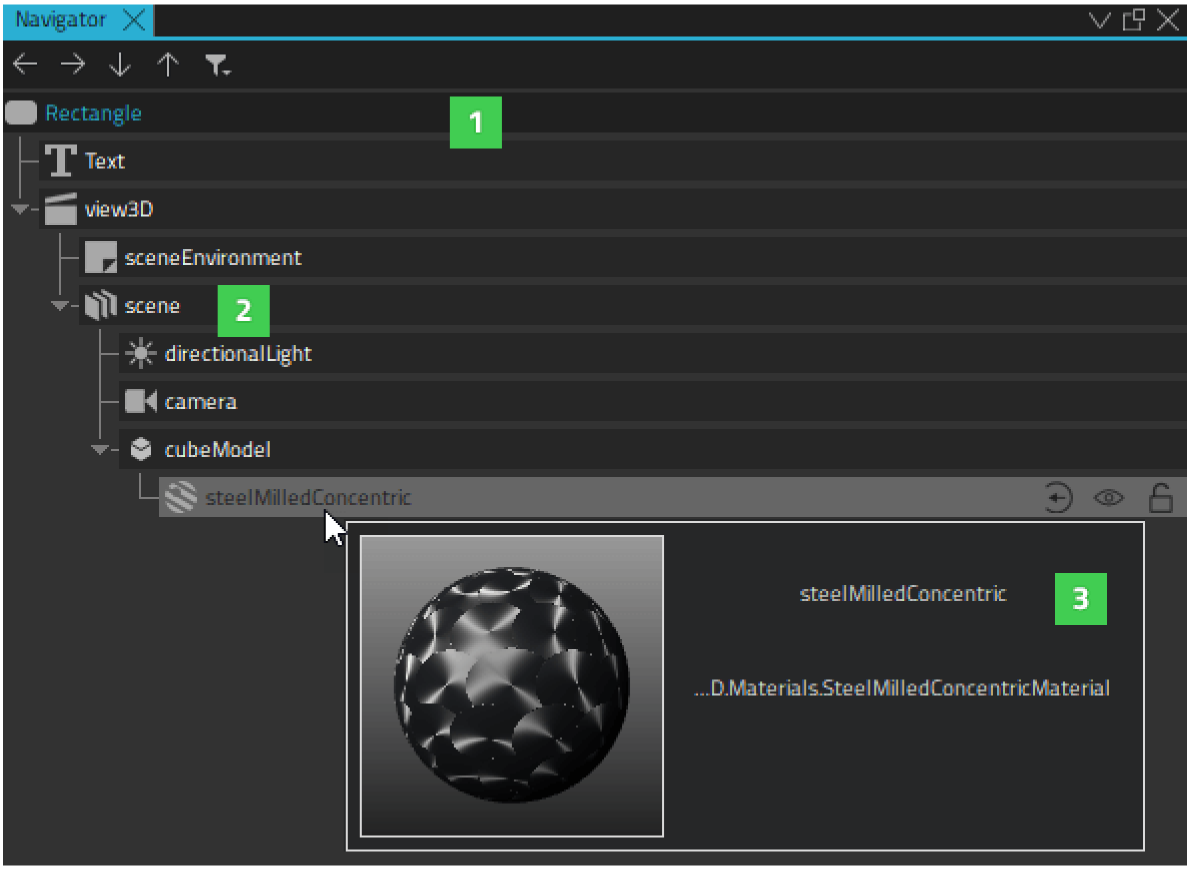
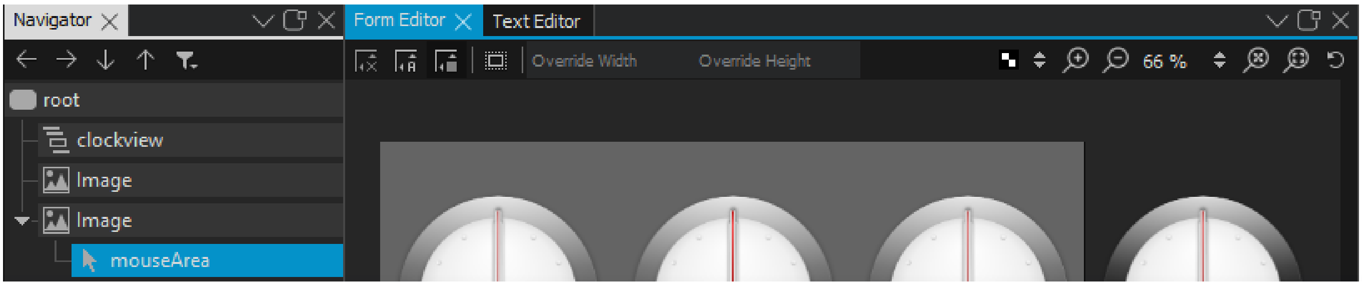
航海家

