**Online Ordering template instruction**  
  
In order to use Online Ordering template, you need to download it from the official Embarcadero website.

The purpose of creating this template is that you as a developer of the online ordering application already have the main application design with ready basic functionality. You just have to connect your backend to the application and configure the output of your content in the existing methods for outputting data to graphic controls using our styles. In addition, of course, there is an ability to implement independently the rest of the functionality, which is suitable exclusively for your restaurant.

The template of the application we presented consists of 10 functional forms and the main form, which displays most of the functional forms inside it.

When the application starts, the start form is launched. It displays the name of your restaurant and basic contact information. Also to the right of the contact information, there are buttons that allow user to make a call from one of the available applications list. In addition, there is an ability to send an email to the mailing address of your restaurant using one of the proposed applications. At the bottom of the form there is a Get Started button. By clicking this button, you will be redirected to the main form of the application.  
  
The main form of the application, at startup, immediately loads and displays the Dashboard form. All forms loaded from the side menu of the main form or through the duplicate buttons of the Dashboard form are displayed on the pages of the TTabControl component. This allows you to use the main menu of the program on all forms where necessary. It also allows you to manage effectively open forms and transitions between them from one form through a single mechanism. The main method that is used to create forms, display their content on the TTabControl pages of the main form, as well as assign functionality to the controls of these forms, is performed in the LoadScreenByName () method. Having studied it and the methods called from it, you will figure out how the logic for displaying forms in the application is built.

We list the remaining forms that are implemented in this template:

* Menu form.
* Cart form.
* User account adding / editing form.
* The form of detailed information about the selected dish from the Menu form.
* The form for adding / removing modifiers of the selected dish to/from the Menu.
* Discount coupons form.
* Gallery form of your restaurant.
* About Us form.

Now, we will analyze in more detail how the work with the application data is organized. In addition, we will consider methods for outputting data to Listbox and other graphic elements of application forms.

Application test data, for an example, is implemented using the TFDMemTable component. In this case, the list of fields is not limited, and you do not need to use exactly this component in your project. Any TQuery or TDataSet should be fine. This version of the template assumes the ability to run the application on any platform.

In the application’s uDMUnit data module, a number of methods have been created that are necessary only to populate the TFDMemTable components with test data. You will not need these methods and you will delete them. Nevertheless, during studying our template, they will help you visually understand what data and what type of data you will need to receive from your backend or you can choose from your database in order to use it in the template. So that the template will work correctly with the data that you submitted.

Styles of all the components that are presented in StyleBook on the main form of the frmMain template can be changed as you wish. Therefore, if necessary, or if you need to display additional data in a particular component, you can easily remake existing styles and prepare the application for your needs.

In order for your data, rather than the test data, to be displayed on the TListBox visual elements, you need to upload your data to the TFDMemTable located on a data module or make changes to the method of filling it when the data module is initialized (implementing your methods of receiving data from the backend or from the database). The following are all the methods that populate TFDMemTables with the test data at application startup:

* InsertTestDataForTheMenu ();
* InsertTestDataForTheCoupons ();
* InsertTestDataForTheGallery ();
* InsertTestDataForTheOptions ();
* InsertTestDataForAboutUs ();
* InsertTestDataForContactInfo ();

From the name of the methods, it is clear which one is responsible for filling out one form or another with the test data. Having studied any of the methods, it becomes clear which of TFDMemTable1 ..TFDMemTable6 is used for which form.

In order to add images to TFDMemTables, use TImageLists with images already loaded in them. This is done in order to demonstrate how to load an image into the components of the TQuery family, and to show how to load images from fields of type TBlob into the elements of graphic controls on the form. Such, for example, as TListBox.

Loading data from TFDMemTables into graphic control elements is implemented on each form separately. A list of method names is given below:

* TfrmAboutUs.BuildAboutUsInfo ();
* TfrmAboutUs.BuildContsctInfoList ();
* TfrmAddToCart.BuildForm ();
* TfrmAddToCart.BuildOptionsList ();
* TfrmCart.BuildCartList ();
* TfrmCoupons.LoadCouponsList ();
* TfrmGallery.LoadGalleryList ();
* TfrmGetStarted.ShowInfoAboutRestaurant ();
* TfrmGetStarted.BuildContsctInfoList ();
* TfrmMenu.LoadMenuList ();
* TfrmOptionsList.BuildOptionsList ();

It should be admitted where, by what methods and in what way the work with the customer’s shopping cart is organized. The cart is implemented using an array of record, presented as a TCartList class. Dishes and their modifiers are added to this array with one or another type of TCartItemType. And the connection between the dish and the modifiers is organized through the parameters Index, OwnerIndex of the TCartList class.

For the convenience of working with the cart, several methods are implemented:

AddItemToCart(aItemId: int64 = 0; aOwnerID: int64 = 0; const aItemName: string = ''; aItemType: TCartItemType = citItem; aQuantity: Extended = 1; aItemPrice: Extended = 0; aOwnerIndex: Integer = -1): Integer;

DelItemFromCart(aItemIndex: Integer = 0);

ClearCart();

GetCartTotalAmount(): Extended;

In any case, you can change these methods as you like, or write your own methods for filling out forms and their visual components with the data. The methods given in the template only show how filling the data of the template forms visual elements is implemented, as well as the editing of this data. The methods above provide ways for assigning click processing methods to TlistBoxItems or buttons located inside TlistBoxItems. After analyzing these methods, you will notice that the Tag property is used to transfer the identifiers of the record or item.

Each of the templates in the current implementation represents only stylistic, graphic and animation effects. And the test data is presented only for understanding the processes of filling graphic elements of forms.

Icons of graphic elements used in the templates are taken from the resource:

<https://material.io/tools/icons/?style=baseline>

Below is a list of Internet resources from which images were downloaded to design the appearance of the application.

GetStarted form BG image: <https://wallpapershome.ru/eda/glavnie-blyuda/makaroni-pomidori-olivki-16053.html>

<https://www.almerostudentmansions.com/luxury-takeaway-exeter/>

<https://www.budgetbytes.com/garlic-bread/>

<https://www.epicurious.com/recipes/food/views/spicy-and-greasy-rhode-island-calamari-51237100>

<https://www.alfrescoallnatural.com/recipe/sundried-tomato-chicken-sausage-risotto>

<https://www.yelp.com/biz/presto-pasta-santa-clarita-2>

<https://www.pinterest.com/pin/462393086733346619/>

<https://ru.depositphotos.com/stock-photos/%D0%B4%D0%B5%D1%81%D0%B5%D1%80%D1%82.html?qview=12002700>

<https://ru.depositphotos.com/stock-photos/%D0%B4%D0%B5%D1%81%D0%B5%D1%80%D1%82.html?qview=24212433>

<https://wallpapers.99px.ru/wallpapers/199521/>

Social icons:

<https://icon-icons.com/ru/download/59205/PNG/128/>

<https://icon-icons.com/ru/download/83078/PNG/128/>

<https://icon-icons.com/ru/download/83102/PNG/128/>

<https://icon-icons.com/ru/download/83061/PNG/128/>