

UI Files

If you switch between Qt Creator and Qt Design Studio or cooperate with designers on a project, you might encounter UI files (.ui.qml). They are intended to be edited in Qt Design Studio only.

The following features are not supported in .ui.qml files:

- › JavaScript blocks
- › Other bindings than pure expressions
- › Signal handlers
- › States in other components than the root component
- › Root components that are not derived from `QQuickItem` or `Item`
- › Referencing the parent of the root component

The following components are not supported:

- › Behavior
- › Binding
- › Canvas
- › Shader Effect
- › Timer
- › Transform

Supported Methods

Qt Creator supports most JavaScript functions that are supported by the QML engine, as well as a subset of Qt QML methods.

This section lists the functions that you can use in .ui.qml files.

JavaScript Functions

As a rule of thumb, *pure functions* are supported. They only depend on and modify states of parameters that are within their scope, and therefore always return the same results when given the same parameters. This makes it possible to convert and reformat property bindings without breaking the .ui.qml files.

The following JavaScript functions are supported:

- › `charAt()`

```
concat()  
> endsWith()  
> includes()  
> indexOf()  
> isFinite()  
> isNaN()  
> lastIndexOf()  
> substring()  
> toExponential()  
> toFixed()  
> toLocaleLowerCase()  
> toLocaleString  
> toLocaleUpperCase()  
> toLowerCase()  
> toPrecision()  
> toString()  
> toUpperCase()  
> valueOf()
```

In addition, all functions of the `Math` and `Date` objects are supported.

For more information, see [List of JavaScript Objects and Functions](#).

Qt QML Methods

Qt Creator supports color methods, helper methods for creating objects of specific data types, and translation methods.

The following color methods are supported:

```
> Qt.darker()  
> Qt.hsla()  
> Qt.hsva()  
> Qt.lighter()  
> Qt.rgba()  
> Qt.tint()
```

The following helper methods are supported:

```
> Qt.formatDate()  
> Qt.formatDateTime()  
> Qt.formatTime()  
> Qt.matrix4x4()  
> Qt.point()
```



- › [Qt.size\(\)](#)
- › [Qt.vector2d\(\)](#)
- › [Qt.vector3d\(\)](#)
- › [Qt.vector4d\(\)](#)

The following translation methods are supported:

- › [qsTr\(\)](#)
- › [qsTranslate\(\)](#)
- › [qsTranslateNoOp\(\)](#)
- › [qsTrId\(\)](#)
- › [qsTrIdNoOp\(\)](#)
- › [qsTrNoOp\(\)](#)

Note: Do not mix translation methods in a UI file.

For more information about using the methods, see [Qt QML Methods](#).

[‹ Converting UI Projects to Applications](#)

[Using QML Modules with Plugins ›](#)

© 2022 The Qt Company Ltd. Documentation contributions included herein are the copyrights of their respective owners. The documentation provided herein is licensed under the terms of the [GNU Free Documentation License version 1.3](#) as published by the Free Software Foundation. Qt and respective logos are trademarks of The Qt Company Ltd in Finland and/or other countries worldwide. All other trademarks are property of their respective owners.



[Contact Us](#)

Company

[About Us](#)
[Investors](#)
[Newsroom](#)
[Careers](#)
[Office Locations](#)

Licensing

[Terms & Conditions](#)
[Open Source](#)
[FAQ](#)



Professional Services
Partners
Training

Downloads
Qt Login
Contact Us
Customer Success

Community

Contribute to Qt
Forum
Wiki
Downloads
Marketplace

© 2022 The Qt Company

[Feedback](#) [Sign In](#)