“**导航器**”视图显示当前组件文件中的组件及其关系。组件（1）列在树结构中，位于其父级（2）下方。您可以通过将鼠标悬停在组件上来预览组件（3）。



可以在“**导航器**”视图中选择组件，以便在“属性”视图中编辑其属性。组件可以访问其父组件的属性。要在 2D 视图中选择零部件，请在零部件上单击鼠标右键，然后在“**选择**”子菜单中选择另一个零部件。

通常，子组件位于 2D 视图中的父组件内。但是，它们不一定必须适合父组件。例如，您可能希望使鼠标区域大于其下方的矩形或图像。



复制组件时，也会复制其所有子组件。删除组件时，子组件也会被删除。

选择上下文菜单命令以将命令应用于组件。命令的可用性取决于组件类型。例如，可以通过在上下文菜单中选择"**更改源** URL"来更改图像组件的源。

导航器按钮摘要

下表列出了"**导航器**"按钮：

图标	工具提示	阅读更多
←	将组件在组件树中向上移动一级，以便它成为其当前父级的最后一个同级组件。	排列组件
→	将组件在组件树中向下移动一级，以便它成为其最后一个同级组件的子级。	排列组件
↓	将组件在其父级中下移。	排列组件
↑	将组件在其父级中上移。	排列组件
▼	在**导航器**中显示和隐藏不可见的组件。	显示和隐藏组件
@	添加可从组件外部使用的属性别名。	添加属性别名
👁	在 2D 视图中显示和隐藏组件。	显示和隐藏组件
🔒	锁定所有视图中的组件。	锁定组件

显示和隐藏组件

要在聚焦于应用程序的特定部分时在 2D 视图中显示和隐藏组件，请在"**导航器**"中单击👁。

To change the visibility of a component in the application code, select the **Visibility** check box in the **Properties** view or select **Edit** > **Visibility** in the context menu.

You can also set the **Opacity** field to 0 in **Properties** to hide components in the UI that you want to apply animation to.

As all properties, visibility and opacity are inherited from the parent component. To hide or show child components, edit the properties of the parent component.

To hide invisible components in **Navigator**, click ▼ (**Filter Tree**) and select **Show Only Visible Components**.

Locking Components

When designing complex applications, it is easy to accidentally modify the properties of a component in one of the Qt Design Studio views in ways that lead to surprising results. For example, the **2D** view can become crowded and other components can get in the way when you are trying to select or transform a particular component, so that you end up transforming more components than you wanted to.

You cannot select locked components in the **2D** view or the 3D view nor access their properties in **Properties**.

If you attempt to remove a state that changes the properties of a locked component, you are prompted to confirm the removal.

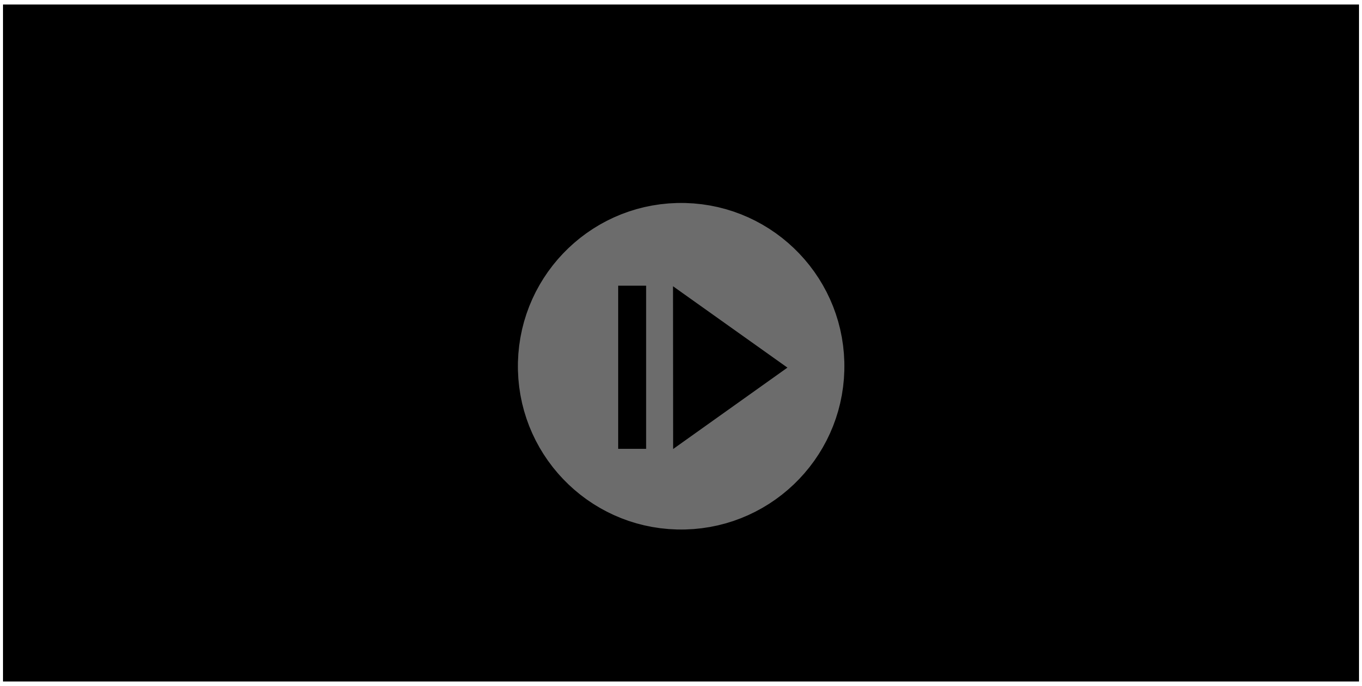
If you have added easing curves to keyframe animations, you can lock and unlock them in the Curves view. If you lock the components that contain the easing curves, the lock status is synchronized between **Navigator** and **Curves**.

