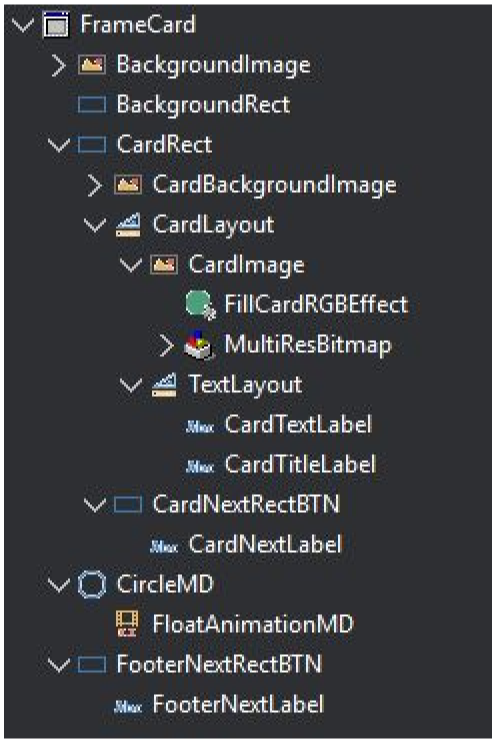
# Card View Wizard Template

## Overview

The Card View Wizard Template is a FireMonkey layout template that incorporates a number of card view pages that can be navigated forward and backward through as one would use when building an in-app tutorial. The Card View Wizard Template can be configured at both design time in the RAD Studio IDE and at runtime via code. It supports using the device hardware button to navigate background through the card view pages. It also supports making left and right swipe gestures to navigate forward and backward through the card view pages.



```
FrameCard
  BackgroundImage
  BackgroundRect
  CardRect
    CardBackgroundImage
    CardLayout
      CardImage
        FillCardRGBEffect
        MultiResBitmap
      TextLayout
        CardTextLabel
        CardTitleLabel
    CardNextRectBTN
      CardNextLabel
  CircleMD
    FloatAnimationMD
  FooterNextRectBTN
    FooterNextLabel
```

## Architecture

The Card View Wizard Template is based on the built in TTabControl component. Each card view page is located on a TTabItem. The number of card view pages that can be setup is dynamic. On each TTabItem a TFrameCard (TFrame) is placed and it contains all of the controls that make up the card view page. The Card View Wizard Template  is designed with three different default layouts but is not limited to just those layouts. The configuration of the layouts is applied via a ConfigureCard procedure. Other elements like the text and color of TLabels, buttons, and background images can also be configured via procedures on the TFrameCard control. The Stellar Style is applied to the TStyleBook that is on the main form in the template.

## Three Card View Wizard Variations

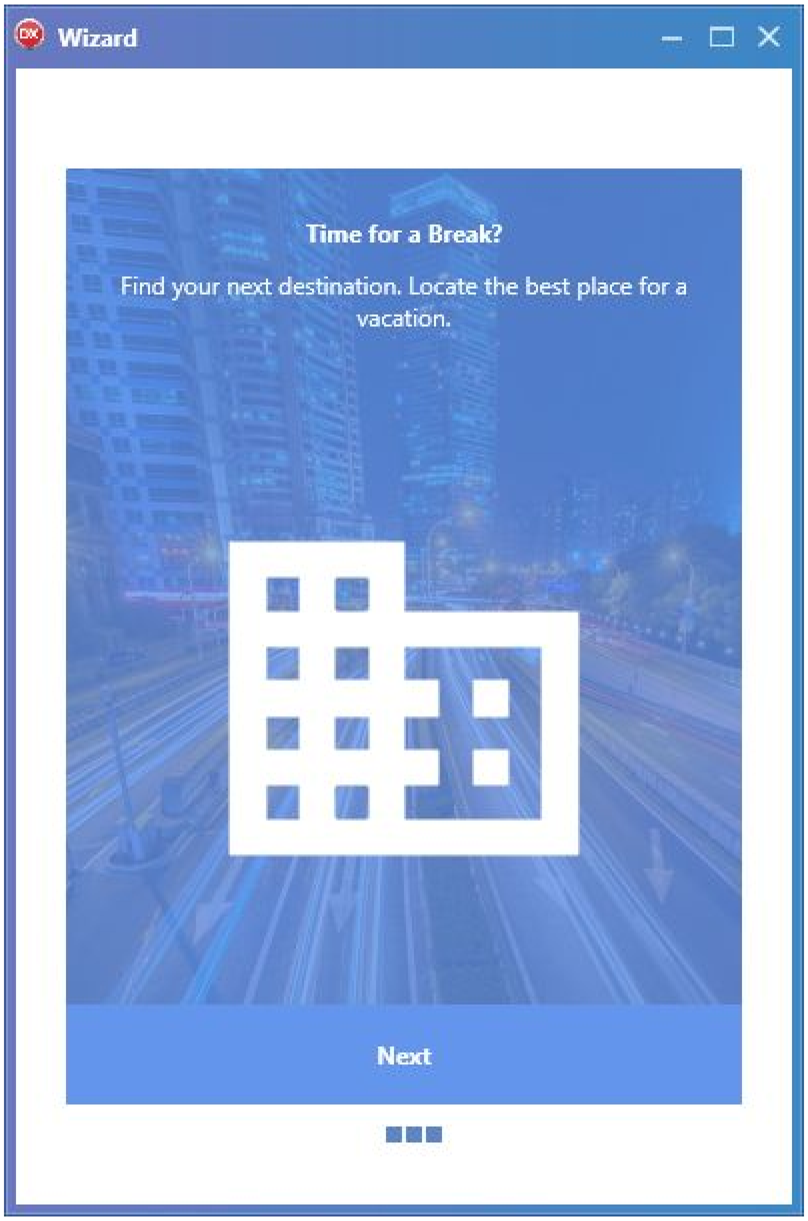
In the main form of the Card View Wizard Template there are three conditional defines. Each define allows to you set one default variation of the card view wizard template. You can comment and uncomment each different DEFINE to try out each of the three variations.

```
{$DEFINE FIRSTSET}

//{$DEFINE SECONDSET}

//{$DEFINE THIRDSET}
```

