Displays Themes Reference

Apply consistent themes to displays.

Display Themes (Reference) provide consistent visual styling across Designer and Runtime environments, with built-in Light/Dark theme pairs optimized for various industrial settings from bright offices to dark control rooms.

Themes provide:

- Automatic styling for UI controls
- Light/Dark theme pairs
- High contrast accessibility options
- Custom color definitions
- Runtime theme switching
- OEM branding support

All themes apply automatically across Designer and Runtime without additional setup.

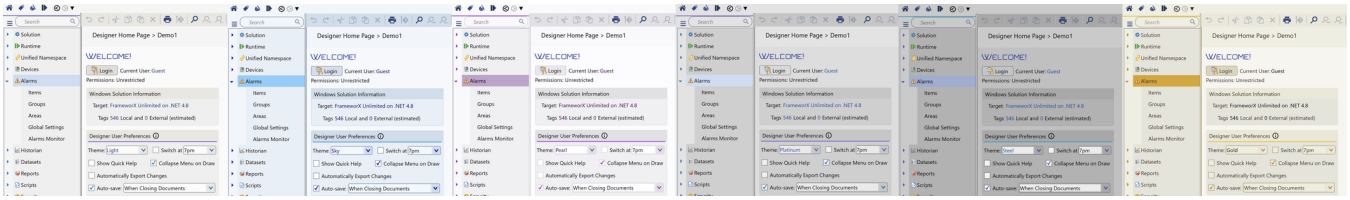
Built-in Themes
Applying Themes
Color Selection
Theme Color Structure
Customizing Themes
Customization
Migration from Legacy Projects
Runtime Theme Switching
Best Practices
Troubleshooting

Built-in Themes

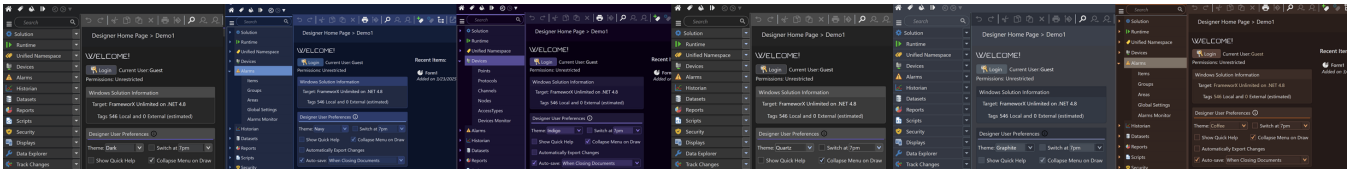
heme Pairs

Light Theme	Dark Theme	Description
Light	**Dark**	Clean white and classic dark
Platinum	**Onyx**	Subtle gray with sharp black
Steel	**Graphite**	Cool grayscale minimal feel
Pearl	**Indigo**	Soft lilac with rich purple
Sky	**Navy**	Airy blue with deep navy
Gold	**Coffee**	Elegant gold with earthy brown
ContrastLight	**ContrastDark**	Maximum accessibility