Arranging Components

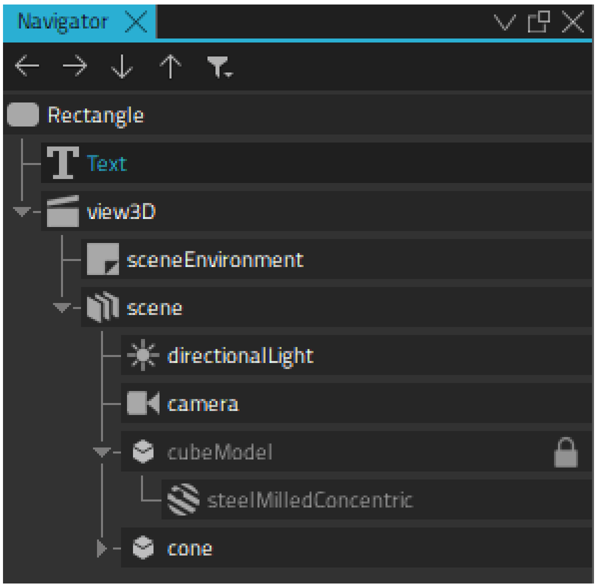
You can view the order of components in a component file in **Navigator** and the Code view. The order of components in the file also determines the order in which they are drawn in the **2D** view. By default, components that are located at the top of the file are listed at the bottom of the **Navigator** tree and behind overlapping components in the **2D** view. To list the components in the order in which they appear in the file, as some other tools do, click ▼ (**Filter Tree**), and select **Reverse Component Order**.

To move a component to the top or bottom of the tree within its parent, right-click it in the **Navigator** or **2D** view and select **Arrange** > **Bring to Front** or **Send to Back**. To move a component up or down, select **Bring Forward** or **Send Backward**.

To reverse the order of the selected components in the **Navigator** and **Code** views, select **Arrange** > **Reverse**.



You can also drag-and-drop the component to another position in the tree or use the arrow buttons to move the component in the tree. You can use the left and right arrow buttons to change the parent of the component.



When you drag-and-drop instances of components to the **2D** view, the new component is added as a child of the component beneath it. When you move the components, it is not possible to determine whether you want to adjust their position or attach them to a new parent component. Therefore, the parent component is not automatically changed. To change the parent of the component, press down the **Shift** key before you drag-and-drop the component into a new position. The topmost component under the cursor becomes the new parent of the component.

Adding Property Aliases

A *property alias* is a property that you can use from outside the component. When you view the code in the Code view, a property alias declaration looks like an ordinary property definition, except that it requires the *alias* keyword instead of a property type, and the right-hand-side of the property declaration must be a valid alias reference:

```
property alias <name>: <alias reference>
```

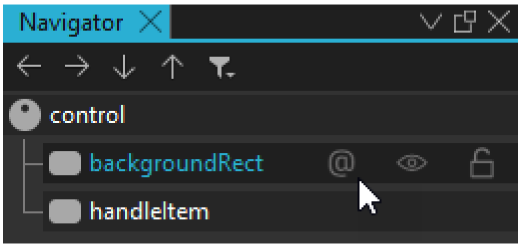
For example, the following alias refers to a button component instance within an item component instance:

```
property alias button: item.button
```

A valid alias reference:

- Can only point to a component instance or property within the component where the property alias is declared.
- Cannot contain arbitrary JavaScript expressions.
- Cannot point to components of another type than the component where the property alias is declared.
- Must be defined when the alias is first declared.
- Cannot point to attached properties.

