JADE Skins

Version 2018

Contents

JADE Skins

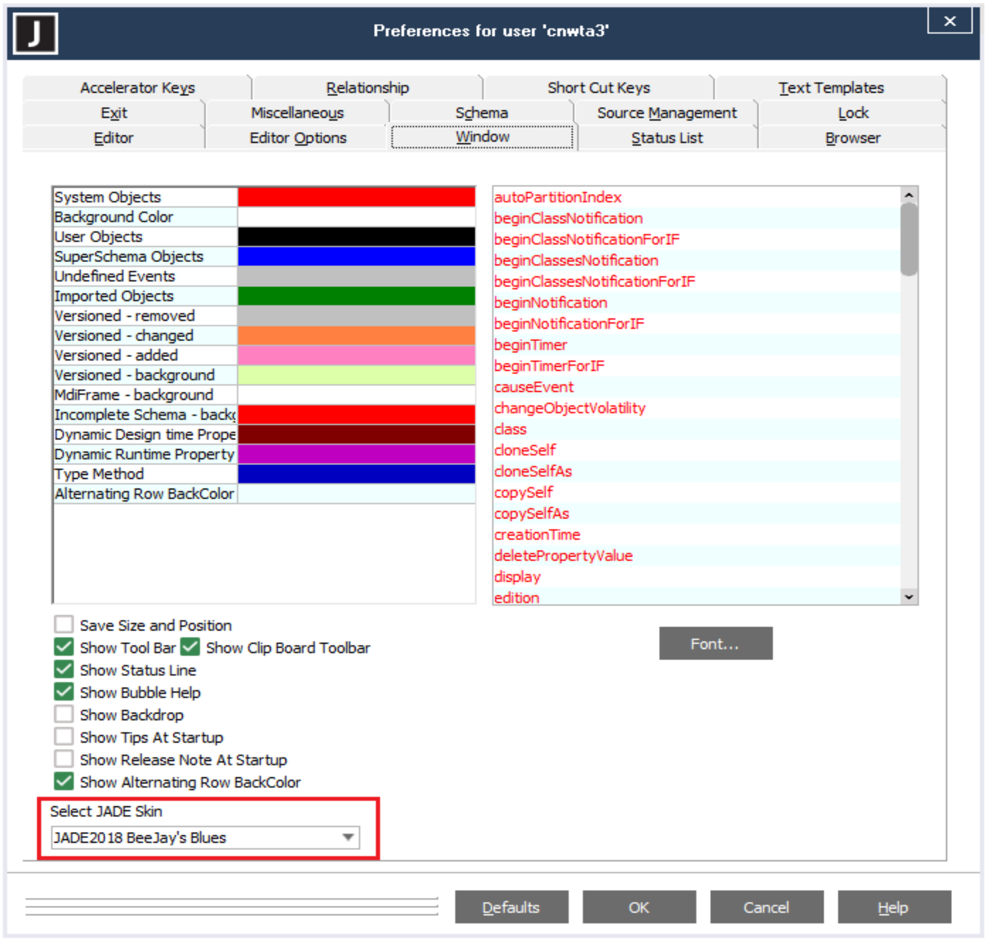
Introduction

JADE skins enable you to customize JADE form Graphical User Interface (GUI) elements. By configuring and saving colors, fonts, shapes, and sizes of the various controls on JADE forms, you can enforce a consistent look-and-feel across multiple applications.

Selecting a Skin for the Development Environment

As the JADE development environment is implemented in JADE, you can use JADE skins to customize the look and feel of the development environment in the same way you can for any other JADE application.

To select a different skin for your JADE development environment, select one of the provided skins in the **Select JADE Skin** drop-down list box on the **Window** sheet of the Preferences dialog (accessed by selecting the **Preferences** command from the Options menu).



Preferences for user 'cnwta3'

Accelerator Keys | Relationship | Short Cut Keys | Text Templates
Exit | Miscellaneous | Schema | Source Management | Lock
Editor | Editor Options | Window | Status List | Browser

System Objects	🟥
Background Color	⬜
User Objects	⬛
SuperSchema Objects	🟦
Undefined Events	⬜
Imported Objects	🟩
Versioned - removed	⬜
Versioned - changed	🟧
Versioned - added	🟪
Versioned - background	🟩
MdiFrame - background	⬜
Incomplete Schema - backg	🟥
Dynamic Design time Prope	🟥
Dynamic Runtime Property	🟪
Type Method	🟦
Alternating Row BackColor	⬜

autoPartitionIndex
beginClassNotification
beginClassNotificationForIF
beginClassesNotification
beginClassesNotificationForIF
beginNotification
beginNotificationForIF
beginTimer
beginTimerForIF
causeEvent
changeObjectVolatility
class
cloneSelf
cloneSelfAs
copySelf
copySelfAs
creationTime
deletePropertyValue
display
edition

Font...

☐ Save Size and Position
☑ Show Tool Bar ☑ Show Clip Board Toolbar
☑ Show Status Line
☑ Show Bubble Help
☐ Show Backdrop
☐ Show Tips At Startup
☐ Show Release Note At Startup
☑ Show Alternating Row BackColor

Select JADE Skin
JADE2018 BeeJay's Blues

Defaults | OK | Cancel | Help

Note **JADE2018 Cashmere** is the default skin for versioned schemas.

If you use this skin for your standard JADE development environment, you should change the default skin for versioned schemas (on the **Miscellaneous** sheet of the Preference dialog).

JadeSkinMaintenance Form

The **JadeSkinMaintenance** form (the Jade Skin Maintenance dialog) is used to maintain existing skins and define new skins.

This form enables you to re-skin any number of controls, forms, and menus. These re-skins are then combined into an application skin, which can be applied to any number of applications.

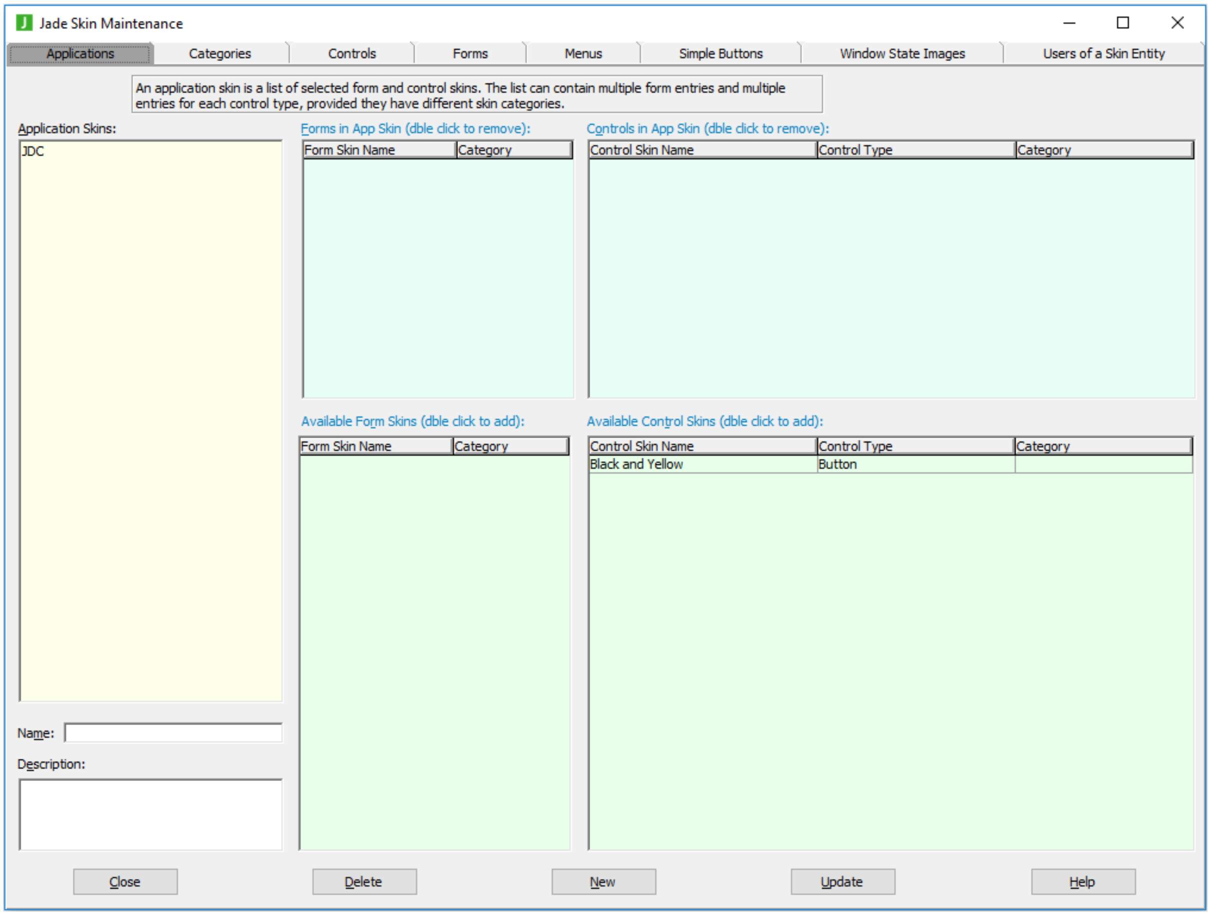
To open the Jade Skin Maintenance dialog, define and run the following JadeScript **createJadeSkinMaintenance** method.

```
createJadeSkinMaintenance();

vars
    form : JadeSkinMaintenance;
begin
    create form transient;
    form.showModal();
epilog
    delete form;
end;
```