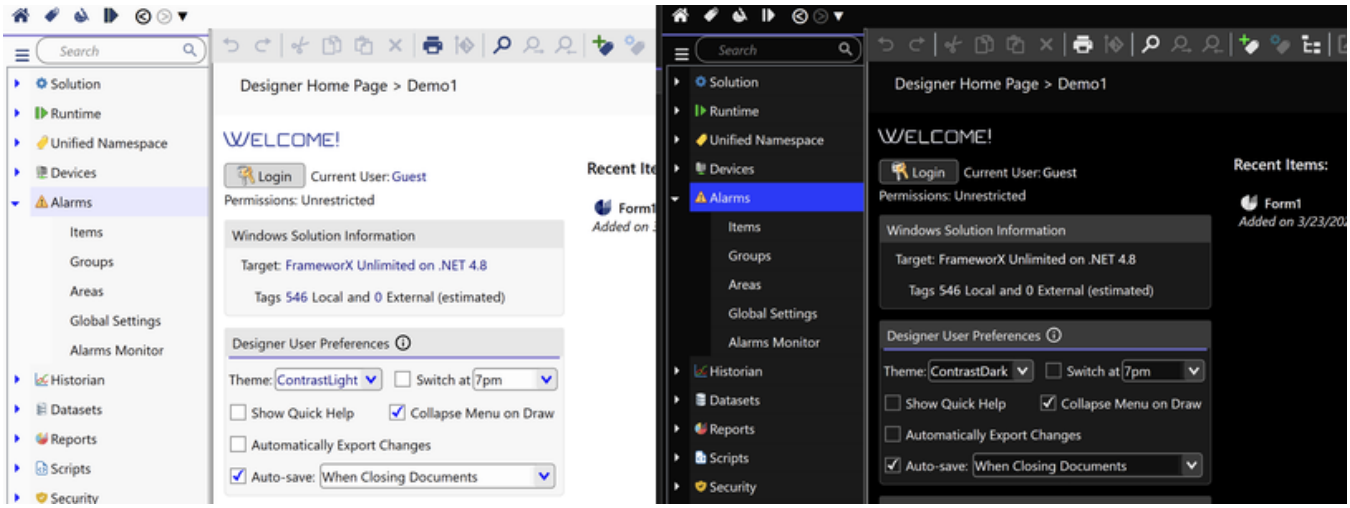
Light Themes



Dark Themes



High Contrast Themes



Themes Description

Theme Name	Description
Light	A clean, neutral light theme ideal for bright environments.
Platinum	A refined light-gray theme with balanced contrast and minimalism.
Steel	A modern, slightly darker light theme with a cool tone.
Pearl	A soft light theme paired with indigo highlights for subtle contrast.
Sky	A light theme with cool blue undertones, refreshing and calm.
Gold	A warm, inviting light theme with golden accents.
ContrastLight	A high-contrast light theme optimized for accessibility.
Dark	A classic dark theme with soft accents for reduced eye strain.
Onyx	A smooth, elegant dark theme with refined styling.
Graphite	A muted, grayscale dark theme with an industrial edge.
Navy	A rich dark theme with deep blue tones and crisp contrast.
Coffee	A cozy dark theme with warm brown hues for a grounded feel.
ContrastDark	A high-contrast dark theme for enhanced readability.

Applying Themes

Designer Configuration

1. **Welcome Page** - Select theme at startup
2. **User Preferences** - Change anytime
3. **Display Editor** - Preview different themes

Runtime Selection

csharp

```
// Set theme dynamically
@Client.Theme = "Dark";

// Check current theme
string current = @Client.Theme;

// Reset to startup theme
@Client.Theme = "";
```

Startup Theme

Configure default at **Displays Client Settings Initial Execution Conditions**

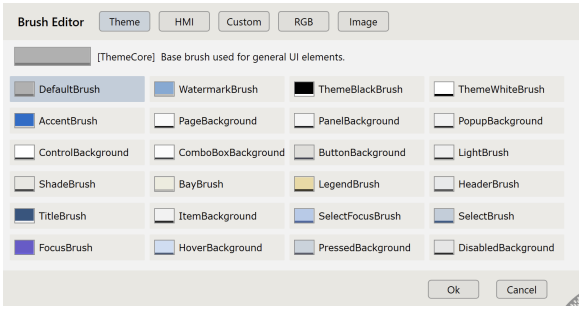
Color Selection

BrushEditor Dialog

When editing colors in Designer:

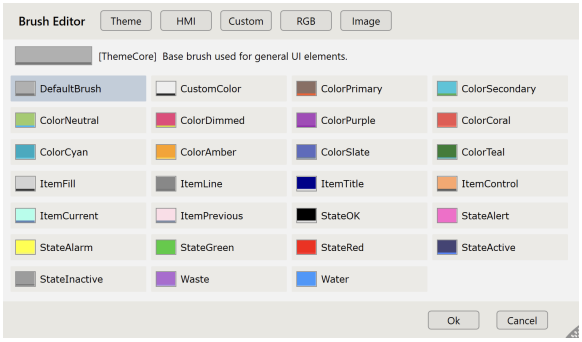
- **Theme Tab** - General-purpose colors
- **HMI Tab** - High Performance HMI colors
- **Custom Tab** - Solution-specific colors

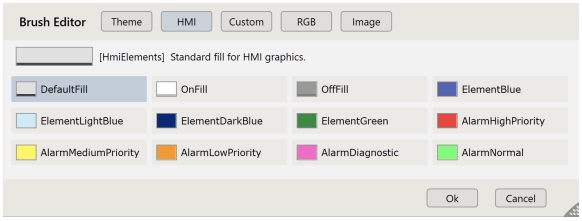
Colors automatically update when themes change.