You can use the @ (**Export**) button in **Navigator** to export a component as a property alias with a valid alias reference.



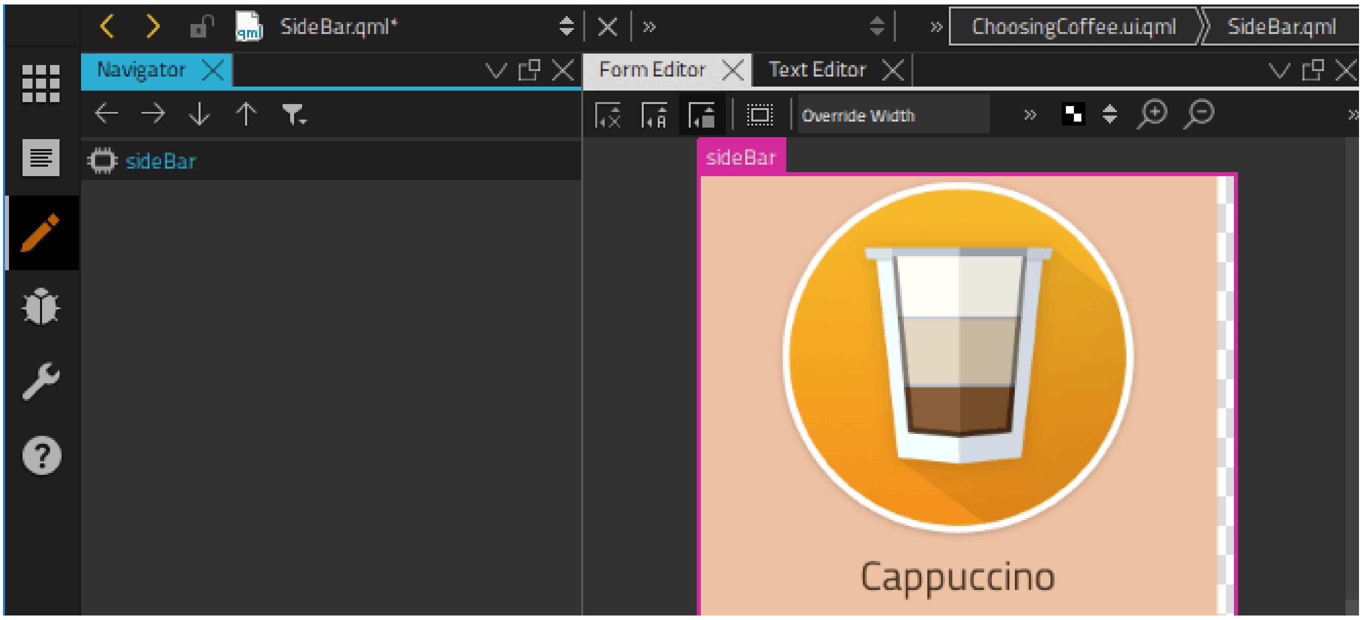
You can then use the property alias in other components to create connections to this component.

Moving Within Components

The files that specify components (,) can contain instances of other components specified in separate files. You can open the file that specifies a component in different ways from different views:`ui.qml.qml`

- In the **2D** or **Navigator** view, right-click an instance of a component and then select **Go into Component** in the context menu or press **F2**.
- In **Properties**, select **Edit Base Component**.

The component hierarchy is displayed as a bread crumb path, where you can click the component names to open the respective files. This enables you to easily navigate back to the top level when you are done editing the component.



Context Menu

The following table summarizes the **Navigator** and **2D** views context menu items and provides links to more information about them.

To Learn About	Go To
Arrange	Arranging Components

Anchors	Setting Anchors and Margins
Group	Organizing Components
Position	Using Positioners
Layout	Using Layouts
Stacked Container	Lists and Other Data Models
Timeline	Creating a Timeline
Event List	Simulating Events
Edit Color	Editing Properties Inline
Edit Annotation	Annotating Designs
Merge File with Template	Merging Files with Templates
Move Component Instances into Separate Files	Turning Component Instances into Custom Components
Add New Signal Handler	Adding Signal Handlers
Go to Implementation	Using UI Files
Go into Component	Moving Within Components

‹ Assets

Properties ›

Qt The Qt Company

Contact Us

Company

About Us

Investors

Newsroom

Careers

Office Locations

Licensing

Terms & Conditions

Open Source

FAQ

Support

Support Services

For Customers

Support Center