The first sheet of the Jade Skin Maintenance dialog is used to combine individual control re-skins into an application skin.

An application skin requires at least one **Form** or **Control** skin to be applied to it before it can be created.

As you create **Form** and **Control** skins, they are added to the lists of **Available Form Skins** and **Available Control Skins**.



An application skin is a list of selected form and control skins. The list can contain multiple form entries and multiple entries for each control type, provided they have different skin categories.

Application Skins:

JDC

Forms in App Skin (dble click to remove):

Form Skin Name	Category

Controls in App Skin (dble click to remove):

Control Skin Name	Control Type	Category

Available Form Skins (dble click to add):

Form Skin Name	Category

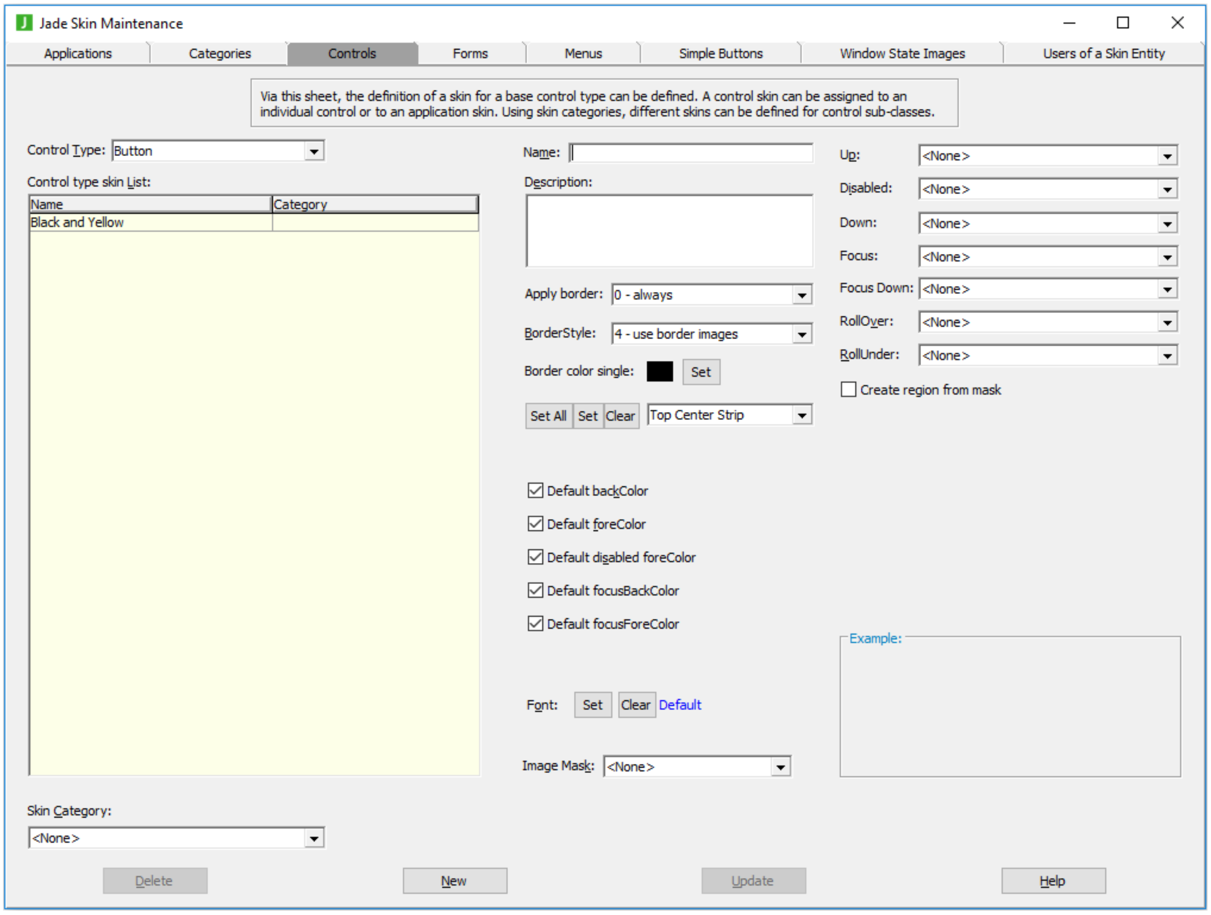
Available Control Skins (dble click to add):

Control Skin Name	Control Type	Category
Black and Yellow	Button	

Name:

Description:

To create a **Control** skin, select the **Controls** sheet of the Jade Skin Maintenance dialog.



Via this sheet, the definition of a skin for a base control type can be defined. A control skin can be assigned to an individual control or to an application skin. Using skin categories, different skins can be defined for control sub-classes.

Control Type: Button

Control type skin List:

Name	Category
Black and Yellow	

Skin Category: <None>

Name:

Description:

Apply border: 0 - always

BorderStyle: 4 - use border images

Border color single: Set

Set All Set Clear Top Center Strip

☑ Default backColor

☑ Default foreColor

☑ Default disabled foreColor

☑ Default focusBackColor

☑ Default focusForeColor

Font: Set Clear Default

Image Mask: <None>

Up: <None>

Disabled: <None>

Down: <None>

Focus: <None>

Focus Down: <None>

RollOver: <None>

RollUnder: <None>

☐ Create region from mask

Example:

Delete New Update Help

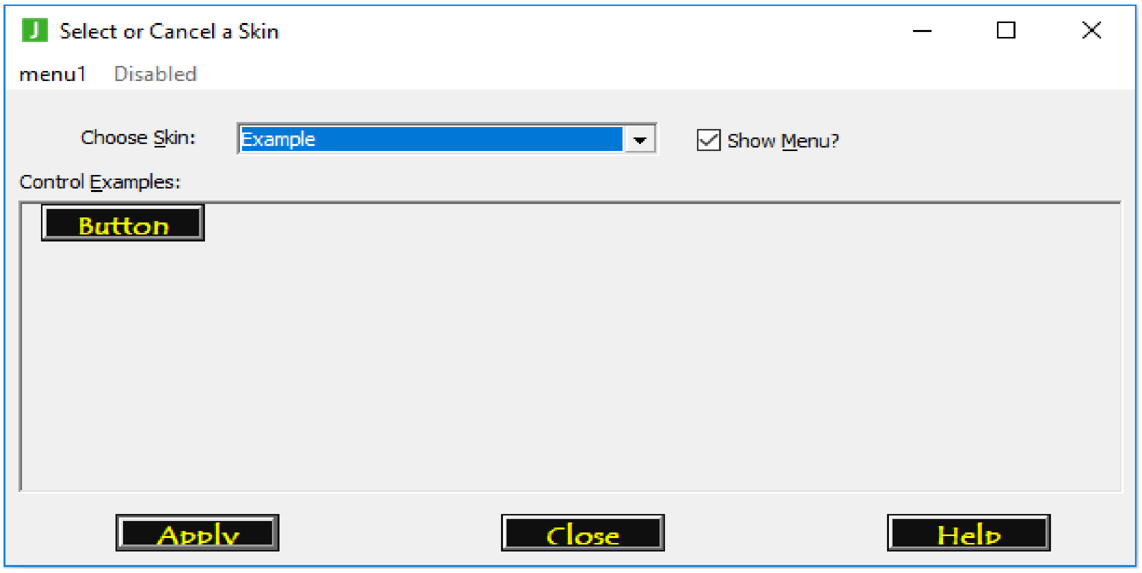
From this form, you can select a variety of options to be applied to all instances of a specific control. For example, you can select the color and font of the text of the button's label, the background color of the button, and the style of the border for the **Button** control type button. Alternatively, you can set an image for each of the parts of the **Button** control.