The Theme, HMI (Hight Performance HMI) and Custom present the colors that are automatically updated when a global theme is applied to the solution.

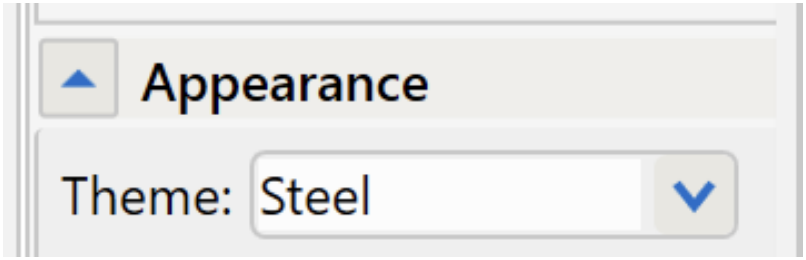
- The **Theme** tab lists general-purpose colors, automatically filtered based on the selected property—such as background, border, or text.
- The **HMI** tab contains colors that follow High Performance HMI Graphics guidelines.
- The **Custom** tab are pre-defined colors for general purpose that can have the name and color customized to each solution.





Preview Runtime Appearance

In the display editor, the Properties Sidebar includes an Appearance section with a Theme dropdown to preview how the display appears under different themes.



That option will change the preview of the display, when running on the Designer tool.

When running the solution, change dynamically the Theme using the property @Client.Theme.

Theme Colors in Color Thresholds

From FrameworX 10.1.6, the limits of the TextColor, LineColor and FillColor dynamics accept a theme color name, such as `StateAlarm`, when you build displays in JSON or with the AI tools. Those limit colors change with the theme, the same way element colors do. See Dynamic Behavior Control Reference for the syntax.

Theme Color Structure

The platform provides a structured color definition system. These colors are applied automatically by the system when creating UI controls, and can also be used directly in custom graphics, using the previously showed color selection dialogs.

The page **Displays Themes** in the Solution Explorer has the full table will allow pre-defined theme colors. Those colors are organized in the following main groups:

- Theme Colors
- High Performance HMI Graphics
- Custom Colors

The General Theme Colors is organized in the following groups:

- General Theme Colors
- Main Backgrounds
- GroupBox Elements
- Diagram Elements
- Text Elements
- Dashboard Elements
- State and Interaction
- Disabled Controls

Below is a list of all standard theme colors, grouped by their functional role

---

Core Theme Colors

General Theme Colors

Color Name	Description
DefaultBrush	Base color for generic content
DefaultBorder	Border color for default elements
WatermarkBrush	Watermark or placeholder text color
ThemeBlackBrush	System black
ThemeWhiteBrush	System white
AccentBrush	Primary accent color for key elements

Main Backgrounds

Color Name	Description
PageBackground	Global background color for pages
PanelBackground	Panel or container background
PopupBackground	Background for popups or dialogs
ControlBackground	General control (e.g. textbox) background
ComboBoxBackground	Combo box background
ButtonBackground	Standard button background

GroupBox Elements

Color Name	Description
LightBrush	Inner brush for group-like controls
ShadeBrush	Outer/contrast edge for grouped zones

Diagram Elements

Color Name	Description
BayBrush	Diagrams or structural sections
LegendBrush	Text/legend colors for diagrams

Text Elements

Color Name	Description
TextForeground	Primary text color, high contrast
TextAccentForeground	Accent text color for emphasis
TextSubtleForeground	Subdued text for hints or comments
TextHyperlinkForeground	Hyperlink blue for clickable text

Dashboard Elements

Color Name	Description

Color Name	Description

HeaderBrush	Section headers
TitleBrush	Panel titlebars

ItemBackground	Item content block
ItemBorder	Border around blocks

State and Interaction

Color Name	Description
SelectFocusBrush	Focused item border
SelectBrush	Highlighted selection
FocusBrush	Keyboard or mouse focus indicator
HoverBackground	Background when hovered
HoverBorder	Border when hovered
PressedBackground	Active press background
PressedBorder	Border on press

