Visual PlanIt Quick-Start Tutorial

Visual PlanIt is, among other things, a library of visual controls for creating PIM style applications. It consists of a number of visual controls as well as some specialized data-storage components, which work in conjunction to create powerful applications with a minimum amount of coding. Each section of this chapter provides one exercise that will help you to master the skills and understanding that you will need to effectively master the library, and all of its components.

The components, which make up the Visual PlanIt library are all documented in detail in other chapters of this manual. They are presented here in an abbreviated format, in order that you will be more familiar with them as you go through the tutorials.

Simple Visual PlanIt PIM Application

This is a simple *(*one line of source code*)* tutorial that will help you get up to speed with Visual PlanIt as quickly as possible. It is highly recommended that you read it completely before exploring the library, so that you will understand the basics of how to use the library. If you have any questions, please don't hesitate to visit the public Visual PlanIt newsgroup at news.turbopower.com.

Here's how to create a basic PIM style application:

Follow the steps completely before compiling and running the application for the first time.

1. Open your IDE and create a new application.

2. Drop a TPanel on the form and set its properties as follows:

```
Align             alTop
Caption           ''
```

3. Drop a TVpBDEDataStore component on the form and set its properties as follows:

```
AliasName         "Visual PlanIt"
AutoConnect       True
AutoCreate        True
AutoCreateAlias   True
ResourceID        0
```

4. Drop a ControlLink component on the form.

5. Drop a DayView component on the form and set its properties as follows:

```
Align              alLeft
ShowResourceName   False
```

Note: The DayView, WeekView, MonthView, ContactGrid, and TaskList are TVpLinkableControl descendants, so they will automatically connect to the DataStore and ControlLink when they are dropped on the form.

6. Drop a MonthView component on the form. Place it on the form, just to the right of the DayView, Even with the top of the DayView. Set its properties as follows:

```
DayNameStyle      dsShort
ShowEvents        False
```

7. Drop a ContactGrid component on the form, Place it just beneath the MonthView. There are no properties to set.

8. Drop a TaskList component on the form and set its properties as follows:

```
Align          alRight
```

9. Drop a TVpResourceCombo on the panel. (You may want to place a label next to it that says, “Resource”)

10. Drop a TVpResourceEditDialog component on the panel.

11. Drop a TButton on the panel, just to the right of the ResourceCombo, and set its properties as follows:

```
Caption          “Create a new resource”
Width            130
```

12. Define the TButton's OnClick event handler as follows (Note: This is the only source code in the tutorial):

Delphi:

```
procedure TForm1.Button1Click(Sender : TObject);
begin
  VpResourceEditDialog1.AddNewResource;
end;
```

C++Builder:

```
void __fastcall TForm1::Button1Click(TObject *Sender)
{
  VpResourceEditDialog1->AddNewResource();
}
```

Your application's only form will look something like Figure 1.1.

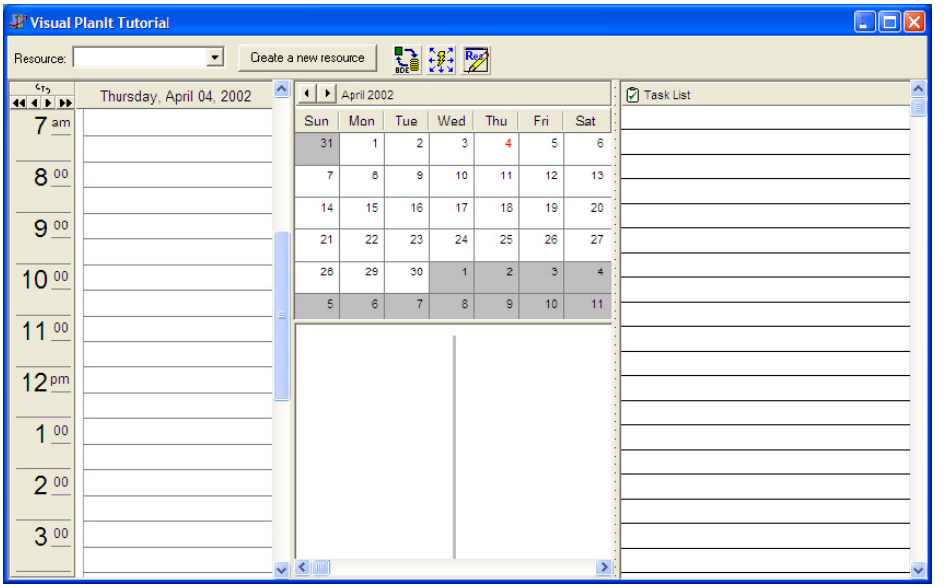

Figure 1.1: Tutorial Application's Main Form.

13. Make sure to save the project.

14. If you have the debugger set to stop on Delphi Exceptions (In Delphi 5 click Tools | Debugger Options click the "Language Exceptions" tab and it is the "Stop on Delphi Exceptions" check box at the bottom of the form.), you will see exceptions as the DataStore attempts to open each table. Click OK on each one and press F9. Each table will be created. There are 4 tables to create, so you will get 4 errors. These errors are caught at normal run time, you will only see them in the debugger.