JadeSkinSelection Form

The **JadeSkinSelection** form (the Skin Selection dialog) is used to select the skin to be applied to the current application. The form can be created directly from the end-user application; for example, from a menu or the **click** event of a button. It can also be called from a JadeScript method, to preview what a skin will look like. The form can be created with the following code.

```
createJadeSkinSelection();

vars
    form : JadeSkinSelection;
begin
    create form transient;
    form.showModal();
epilog
    delete form;
end;
```

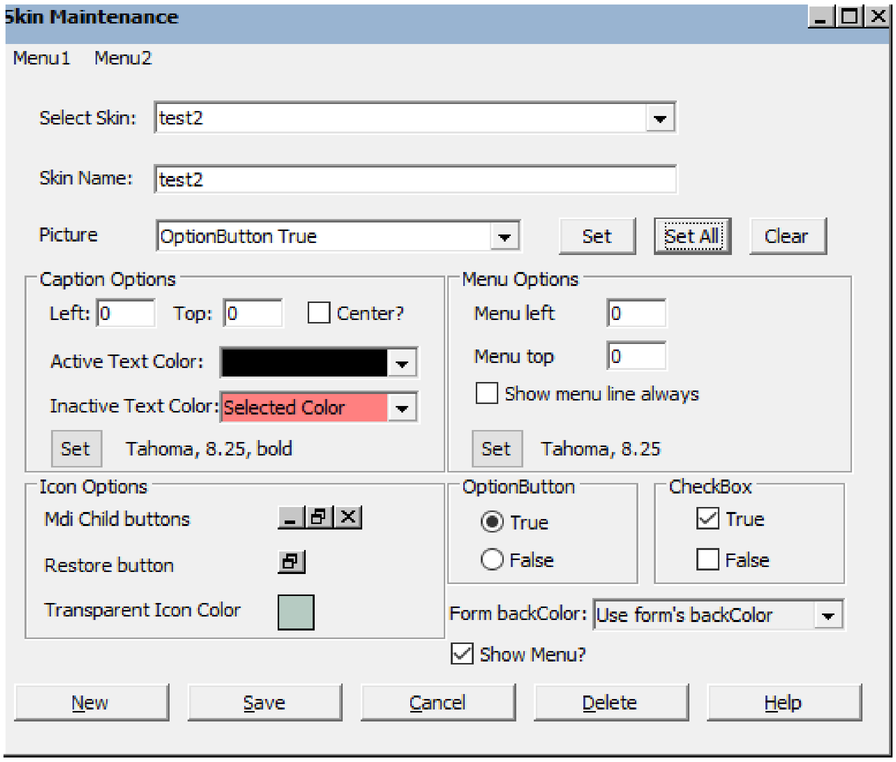
When viewing the Skin Selection dialog, the dialog itself is displayed using the selected skin. This allows the user to preview what the skin will look like before selecting a skin for use at run time.



In this simple example, the **Example** application skin has customized buttons with yellow text on a black background. The Skin Selection dialog displays its buttons in this style while **Example** is selected as the skin. Click **Apply**, to apply the **Example** skin to all application forms for the remainder of the application's lifetime.

JadeSkinMaint Form

The **JadeSkinMaint** form (the Skin Maintenance dialog) was the precursor to the **JadeSkinMaintenance** form and it is retained in JADE for backwards compatibility.



Select Skin: test2

Skin Name: test2

Picture: OptionButton True

Caption Options: Left: 0 Top: 0 Center? ☐

Active Text Color: ■

Inactive Text Color: Selected Color

Tahoma, 8.25, bold

Menu left: 0

Menu top: 0

Tahoma, 8.25

OptionButton: [✓ True] False

CheckBox: ☑ True ☐ False

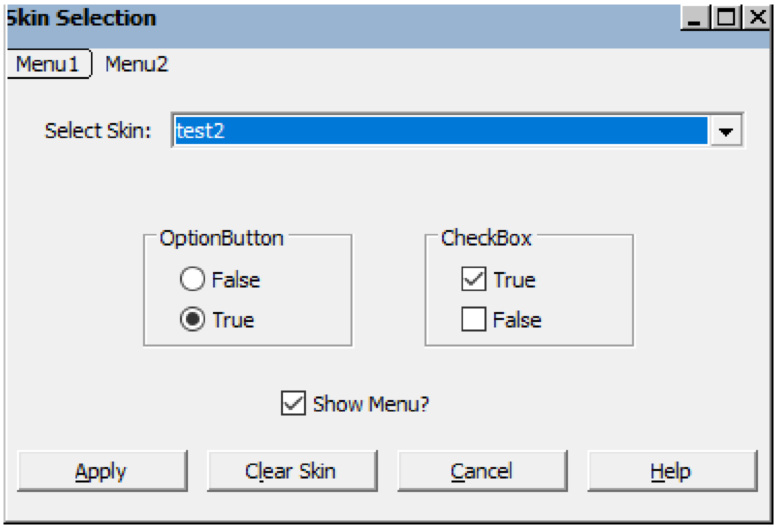
Form backColor: Use form's backColor

☑ Show Menu?

Note We recommend that you use the Jade Skin Maintenance dialog rather than the Skin Maintenance dialog.

JadeSkinSelect Form

The **JadeSkinSelect** form was the precursor to the **JadeSkinSelection** form and it is retained in JADE for backwards compatibility.



Select Skin: test2

OptionButton: ○ False ● True

CheckBox: ✓ True ☐ False

✓ Show Menu?

Note We recommend that you use the **JadeSkinSelection** form rather that the **JadeSkinSelect** form.

Exercise 1 – Creating a Button Control Skin

In this exercise, you will create a simple application skin consisting of a few control skins.