Disabled Controls

Color Name	Description
DisabledForeground	Disabled/inactive text color
DisabledBackground	Background for disabled fields
DisabledBorder	Border around disabled controls

High Performance HMI Graphics

Standard Colors

Color Name	Description
DefaultFill	HPG base fill
DefaultStroke	HPG base stroke
OnFill	HPG “ON” fill color
OnStroke	HPG “ON” stroke color
OffFill	HPG “OFF” fill color
OffStroke	HPG “OFF” stroke color
ElementBlue	Standard blue for HPG blocks
ElementLightBlue	Light blue accent in HPG elements
ElementDarkBlue	Deep blue variant for HPG
ElementGreen	Green fill used in HPG indicators

Alarm Colors

Color Name	Description
AlarmHighPriority	High severity alert
AlarmMediumPriority	Medium alert
AlarmLowPriority	Low severity warning
AlarmDiagnostic	Diagnostic / info condition
AlarmNormal	Default or cleared state

Custom Colors

General Colors

Each theme includes a consistent set of color brushes designed for use in dynamic symbols, dashboards, and runtime visuals. These colors provide both visual consistency and functional clarity across light and dark environments.

ColorName	Description
CustomColor	Theme Neutral brush, general use
ColorPrimary	Contrast color, Taupe-brown for general UI accents
ColorSecondary	Contrast color, Aqua for secondary strokes or graphics
ColorNeutral	Contrast color, Lime green for subtle visual contrast
ColorDimmed	Muted rose pink for soft accents or tags
ColorPurple	Orchid purple for thematic color or data status
ColorCoral	Coral pink-red for warnings or minor alerts
ColorCyan	Cyan blue for visualizations and metrics
ColorAmber	Amber for status or caution

ColorSlate	Slate indigo for rich accents
ColorTeal	Dark teal green for stable UI components
ItemFill	Pastel background fill for neutral UI panels
ItemLine	Gray line color for outlines and separators.
ItemTitle	Deep blue for bold titles
ItemControl	Orange accent used on control surfaces
ItemCurrent	Bright mint to indicate current item
ItemPrevious	Light pink to represent a previous value or state
StateOK	Default OK state, black for cleared or normal
StateAlert	Bright alert magenta state
StateAlarm	Yellow alarm state, may blink or pulse
StateGreen	Indicates running or active status
StateRed	Stop or error status, critical fault
StateActive	Muted blue for current or active states
StateInactive	Gray to show inactive or idle
Waste	Lilac tone for waste process visuals
Water	Blue tone used to represent water elements.

Customizing Themes

Customization

Modifying Custom Colors

1. Navigate to **Displays Themes**
2. Edit colors with ID 100
3. Modify CustomLight/CustomDark themes

<ac:structured-macro ac:name="info"> ac:rich-text-body System colors (ID < 100) are protected to ensure UI consistency. Only Custom Colors can be renamed or recolored. </ac:rich-text-body> </ac:structured-macro>

Creating Custom Themes

Use CustomLight and CustomDark columns:

1. Start from existing theme
2. Modify all color values
3. Save as new theme set



System Colors

You can only customize any Color after CustomColor (ID >= 100), or any color in CustomLight and CustomDark themes. The other names in and colors in the table, which as system colors, are protected.

Custom Light and Dark Themes

The table includes two columns for a CustomLight and a CustomDark theme, which are originally exactly the same of the Light and DarkThemes.

Use those columns if you to have a complete new theme in your solution.

Reset Colors Command

Attention when using the RESET or Paste commands on that user interface and they have an undo operation.

The Reset will overwrite all your CustomColors and the Custom Themes back to original software settings.

Migration from Legacy Projects

Color Mapping File

Create `ProjectName-CustomColors.csv`:

csv

```
ID,ItemName,ColorName
1030,PanelBack,LightBrush
1041,AlertIndicator,StateAlarm
1042,AlarmOK,StateOK
```

- **ID**: Legacy color ID
- **ItemName**: New display name
- **ColorName**: Target theme color

Migration Process

1. Prepare CSV mapping file
2. Import legacy project
3. System applies mappings
4. Adjust colors in Themes table

In previous releases, the ColorID (and its names) could be completely customized for each project. Although this created flexibility, it also caused issues when trying to create reusable components across multiple projects.