    If you de-select the "Stop on Delphi Exceptions" check box, you won't get those exceptions in the debugger.

15. Compile and run the application.

The new database is empty so you will not be able to create any events, tasks, or contacts until you create a resource.

To explore how you can interact with the application, try the following:

- Double-click on the DayView to add a new event.
- Double-click on the TaskList to add a task.
- Double-click on the ContactGrid to add a contact
- Click around (and move around with the arrow keys) on the MonthView and notice how the DayView is in-sync (made possible by the work of the ControlLink.)
- Shut down the application and add a WeekView.
- Select a day in the past. Add a new event. Set its alarm. Save it. It should trigger a reminder dialog box as soon as the next minute passes. It will trigger the alert immediately if you shut down the application and re-start it.

Overview of Printing with Visual PlanIt

The Printer property of the TVpControlLink component maintains the print formats used. Visual PlanIt uses a sophisticated printing and page rendering mechanism that supports multiple print formats, each containing multiple print elements.

The print formats are associated with a TVpControlLink component. Each control link can have its own set of print formats. Print formats can either be created at design time, or created in an XML file and loaded at run time.

The basic structure is the TVpPrinter class (the Printer property of the TVpControlLink component) contains extra information needed for printing in addition to a PrintFormats property. The print format property contains a list of print formats. Each print format contains a list of components that are needed to print the page.

To access the print formats at design time, open up the Printer property on a TVpControlLink component. This part of the print format stores information that is needed, but not part of the print format. This information includes things like the start and end of days for TVpDayView components as well as the granularity of that component. Since the start and end times is more of a matter of personal preference as opposed to an integral part of the print format, the information is stored here. These fields are not saved or loaded in the XML file.

The PrintFormats property of the TVpPrinter class is where the actual definition of the print format starts. This is a TCollection in which every item in the TCollection is a print format. Each print format has several properties. The FormatName and Description properties give the format a name and an optional detailed description. The FormatName is used by other components to lookup the print format and is a required property. The DayInc and DayIncUnits properties are used to describe how many days are represented on a single page. Depending on the elements that are used in the print format, the number of days represented on a single page can vary greatly.

The actual print elements are accessed via the Elements property. The Elements property is also a TCollection in which each item represents an individual print element for this print format. The type of element to print is controlled by the ItemType property. Most of the item types match directly up to Visual PlanIt components. However, the itCaption and itShape item types are pseudo-elements used to add simple shapes and text to the print format. These properties are defined in the Shape and Caption properties of the element. The itCaption element also supports the insertion of variables. Several variables are automatically created that represent useful things about the page that is being printed. Examples of these are page numbers and dates (in various formats). Additional variables

can be added at run time and variables can even be modified while they are being looked up. The documentation on the TVpPrintCaption and TVpPrinter classes contains more information on the print caption variables.

The elements are positioned on the page using the Left, Top, Width, Height, and Measurement properties. Measurement defines how the Left, Top, Width, and Height properties will be used. By default, items are positioned using a percentage of the page. Elements can also be positioned using inches or pixels directly. As a rule of thumb, all elements in a print format should use the same value setting for the Measurement attribute (see Table 1.2 on page 8 for possible values).

If the element uses date information (like DayViews and WeekViews), the starting date of the element on the page is modifiable by using the DayOffset and DayOffsetUnits properties. This is useful in cases where multiple instances of the same element occur on the same page. For example, a print format consisting of two day views can use these properties so the second day view shows the day after the first day. When doing this, it is important to go back to the print format and adjust the DayInc and DayIncUnits properties to correctly reflect the number of days that is actually rendered.

The rotation property allows the element to be rotated in 90-degree increments.

When printing, the properties of components on the form will be used to determine how the component will be printed. If no component can be found, a default one will be created and the default values used.

Print format XML layout

The LoadFromFile and SaveToFile methods of the TVpPrinter class maintain the print formats as an XML file.

The root node of the XML file is the Visual PlanIt PrintFormats element. This element consists of a single attribute, Version, which contains the version level of the print formats. The version stamp consists of three digits separated by dots. The TVpPrintFormat class contains zero or more of the PrintFormat element.

The PrintFormat element has several attributes. These are listed in Table 1.1.

Table 1.1: *PrintFormat attributes*

Attribute	Meaning
DayIncrement	Numeric value of the number of days represented in a page. This value is modified by the DayIncrementUnits attribute.
DayIncrementUnits	The units that DayIncrement is measured in. Legal values for this attribute are Day, Week, Month or Year.
Description	Description of the print format.
Name	Name of the print format. This is required and must be unique.

The PrintFormat element can contain one or more Element elements. The attributes of the Element are listed in Table 1.2.