1. Add the following JadeScript **createJadeSkinMaintenance** method to **BankingViewSchema**.

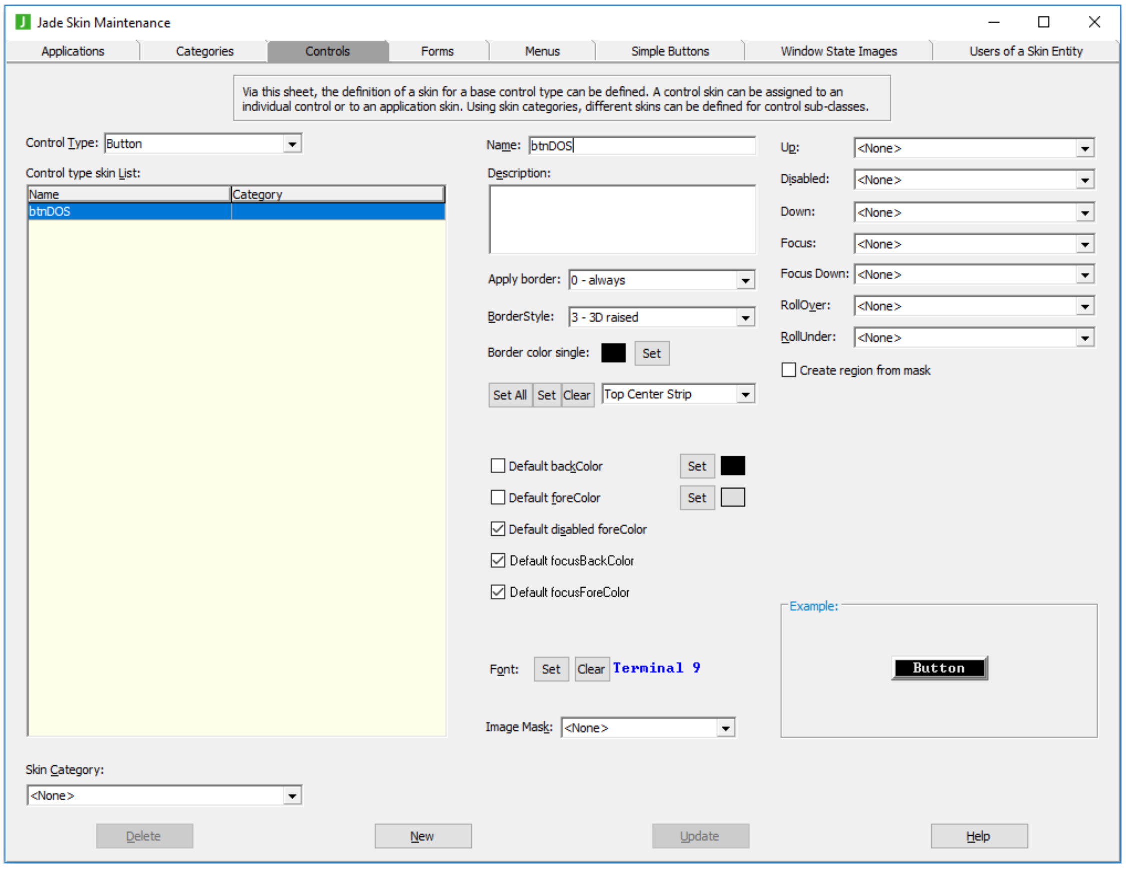
```
createJadeSkinMaintenance();

vars
    form : JadeSkinMaintenance;
begin
    create form transient;
    form.showModal();
epilog
    delete form;
end;
```

2. Run the method, to open the Jade Skin Maintenance dialog.
3. Select the **Controls** sheet and then select **Button** in the **Control Type** combo box.

Tip As the options in the following steps are examples only, feel free to customize the elements to your requirements.

4. Set **BorderStyle** to **3 - 3D raised**.
5. Uncheck the **Default backColor** check box and select **Black** as the color.

6. Uncheck the **Default foreColor** check box and select **Light Gray** as the color.
7. Set the font to **Terminal** and **Font Size** to **9**.
8. Call the skin **btnDOS**. The form should then look like the following.


Jade Skin Maintenance

Applications | Categories | Controls | Forms | Menus | Simple Buttons | Window State Images | Users of a Skin Entity

Via this sheet, the definition of a skin for a base control type can be defined. A control skin can be assigned to an individual control or to an application skin. Using skin categories, different skins can be defined for control sub-classes.

Control Type: Button

Control type skin List:

Name	Category
btnDOS	

Name: btnDOS

Description:

Apply border: 0 - always

BorderStyle: 3 - 3D raised

Border color single: Set

Set All Set Clear Top Center Strip

☐ Default backColor Set

☐ Default foreColor Set

☑ Default disabled foreColor

☑ Default focusBackColor

☑ Default focusForeColor

Font: Set Clear Terminal 9

Image Mask: <None>

Up: <None>

Disabled: <None>

Down: <None>

Focus: <None>

Focus Down: <None>

RollOver: <None>

RollUnder: <None>

☐ Create region from mask

Example: Button

Skin Category: <None>

Delete New Update Help

9. Click **Update**, to save the **Button** skin.
10. Select the **Applications** sheet and double-click the **btnDOS** skin to add it to the current skin.
11. Enter **DOS** as the Application skin name and then click **Update**, to save it.

Exercise 2 – Previewing a Skin

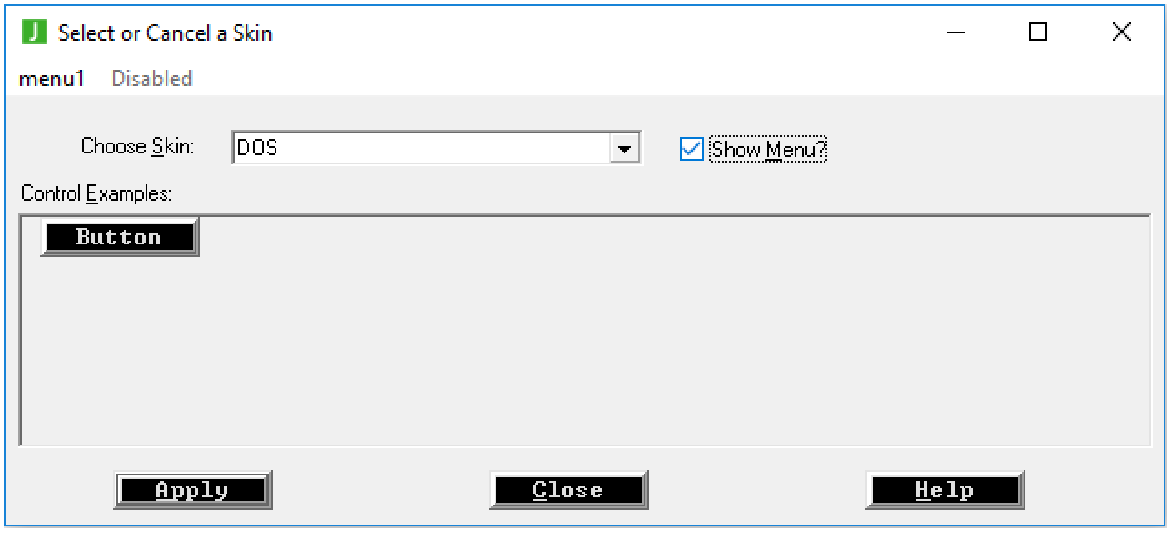
In this exercise, you will use the **JadeSkinSelection** form to preview the changes to buttons made to your skin.

1. Add the following JadeScript **createJadeSkinSelection** method to **BankingViewSchema**.

```
createJadeSkinSelection();

vars
    form : JadeSkinSelection;
begin
    create form transient;
    form.showModal();
epilog
    delete form;
end;
```

2. Run the method defined in the precious step and select **DOS** from the **Choose Skin** combo box.

You should see the buttons displayed using the skin you have designed in the first exercise in this module.