Another difference: it was very common in previous releases for the Theme features to be used simply as a dictionary of colors, in order to keep the color selection consistent without actually changing the Theme. The application had only one theme, and the selection was just a dictionary of names.

In version 10, colors are still completely customizable. However, a set of standard colors, used by Windows Controls, popup backgrounds, dashboards, and other standard UI elements, now have a fixed ID to ensure consistency across multiple solutions and when applying Dark/Light theme changes.

When migrating from previous versions, you can retain specific color customizations by preparing a CSV file named <ProjectName>-CustomColors.csv, mapping the IDs to new standards. This file is automatically applied during the import of a legacy project (<ProjectName>.tproj).

Preparing a <Project>.CustomColors.csv File

The file must follow this format:

ID	ItemName	ColorName

- **ID** and **ItemName**: These two columns reference the customized color definitions from your legacy system. This data is typically extracted from the Run / Dictionaries / Themes table of earlier versions.
- **ColorName**: This third column specifies which theme resource in the current system the legacy color should map to.

The first column, ID**,** is the color you want to map (from the Run / Dictionaries / Themes in previous releases).

The third column, ColorName, is the name of the color in the current release that the resource will be mapped to.

CustomColors Algorithm

When migrating the displays and symbols, the colors from legacy project mapped to the **ID,** will be translated to the id's of the ColorName.

The second column, ItemName, has no effect on the ID translation. However, it can be used to change the name of the custom colors in the new solution. If you want to rename a ColorName, specify the new name in the ItemName column.
If you want to keep the standard ColorName, simply repeat the ColorName value in the ItemName column.

Here is an example of configuration file. The Items in Bold, the decision was to keep the new name.

Project1-CustomColors.csv

```
ID,ItemName,ColorName
1030,PanelBack,LightBrush
1041,AlertIndicator,StateAlarm
1042,AlarmOK,StateOK
1043,DataColor,ItemTitle
1044,LabelColorLight,ThemeWhiteBrush
1046,BadData,StateAlert
1047,RunTrue,StateGreen
1048,RunFalse,StateRed
1049,LabelColorDark,ThemeBlackBrush
1050,ItemGlow,SelectBrush
1052,Water,Water
1053,ItemControl,ItemControl
1054,CurrentDayBG,ItemCurrent
1055,PreviousDayBG,ItemPrevious
```

Note that for IDs 1052 and 1053, the ItemName was set to be the same as the ColorName, meaning the system will translate the IDs, but the name of the color will be kept to the new standard.

When trying to apply new names, only the names in the CustomColors group will accept the new names, the predefined colors for General Theme Colors and HMI will have their IDs translated but keep their standard names.

You don’t need to include every color from the legacy project, only the customized ones or those for which the default mapping does not meet your requirements. Standard theme roles are applied automatically during import.

After import, you can further refine or rename the colors directly in the Displays / Themes configuration table within the Designer.

The final step of the migration process is executed directly in the new Solution by editing the DisplaysThemes table.

The importer tool translated the resource IDs in the displays to the new IDs, but the color that corresponds to each ID was not changed. This is because it would not be possible for the importer to know, among the many themes from the previous release, which one you used as a reference for your color dictionary, nor which of the new themes would be your replacement.

In the themes you intend to use, if the custom color RGB is not what you want, you can customize it by editing that table.