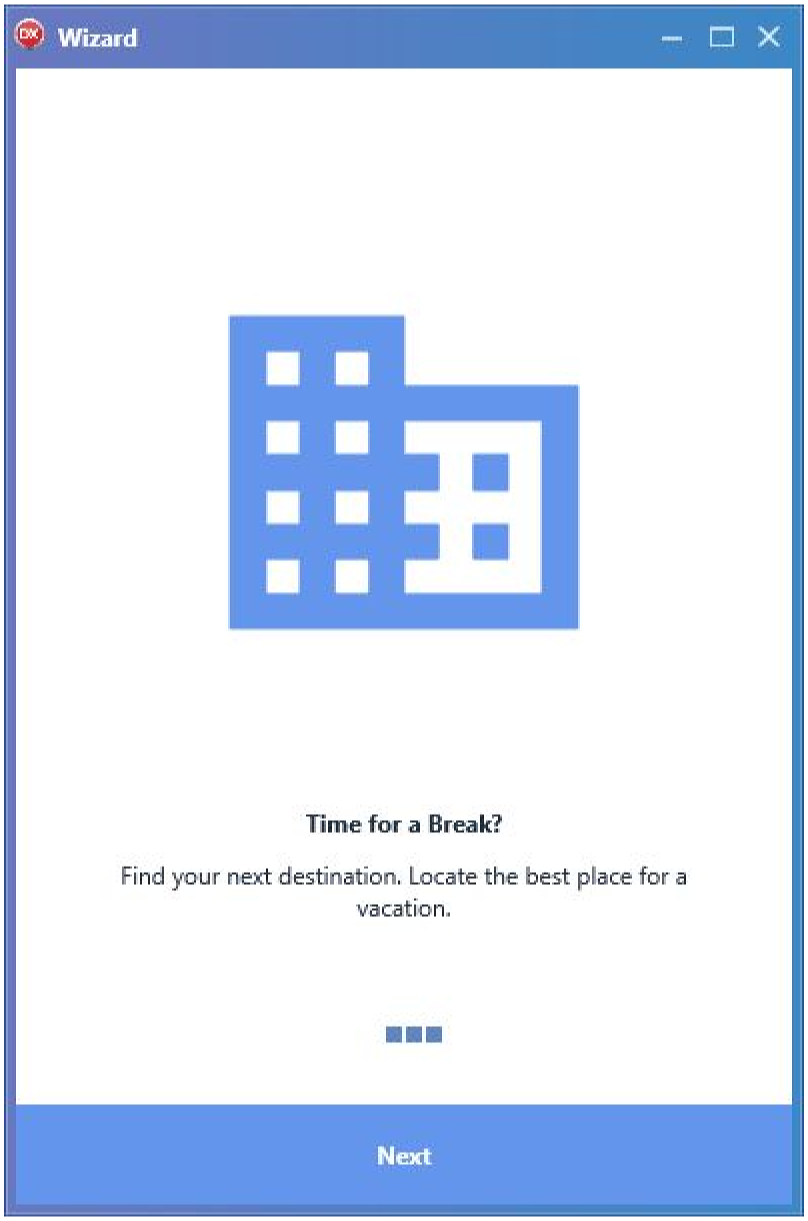
## First Set Variation



Time for a Break?

Find your next destination. Locate the best place for a vacation.