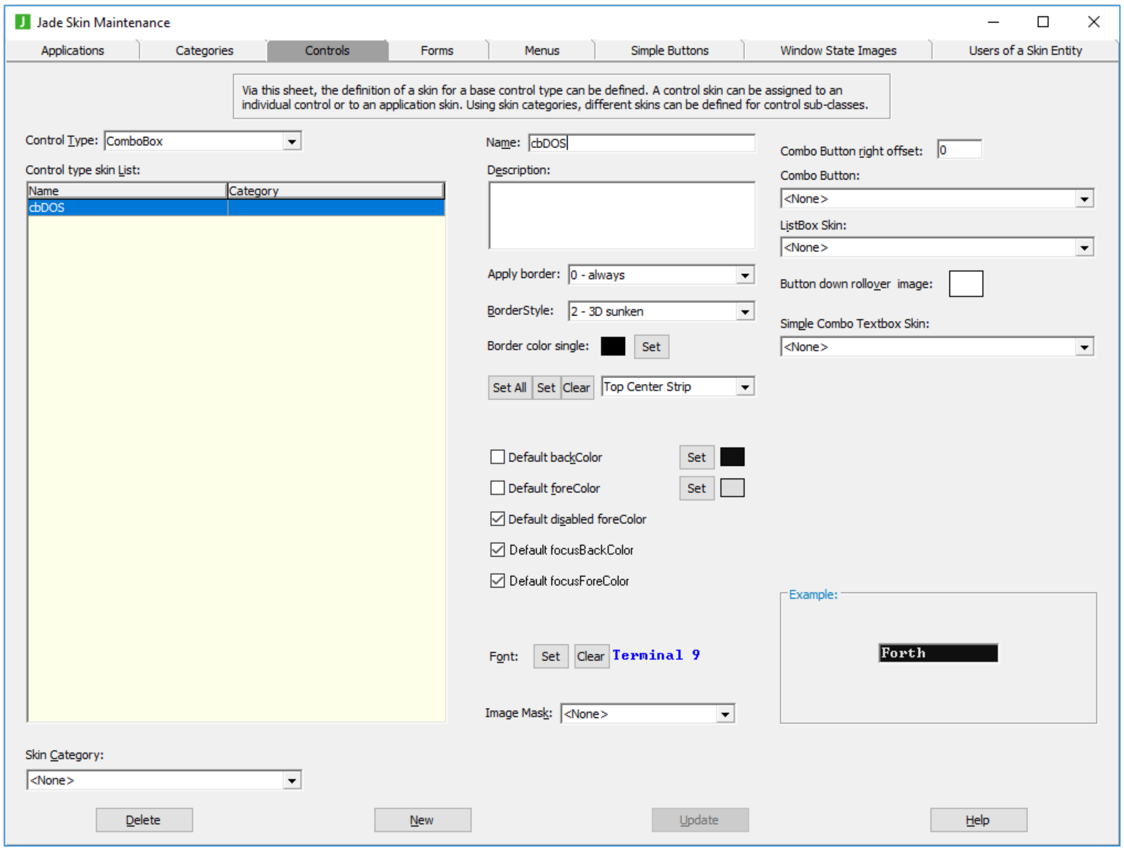


Exercise 3 – Creating a ComboBox Control Skin

In this exercise, you will add a control skin to combo boxes in the **DOS** application skin.

1. Open the Jade Skin Maintenance dialog by running the JadeScript **createJadeSkinMaintenance** method.
2. Select the **Controls** sheet and then set the control type to **ComboBox**.
3. Set the **BorderStyle** to **2 - 3D sunken**.
4. Uncheck the **Default backColor** check box and select **Black** as the color.
5. Uncheck the **Default foreColor** check box and select **Light Gray** as the color.
6. Set the font to **Terminal** and the **Font Size** to **9**.
7. Call the skin **cbDOS**.

The form should then look like the following.



Via this sheet, the definition of a skin for a base control type can be defined. A control skin can be assigned to an individual control or to an application skin. Using skin categories, different skins can be defined for control sub-classes.

Control Type: ComboBox

Control type skin List:

Name	Category
cbDOS	

Skin Category: <None>

Name: cbDOS

Description:

Apply border: 0 - always

BorderStyle: 2 - 3D sunken

Border color single: ■ Set

Set All Set Clear Top Center Strip

☐ Default backColor Set ■

☐ Default foreColor Set ☐

☑ Default disabled foreColor

☑ Default focusBackColor

☑ Default focusForeColor

Font: Set Clear Terminal 9

Image Mask: <None>

Combo Button right offset: 0

Combo Button: <None>

ListBox Skin: <None>

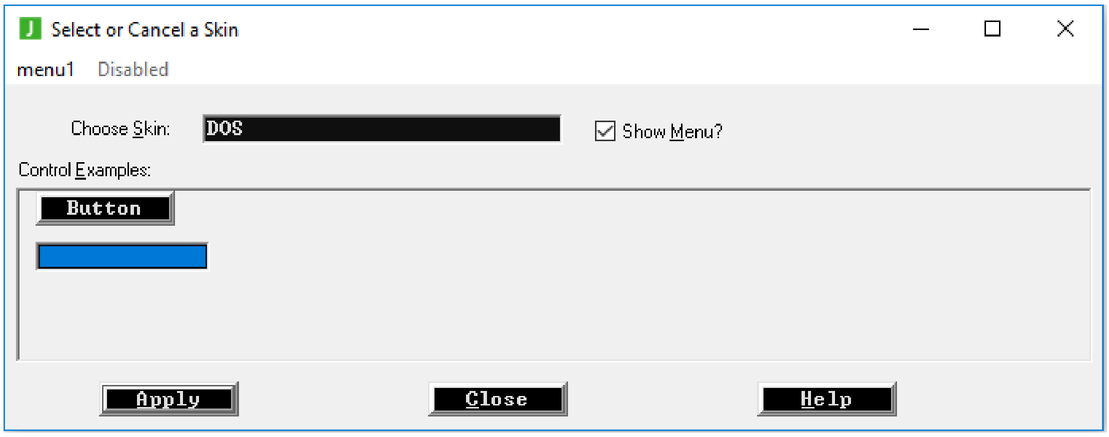
Button down rollover image:

Simple Combo Textbox Skin: <None>

Example: Forth

8. Click **Update**, to save the **ComboBox** skin changes.
9. Select the **Applications** sheet, add **cbDOS** to the **DOS** application skin, and then click **Update**.
10. Preview your changes in the **JadeSkinSelection** form as you did in the previous exercise.

The form should now look like the following.

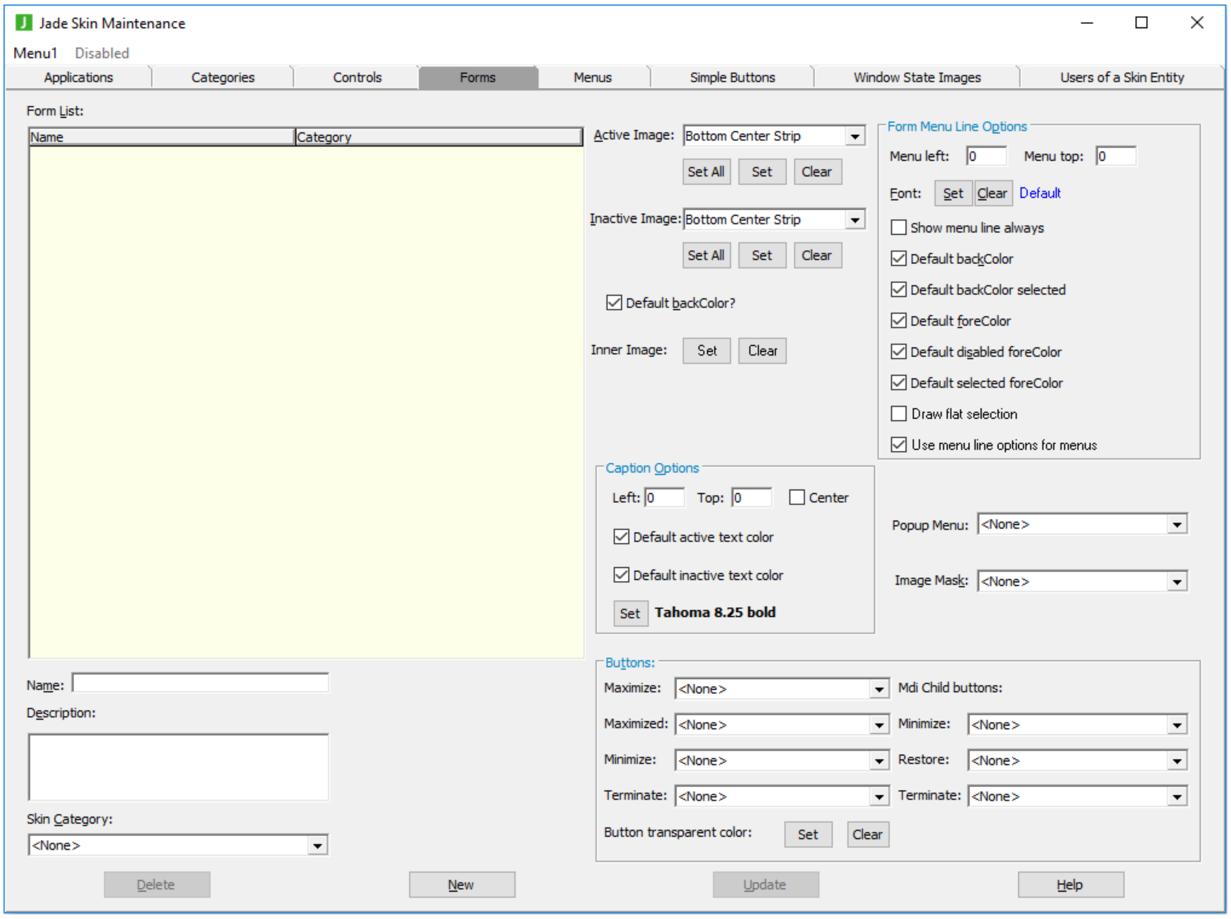


Choose Skin: DOS

☑ Show Menu?