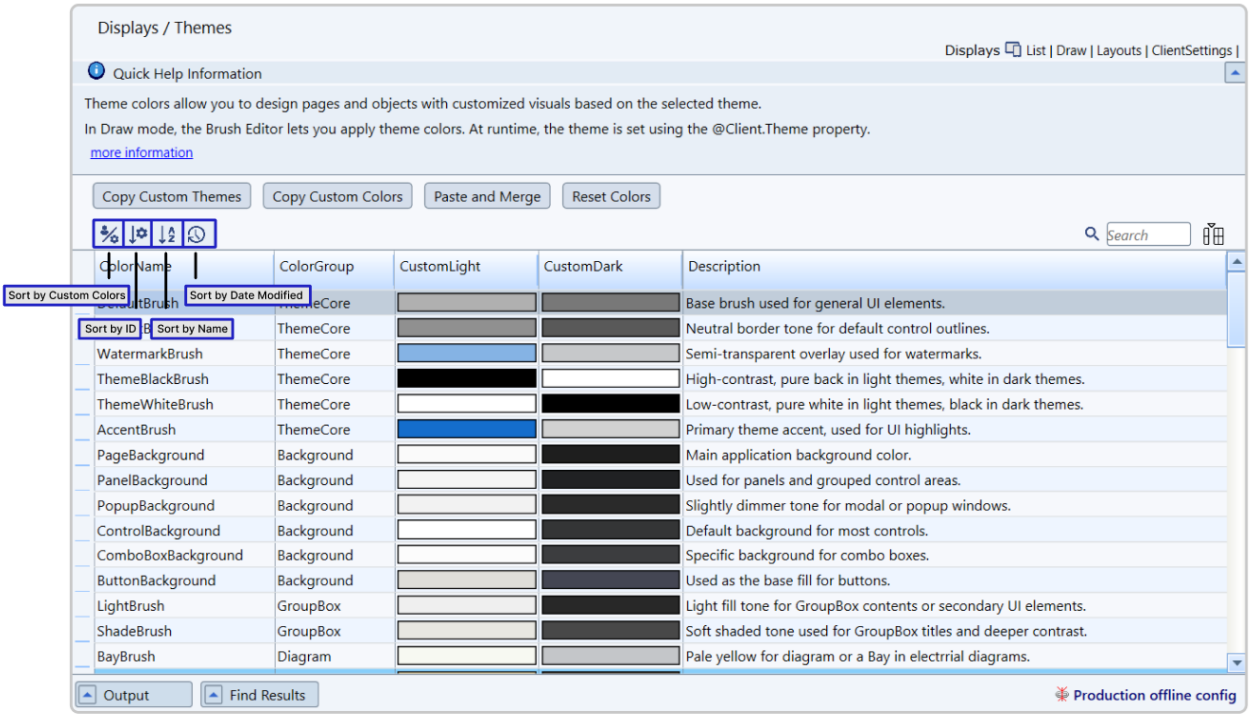
You can either edit the colors directly by clicking on the color you want to change or export and import all the CustomColors to a CSV file.

The colors that allow customization are all colors in the CustomDark and CustomLight themes, as well as any colors with ID > 200 (from CustomColors downward in the themes table). Core theme colors cannot be changed to ensure consistency across different displays and their interactive controls.

Sorting and Filtering the Themes Page

You can customize your visualization on the Themes page by applying sorting and filtering options according to your preferences.

You can filter by All Themes or Custom Themes, and sort by ID, Name, or Date Modified by clicking to the correspondent button. Additionally, you can right click on the Column Headings to find even more customability options.



Migration from Legacy Projects

Color Mapping File

Create `ProjectName-CustomColors.csv`:

csv

```
ID,ItemName,ColorName
1030,PanelBack,LightBrush
1041,AlertIndicator,StateAlarm
1042,AlarmOK,StateOK
```

- **ID**: Legacy color ID
- **ItemName**: New display name
- **ColorName**: Target theme color

Migration Process

1. Prepare CSV mapping file
2. Import legacy project
3. System applies mappings
4. Adjust colors in Themes table

Runtime Theme Switching

User Preference

csharp

```
public void ToggleTheme()
{
    if (@Client.Theme == "Light")
        @Client.Theme = "Dark";
    else
        @Client.Theme = "Light";
}
```

Time-Based

csharp

```
// Auto dark mode at night
if (DateTime.Now.Hour >= 18 || DateTime.Now.Hour < 6)
    @Client.Theme = "Dark";
else
    @Client.Theme = "Light";
```

Best Practices

- [ ] **Choose Appropriate Pairs** - Match environment lighting
- [ ] **Test Accessibility** - Verify contrast ratios
- [ ] **Use Theme Colors** - Avoid hardcoded values
- [ ] **Document Custom** - Explain color purposes
- [ ] **Preview Changes** - Test all themes
- [ ] **Consider Users** - Support preferences
- [ ] **Maintain Consistency** - Follow standards

Troubleshooting

Theme not applying:

- Check Client.Theme property
- Verify theme name spelling
- Review startup settings
- Test theme preview

Colors not updating:

- Use theme colors, not hardcoded
- Check color binding
- Verify theme selection
- Review custom overrides

Migration issues:

- Verify CSV format
- Check ID mappings
- Review color names
- Test with sample

In this section...