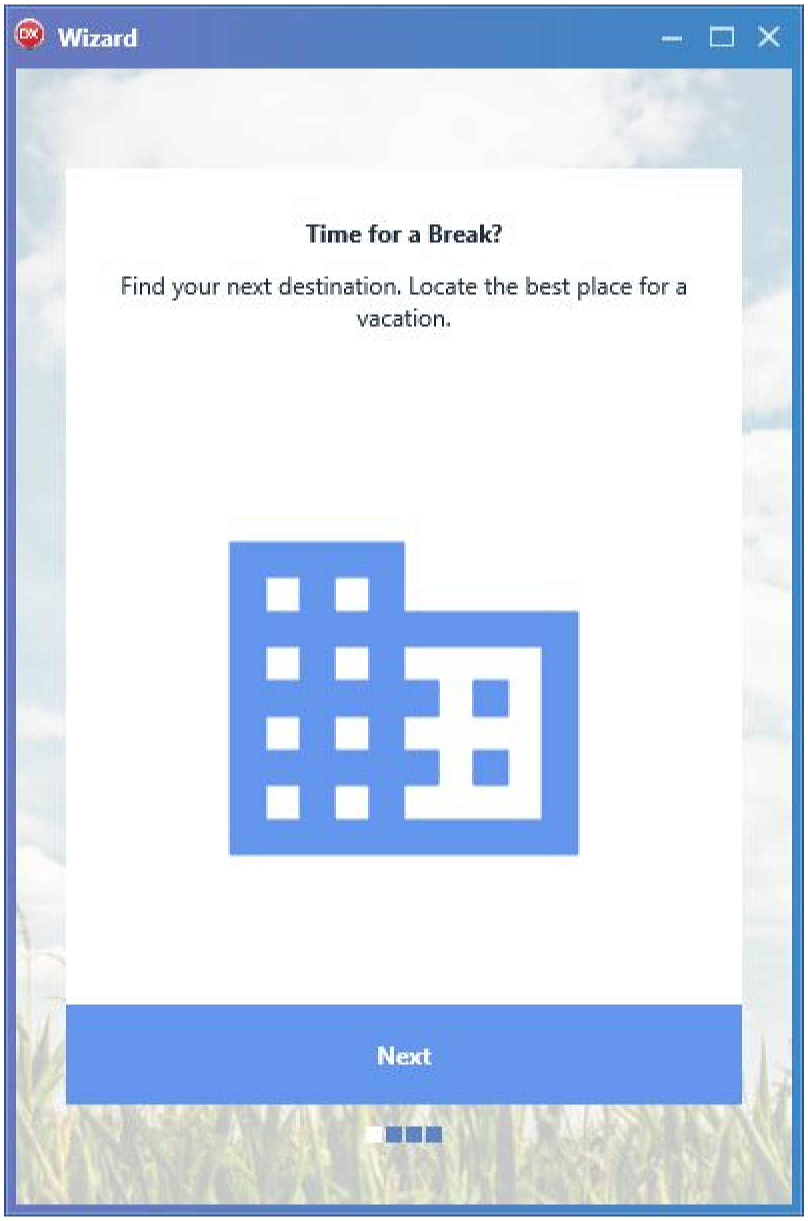
Second Set Variation



Time for a Break?

Find your next destination. Locate the best place for a vacation.

Third Set Variation


Wizard
Time for a Break?
Find your next destination. Locate the best place for a vacation.
Next

## Configure A Card View Page At Design Time

All of the elements of a card view page can be configured at design time. You can edit the TFrameCard itself directly before adding them to a TTabItem on your form. All of the elements are configurable just like any other control or TFrame would be in the RAD Studio IDE.

However, if you make changes to a TFrameCard at design time once you place it on a TTabItem those changes will have to be reapplied if you ever have to remove the TFrameCard from the form and re-add it. You can make changes to the TFrameCard in the Frame itself and those changes will apply to all new TFrameCards that you add to your form.

## Configure A Card View Page At Runtime

There are a number of procedures defined on the TFrameCard that you can use to quickly configure each card view page. The most basic procedures are SetCardTitle(), SetCardText(), SetNextButtonText, and ConfigureCard. They are all pretty self explanatory. ConfigurCard has three parameters which allow you to configure the location of the next button (the TCardButton record), the location of the background image (the TCardBGImage record), and the order of the main title and main image (the TCardLayout record). Here is some sample code for configuring the title, description, next button text, and choosing a layout.

```
  FrameCard1.SetCardTitle('Title');

  FrameCard1.SetCardText('Description');

  FrameCard1.SetNextButtonText('Next');

  FrameCard1.ConfigureCard(TCardButton.Inner, TCardBGImage.Inner,
TCardLayout.TextImage);
```