Customizing Forms with JADE Skins

The **Forms** sheet of the Jade Skin Maintenance dialog is used to specify a forms skin, which applies to the forms of an application themselves, rather than the controls on that form.



Active Image: Bottom Center Strip

Inactive Image: Bottom Center Strip

☑ Default backColor?

Menu left: 0 Menu top: 0

Font: Set Clear Default

☐ Show menu line always
☑ Default backColor
☑ Default backColor selected
☑ Default foreColor
☑ Default disabled foreColor
☑ Default selected foreColor
☐ Draw flat selection
☑ Use menu line options for menus

Left: 0 Top: 0 ☐ Center
☑ Default active text color
☑ Default inactive text color
Set Tahoma 8.25 bold

Popup Menu: <None>
Image Mask: <None>

Maximize: <None>
Maximized: <None>
Minimize: <None>
Terminate: <None>
Minimize: <None>
Restore: <None>
Terminate: <None>

Skin Category: <None>

This **Forms** sheet has the group boxes listed in the following table.

Group Box	Element to be Modified
Form Menu Line Options	The menu line at the top of forms. The drop-down menus also use this skin unless a menu skin is defined on the **Menu** sheet of the Jade Skin Maintenance dialog and applied to forms in the **Popup Menu** combo box.
Caption Options	The title caption of the forms.
Buttons	The minimize, maximize, and close buttons of the forms.

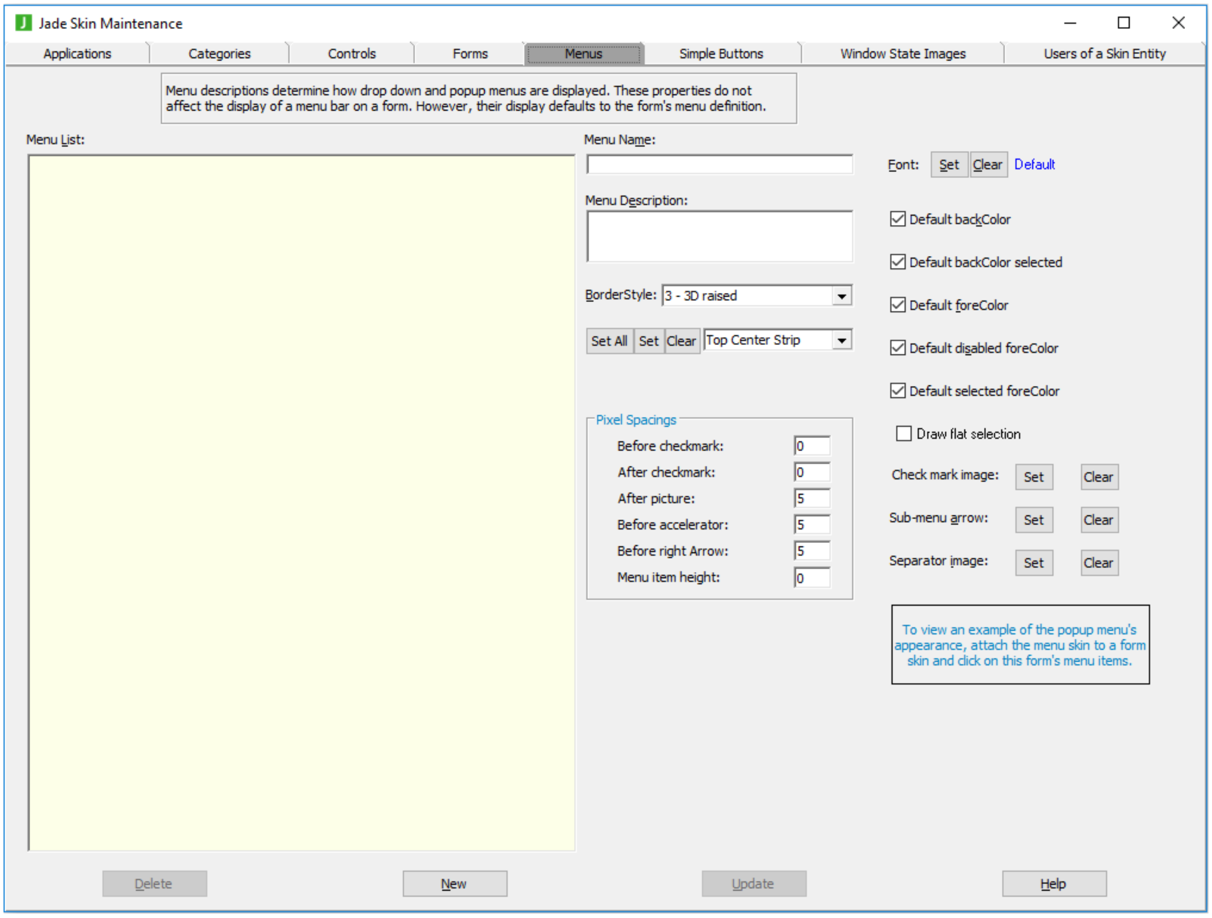
The **Forms** sheet also has the following options that are not in group boxes.

Option	Element to be Modified
Active Image	The border elements of the forms. You can set an image for each one, to fully customize the look of forms.

Option	Element to be Modified
Inactive Image	The border elements of the forms when they do not have focus. You can set an image for each one, to fully customize the look of forms.
Default backColor?	When unchecked, allows you to set a new default background color for the forms.
Inner Image	Displays the selected image as the background for forms.
Is brush	Displayed on the **Forms** sheet only when inner image is set. Checking this check box causes the inner image to be tiled on forms.
Popup Menu	The drop-down and popup menus. To use this option, you must first create a menu skin on the **Menus** sheet of the Jade Skin Maintenance dialog.
Image Mask	Allows for a non-rectangular region mask image to be applied to the forms' skin.

Customizing a Menu with JADE Skins

The **Menus** sheet of the Jade Skin Maintenance dialog is used to apply a skin to any drop-down and popup menus on forms. Menu skins do not apply to the menu line of forms, but the menu line's skin is used as the default menu skin until it is set.



Menu descriptions determine how drop down and popup menus are displayed. These properties do not affect the display of a menu bar on a form. However, their display defaults to the form's menu definition.

Menu Name:

Menu Description:

BorderStyle: 3 - 3D raised

Top Center Strip

Pixel Spacings

Before checkmark:	0
After checkmark:	0
After picture:	5
Before accelerator:	5
Before right Arrow:	5
Menu item height:	0

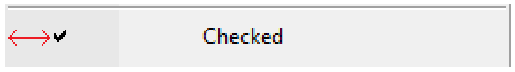
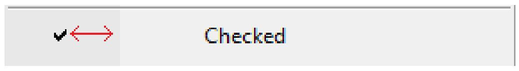
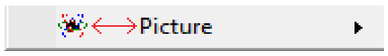
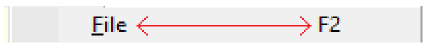
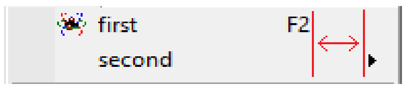
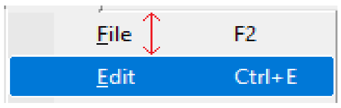
- [✓] Default backColor
- [✓] Default backColor selected
- [✓] Default foreColor
- [✓] Default disabled foreColor
- [✓] Default selected foreColor
- [ ] Draw flat selection

To view an example of the popup menu's appearance, attach the menu skin to a form skin and click on this form's menu items.

The **Menus** sheet has the following options.