Table 1.2: *Element attributes*

Attribute	Meaning
DayOffset	The date offset used to render this element. This value is an integer number of days that will be modified by the DayOffsetUnits attribute.
DayOffsetUnits	The units that the DayOffset attribute will be measured in. Legal values for this attribute are Day, Week, Month or Year.
Height	Floating point value indicating the height of the rectangle the element will be rendered in. This value will be measured using the units specified in the Measurement attribute.
Item	The type of element to use. Legal values for this attribute are DayView, WeekView, MonthView, Calendar, Shape or Caption.
Left	Floating point value indicating the left side of the rectangle the element will be rendered in. This value will be measured using the units specified in the Measurement attribute.
Measurement	How the Left, Top, Width and Height elements will be measured. Legal values for this attribute are AbsolutePixel, Percent or Inches.

Table 1.2: *Element attributes (continued)*

Attribute	Meaning
Rotation	Amount to rotate the element. Legal values for this attribute are 0, 90, 180 or 270.
Top	Floating point value indicating the top of the rectangle the element will be rendered in. This value will be measured using the units specified in the Measurement attribute.
Width	Floating point value indicating the width of the rectangle the element will be rendered in. This value will be measured using the units specified in the Measurement attribute.

If the Item attribute is “Shape”, Element should contain a single Shape element. The Shape element has the only the Type outline. Type indicates the type of shape to render. Legal values are Rectangle, TopLine, BottomLine, LeftLine, RightLine, TLToBRLine, and BLToTRLine.

Shape requires a single Brush element. The Brush element contains two attributes. The first of these is Color, which stores the brush color. This is a hexadecimal number laid out in the same format as a TColor property. The second attribute is the Style attribute. This stores the brush style that will be used in drawing the shape. Legal values for this attribute are Solid, Clear, Horizontal, Vertical, FDiagonal, BDiagonal, Cross, and DiagCross. Shape contains no elements.

Shape requires a single Pen element. The pen element has the attributes shown in Table 1.3.

Table 1.3: *Pen attributes*

Attribute	Meaning
Color	The pen color. This is a hexadecimal number laid out in the same format as a TColor property.
Style	The style of the pen. Legal values are Solid, Dash, Dot, DashDot, DashDotDot, Clear and InsideFrame.
Width	Width of the pen. This is a number indicating the number of pixels wide the pen is.

If the Item attribute is “Caption”, a single Caption sub-element is required. The Caption element contains a single attribute, Caption. Caption is the caption that will be printed.

The Caption element requires a single Font sub-element. The attributes for the Font element are shown in Table 1.4.

Table 1.4: *Font attributes*

Attribute	**Meaning**
Bold	Indicates if the font should be bold. Legal values are True or False.
CharSet	The character set to use. Legal values for this attribute are ANSI, Default, Symbol, Mac, ShiftJIS, Hangeul, Johab, GB2312, ChineseBig5, Greek, Turkish, Vietnamese, Hebrew, Arabic, Baltic, Russian, Thai, EastEurope or OEM.
Color	The color of the text. This is a hexadecimal number laid out in the same format as a TColor property.
Height	The integer value representing the height of the font.
Italic	Indicates if the font should be italicized. Legal values are True or False.
Name	The name of the font.
Pitch	The font pitch. Legal values for this are Default, Variable or Fixed.
Strikeout	Indicates if the font should be struck out. Legal values are True or False.
Underline	Indicates if the font should be underlined. Legal values are True or False.

Printing Tutorial

Print formats are fairly complicated. This tutorial shows how to create print formats at design time. These formats can be saved and loaded by using the SaveToFile and LoadFromFile methods of the TVpPrinter class.

Creating a single element print format

The simplest print format to create is a DayView that fills the entire page.

First, start a new project and drop a TVpControlLink and TVpPrintPreview component on the form. Double-click the TVpPrintPreview's ControlLink property to set it to ControlLink1.

Click on the TVpControlLink component and open up the Printer property. Double-click on the PrintFormats property to open up the PrintFormats editor. This is a TCollection editor. Each item that is created in this list is a new print format. Click Insert to add a new print format.

Each print format requires a name. Set the FormatName to "DayView." Since this print format will consist of a single day, set the DayInc value to 1 and the DayIncUnits to duDay. Double-click the Elements property to open the element editor. Like the Print formats, this is a TCollection editor. Press Insert to add a new element.

By default, a DayView component is created that fills the entire page.

Run the program. You should see a somewhat shrunken day view in the print preview.

Creating a more complex print format

Exit the program and go back to the PrintFormats TCollection editor. Press Insert to add another print format.

Set the name of this new print format to “Two DayViews.” This new print format is going to consist of two side by side rotated DayView components. Since this will reflect two days, change the DayInc property to 2 and leave DayIncUnits at duDay. Double-click the Elements property to access the Elements property editor.

Inside the Elements editor, click Insert to add an element. Change the Height property to 50. By default, the measurements are in percentage of the page, so this will result in an element that is positioned on the top half of the page. Change the Rotation property to ra90.

Go back to the Elements editor and insert a second element. On this one, change the Height and Top properties to 50. This will put the element on the bottom half of the page. Again, change the Rotation property to ra90. To get the second day view to show the next day, change the DayOffset to one and leave the DayOffsetUnits as duDay.

Lastly, drop a TVpPrintFormatComboBox on the form and connect its ControlLink property to the TVpControlLink component. Run the program. The print format combo box will switch between two print formats.