The automatic layout changes provided by ConfigureCard are defined as follows:

**TCardButton.None** disables the card button.

**TCardButton.Inner** places the card button inside the card.

**TCardButton.Outer** places the card button at the bottom of the screen outside of the card.

**TCardBGImage.Inner** sets up the card to use the background image inside of the card.

**TCardBGImage.Outer** sets up the card to have the background image behind the card instead of inside it.

**TCardBGImage.InnerAndOuter** displays both the inner background image and the outer background image behind the card.

**TCardLayout.TextImage** shows the text at the top of the card with the card image below it.

**TCardLayout.ImageText** shows the text at the bottom of the card with the card image above it.

## Add And Remove Card View Pages

The Card View Wizard Template has 4 card view pages configured by default. If you want to remove pages you can do so by selecting the TTabControl and deleting a TTabItem control. If you want to add card view pages you can right click on the TTabControl and select Add TTabItem.

Once you have a new blank TTabItem you will want to select that Tab and then search for Frames in the Tool Palette. Drag and drop the Frames element from the Tool Palette onto your blank TTabItem. Select the FrameCard frame from the dialog box to drop it on the TTabItem. Select the new TFrameCard control and change it's Align property to Client.

## Configure Outer Card View Page Background Image

Select the TFrameCard on your TTabItem (or within the TFrame itself to apply the change across all TFrameCards) and then select the BackgroundImage control which is a TImage. You can change or clear the MultiResBitmap property to update the background image. The four default TFrameCards on the main form in the template have a background image already and if you do not need it you should clear the BackgroundImage as it is quite large. This background image is only used when using TCardBGImage.Outer or TCardBGImage.InnerAndOuter.

## Configure Outer Card View Page Background Rect

The BackgroundRect control is a TRectangle control which has a Fill property that you can use to configure the color of the TRectangle. If you use the TCardBGImage.Inner property you may want to configure the background color of the TFrameCard through the BackgroundRect.Fill.Color property.

## Configure Inner Card Background Image

Select the CardRect control in the TFrameCard and then select the CardBackgroundImage control. You can configure the MultiResBitmap property to add an image to the card view page.

In the TCardBGImage.Inner and TCardBGImage.InnerAndOuter settings the CardBackgroundImage is modified by an accent color which is the same color as the Next button.

### Configure Inner Card Background Rect

Select the CardRect control which acts as the background of the card view in the TFrameCard. You can configure it's background color via the CardRect.Fill.Color property.

### Configure Main Card Image

The main centered card image is called CardImage and is located inside of the CardLayout within CardRect on TFrameCard. You can edit the MultiResBitmap property to change the image. The size of the CardImage means you should use the MultiResBitmap functionality to define different image sizes for different scales. There are 5 scales already set up within the CardImage MultiResBitmap property. The images sizes go from 256x256 through 1024x1024. When the TCardLayout.ImageText setting is used a TFillRGBEffect is applied to the CardImage to reverse it by giving it a color.

### Configure Main Card Title And Text

CardTitleLabel and CardTextLabel are contained within a TextLayout control within CardLayout within CardRect on TFrameCard. CardTitleLabel and CardTextLabel are TLabel controls and they each have a Text property which contains the text. They also have a TextSettings.FontColor property which allows you to set the color of the text. However, TLabel controls also utilize the current style to set their font properties like FontColor. It is possible to override the color of the FontColor property by disabling StyledSettings.FontColor.

## Animations

The Next buttons in TFrameCard have a mouse down event defined called MaterialDesignMouseDown which creates the expanding click circle effect. The TTabControl transitions use the built-in SetActiveTabWithTransitionAsync procedure to animate the transition between tabs.

## Runtime Procedures

Here is a full list of the procedures defined on TFrameCard which you can use to configure the card view pages at runtime.

```
procedure ConfigureCard(CardButton: Integer; CardBGImage: Integer; CardLayout: Integer);

function GetAccentColor: TAlphaColor;

procedure SetAcceptColor(const AColor: TAlphaColor);

procedure SetCardTitle(const ATitle: String);

procedure SetCardText(const AText: String);

procedure SetCardTextColor(const AColor: TAlphaColor);

procedure SetCardImage(ABitmap: TBitmap);

procedure SetCardBackgroundImage(ABitmap: TBitmap);

procedure SetCardBackgroundColor(const AColor: TAlphaColor);

procedure SetCardColor(const AColor: TAlphaColor);

procedure SetBackgroundImage(ABitmap: TBitmap);

procedure SetBackgroundColor(const AColor: TAlphaColor);

procedure SetNextButtonText(const AText: String);

procedure SetNextButtonTextColor(const AColor: TAlphaColor);

procedure SetNextButtonColor(const AColor: TAlphaColor);
```