Option	Description
BorderStyle	Controls the borders between menu items. You can choose **none**, **single**, **3D sunken**, **3D raised**, or **use border images**.
Set All / Set / Clear	Allows for the setting of image files as the borders between menu items when **BorderStyle** is set to **4 - use border images**.
Font	Sets the font (typeface) of the text in menu items.
Default backColor	Sets the default background color for menu items.
Default backColor selected	Sets the default background color for how menu items are displayed when they are selected.
Default foreColor	Sets the default text color for menu items.
Default disabled foreColor	Sets the default text color for disabled menu items.
Default selected foreColor	Sets the default text color for how menu items are displayed when they are selected.
Draw flat selection	When checked, selected menu items are drawn flat regardless of the border style that is selected. In addition, if the border style is set to **none** and multiple menu items are selected, no line is drawn between them.
Check mark image	Sets the image to be used for check marks in menu items.
Sub-menu arrow	Sets the image to be used to show that a menu item contains a submenu.
Separator image	Sets the image to be used for any separators between menu items.

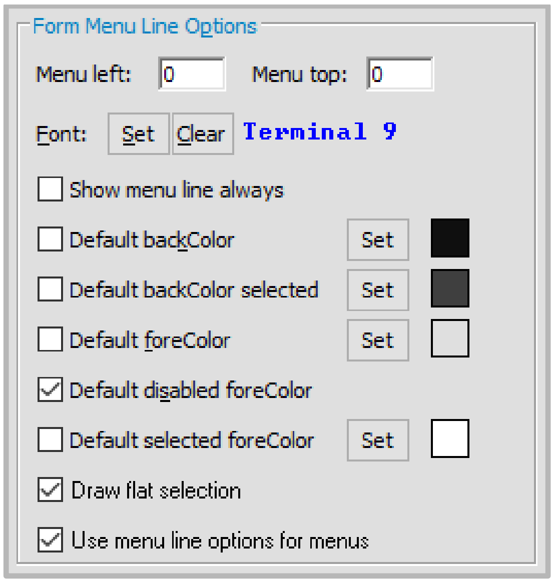
The Pixel Spacings group box on **Menus** sheet has the following text boxes, which enable you to specify the number of pixels used for spacing.

Check Box	Example
Before checkmark	
After checkmark	
After picture	
Before accelerator	
Before right Arrow	
Menu item height	

Exercise 4 – Adding a Forms Skin to an Application Skin

In this exercise, you will add a forms skin to your **DOS** application skin.

1. Run the JadeScript **createJadeSkinMaintenance** method to open the Jade Skin Maintenance dialog.
2. Select the **Forms** sheet on the Jade Skin Maintenance dialog.
3. Set the **Default backColor** to **Light Gray**.
4. In the Form Menu Line Options group box, set the following values.



Form Menu Line Options

Menu left: 0 Menu top: 0

Font: Set Clear Terminal 9

[ ] Show menu line always

[ ] Default backColor Set ■

[ ] Default backColor selected Set ■

[ ] Default foreColor Set □

[✓] Default disabled foreColor

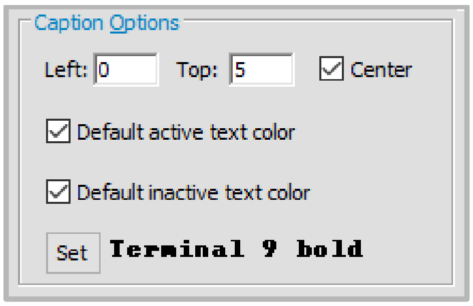
[ ] Default selected foreColor Set □

[✓] Draw flat selection

[✓] Use menu line options for menus

   a. **Font** to **Terminal**, size **9**
   b. **Default backColor** to **Black**
   c. **Default backColor selected** to **Dark Gray**
   d. **Default foreColor** to **Light Gray**
   e. **Default selected foreColor** to **White**
   f. Check the **Draw flat selection** check box

5. In the Caption Options group box, set the following values.



Caption Options

Left: 0 Top: 5 [✓] Center

[✓] Default active text color

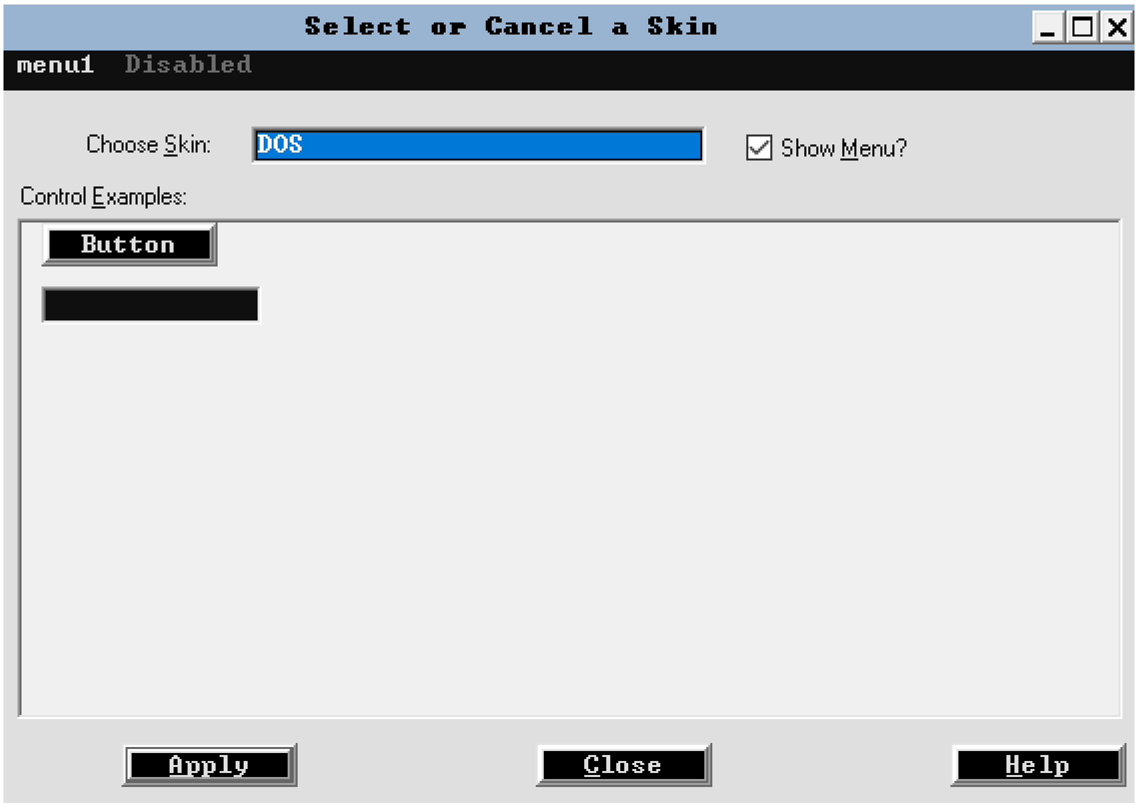
[✓] Default inactive text color

Set Terminal 9 bold

   a. **Top** margin to **5**
   b. Check the **Center** check box
   c. **Font** to **Terminal**, size **9**, **bold**

6. Save the forms skin as **frmDOS**.

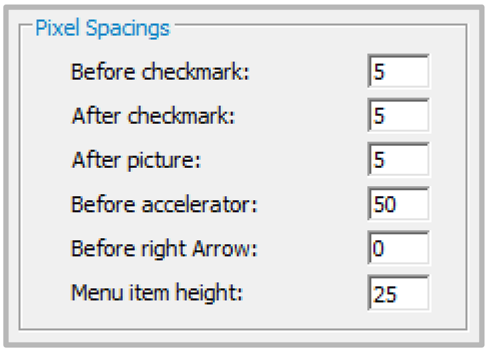
7. Preview the skin using the JadeScript **JadeSkinSelection** method. It should look like the following.



Exercise 5 – Adding a Menus Skin to an Application Skin

In this exercise, you will customize the menus skin for the **DOS** application skin.

1. Run the JadeScript **createJadeSkinMaintenance** method to open the Jade Skin Maintenance dialog.
2. Select the **Menus** sheet on the Jade Skin Maintenance dialog.
3. Set the **BorderStyle** to **0 - none**.
4. In the Pixel Spacings group box, set the values shown in the following image.

