302.31” />
   <tableline id=”3” value=”543.03” />
   <tableline id=”4” value=”1198.12” />
  </table>
 </tables>
 <branch>
  <tablepointer tablename=”instrumentName” tablelineid=”1” />
  <node>
   <tablepointer tablename=”instrumentName” tablelineid=”1” />
   <tablepointer tablename=”price” tablelineid=”2” />
   <tablepointer tablename=”tone” tablelineid=”2” />
  </node>
  <branch>
   <tablepointer tablename=”instrumentName” tablelineid=”2” />
   <node>
    <tablepointer tablename=”instrumentName” tablelineid=”1” />
    <tablepointer tablename=”price” tablelineid=”1” />
    <tablepointer tablename=”tone” tablelineid=”1” />
   </node>
   <node>
    <tablepointer tablename=”instrumentName” tablelineid=”1” />
```

```
      <tablepointer tablename="price" tablelineid="1" />
      <tablepointer tablename="tone" tablelineid="2" />
    </node>
    <node>
      <tablepointer tablename="instrumentName" tablelineid="2" />
      <tablepointer tablename="price" tablelineid="1" />
      <tablepointer tablename="tone" tablelineid="3" />
    </node>
    <node>
      <tablepointer tablename="instrumentName" tablelineid="3" />
      <tablepointer tablename="price" tablelineid="2" />
      <tablepointer tablename="tone" tablelineid="4" />
    </node>
  </branch>
  <branch>
    <tablepointer tablename="instrumentName" tablelineid="3" />
    <node>
      <tablepointer tablename="instrumentName" tablelineid="3" />
      <tablepointer tablename="price" tablelineid="1" />
      <tablepointer tablename="tone" tablelineid="4" />
    </node>
  </branch>
 </branch>
</tree>
```

サンプルファイル(2) Table を有さず Frame を有する例

```
<tree>
  <branch>
    <node>
      <framepointer id=”1” />
    </node>
    <branch>
      <node>
        <framepointer id=”2” />
      </node>
      <node>
        <framepointer id=”3” />
      </node>
      <node>
        <framepointer id=”4” />
      </node>
      <node>
        <framepointer id=”5” />
      </node>
    </branch>
    <branch>
      <node>
        <framepointer id=”6” />
      </node>
    </branch>
  </branch>
</tree>
```

------(ここから下、別ファイルとして切り分けることが望ましい)------

```
<frames numdimension=”3”>
  <frame name=”1” time=”0.0”>
    <framenode id=”1” value=” 0.0,1.0,0.0”>
    <framenode id=”2” value=” 0.0,2.0,0.0”>
    <framenode id=”3” value=” 1.0,1.0,0.0”>
```

```
    <framenode id=”4” value=” 1.0,1.0,0.0”>
    <framenode id=”5” value=” 1.0,1.0,1.0”>
    <framenode id=”6” value=” 1.0,2.0,1.0”>
  </frame>
  <frame name=”2” time=”1.0”>
    <framenode id=”1” value=” 3.0,1.0,0.0”>
    <framenode id=”2” value=” 3.0,2.0,0.0”>
    <framenode id=”3” value=” 4.0,1.0,0.0”>
    <framenode id=”4” value=” 4.0,1.0,0.0”>
    <framenode id=”5” value=” 5.0,1.0,1.0”>
    <framenode id=”6” value=” 5.0,2.0,1.0”>
  </frame>
</frames>
```