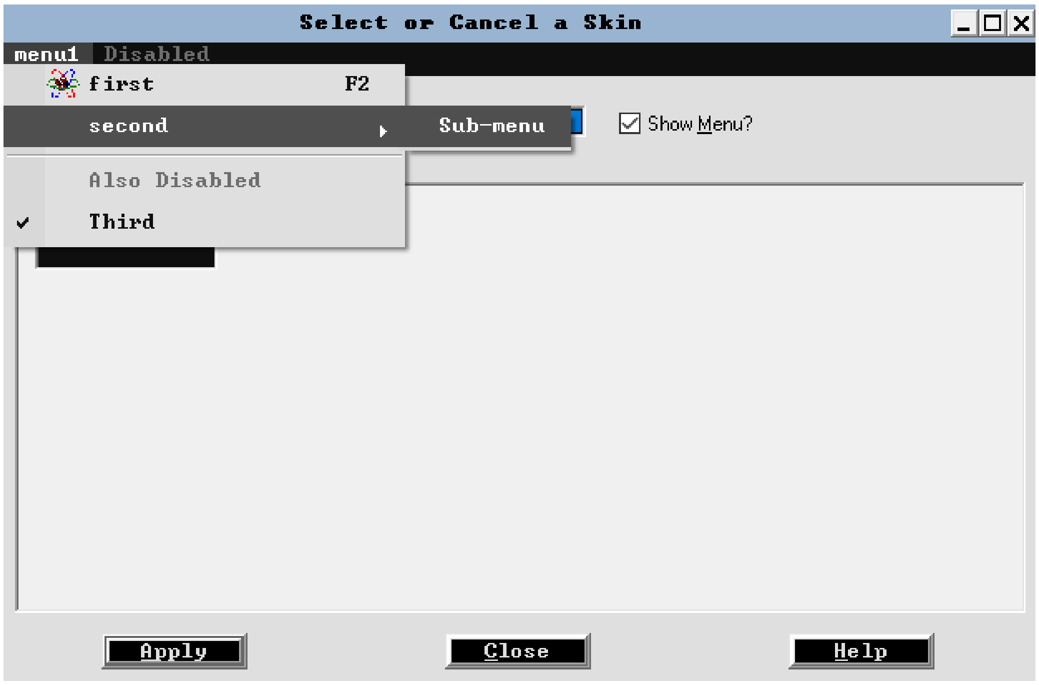


Pixel Spacings

Before checkmark:	5
After checkmark:	5
After picture:	5
Before accelerator:	50
Before right Arrow:	0
Menu item height:	25

5. Set the **Font** to **Terminal**, size **9**.
6. Set the **Default backColor** to **Light Gray**.
7. Set the **Default backColor selected** to **Dark Gray**.

8. Set the **Default foreColor** to **Black**.
9. Set the **Default selected foreColor** to **White**.
10. Check the **Draw flat selection** check box.
11. Save the menus skin as **mnuDOS**.
12. On the **Forms** sheet of the Jade Skin Maintenance dialog, set the **Popup Menu** option to **mnuDOS**.
13. Preview the skin using the JadeScript **JadeSkinSelection** method. It should look like the following.



JadeSkinRoot Class

JadeSkinRoot is the root object that contains collections of all skins (including **Application** skins as well as the **Control** and **Forms** skins that make up an application skin).

As **JadeSkinRoot** is a root object that uses the singleton pattern, it can be accessed safely with **JadeSkinRoot.firstInstance**.

Note The singleton pattern that is used for root objects means that that "There can be only one" is enforced. For more details, see **Module 7 - Root Object** of the JADE Developer's course.

The **JadeSkinRoot** class can be used to find specific skins with a dictionary lookup by name, as shown in the following example.

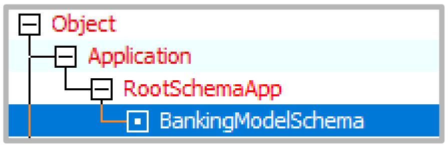
```
initialize() updating;

vars
    jsRoot      : JadeSkinRoot;
    exampleSkin : JadeSkinApplication;

begin
    jsRoot      := JadeSkinRoot.firstInstance;
    exampleSkin := jsRoot.allApplicationSkins["Example"];

    app.setApplicationSkin(exampleSkin);
end;
```

This method sets the application's skin to the skin called **Example** when the application is initialized.

Exercise 6 – Adding an Application Skin to an Application

In this exercise, you will apply the **DOS** application skin to the **Banking** application.

1. Open **BankingModelSchema** in the Class Browser.
2. Navigate to the **BankingModelSchema** application subclass.



3. Add a method called **applySkin** to the **BankingModelSchema** application class and code it as follows.

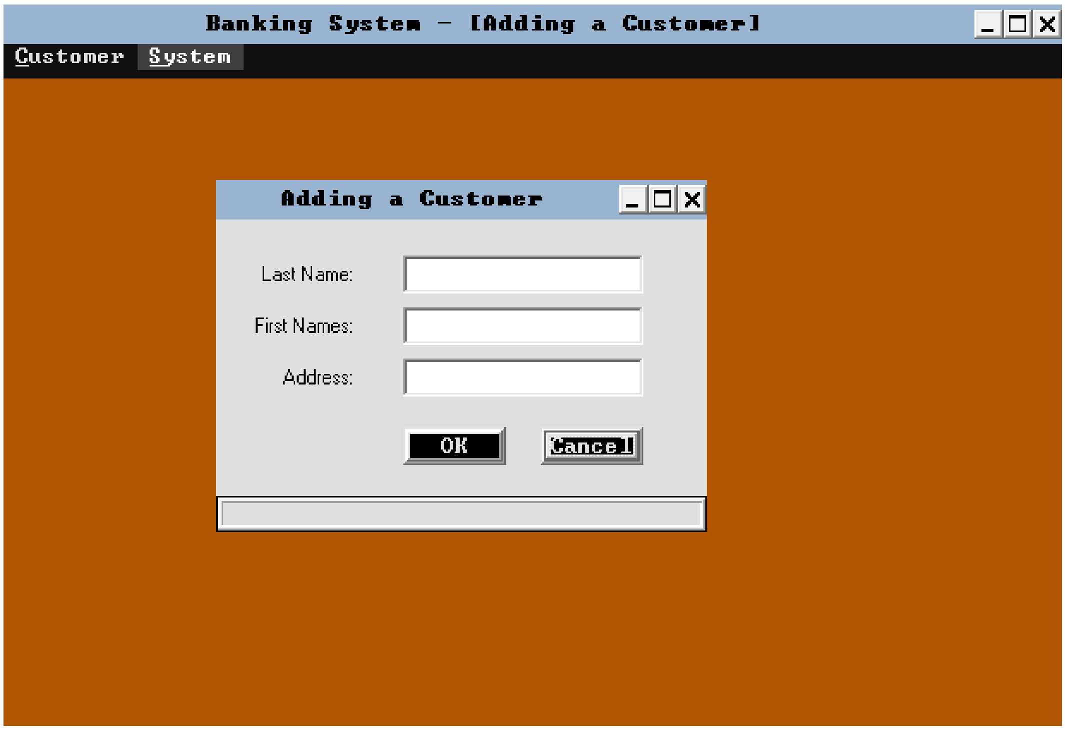
```
applySkin();

vars
    jsRoot  : JadeSkinRoot;
    skin    : JadeSkinApplication;
begin
    jsRoot  := JadeSkinRoot.firstInstance;
    skin    := jsRoot.allApplicationSkins["DOS"];
    app.setApplicationSkin(skin);
end;
```

4. Modify the **initialize** method in the **BankingModelSchema** application class to call the **applySkin** method, as follows.

```
initialize() updating;

begin
    self.applySkin();

    on Exception do self.genericExceptionHandler(exception) global;
    self.myBank := Bank.firstInstance();
    if self.myBank = null then
        beginTransaction;
        create myBank persistent;
        commitTransaction;
    endif;
end;
```

5. Run the **Banking** application and open the Add Customer dialog.

The **Banking** system should have the **DOS** application skin applied to it.


Banking System - [Adding a Customer]
Customer
System
Adding a Customer
Last Name:
First Names:
Address:
OK
Cancel