Qt Quick for Qt Developers

Composing Qt Quick User Interfaces

Based on Qt 5.4 (QtQuick 2.4)

Contents

The Qt Company

- Nested Elements
- Graphical Elements
- Text Elements
- Anchor Layout

Objectives

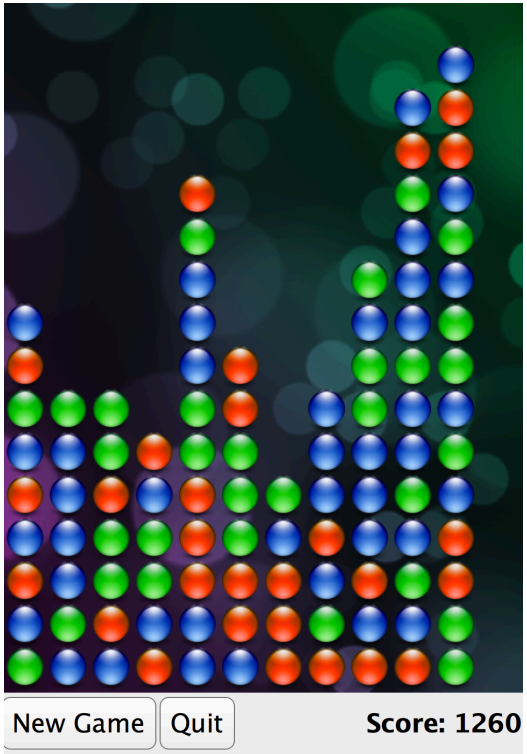
The Qt Company

- Elements are often nested
  - One element contains others
  - Manage collections of elements
- Colors, gradients and images
  - Create appealing UIs
- Text
  - Displaying text
  - Handling text input
- Anchors and alignment
  - Allow elements to be placed in an intuitive way
  - Maintain spatial relationships between elements

Why Use Nested Items, Anchors and Components?

The Qt Company

- Concerns separation
- Visual grouping
- Pixel perfect items placing and layout
- Encapsulation
- Reusability
- Look and feel changes



Demo: <Qt Examples>/declarative/demos/samegame/

Nested Elements

Nested Elements

The Qt Company

```
Rectangle {
    width: 400; height: 400
    color: "lightblue"
    Rectangle {
        x: 50; y: 50; width: 300; height: 300
        color: "green"
        Rectangle {
            x: 200; y: 150; width: 50; height: 50
            color: "white"
        }
    }
}
```

- Each element positioned relative to its parents

Demo: qml-composing-uis/ex-elements/nested2.qml

Graphical Elements

Colors

The Qt Company

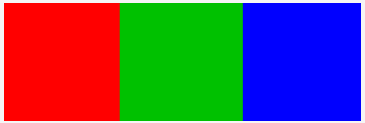
- Specifying colors
  - Named colors (using SVG names): `"red"`, `"green"`, `"blue"`,...
  - HTML style color components: `"#ff0000"`, `"#008000"`, `"#0000ff"`,...
  - Built-in function: `Qt.rgba(0,0.5,0,1)`
- Changing items opacity:
  - Using the `opacity` property
  - Values from `0.0` (transparent) to `1.0` (opaque)

See Documentation: QML Basic Type colors

Colors

The Qt Company

```
Item {
    width: 300; height: 100
    Rectangle {
        x: 0; y: 0; width: 100; height: 100; color: "#ff0000"
    }
    Rectangle {
        x: 100; y: 0; width: 100; height: 100 color: Qt.rgba(0,0.75,0,1)
    }
    Rectangle {
        x: 200; y: 0; width: 100; height: 100; color: "blue"
    }
}
```



Demo: qml-composing-uis/ex-elements/colors.qml

Images

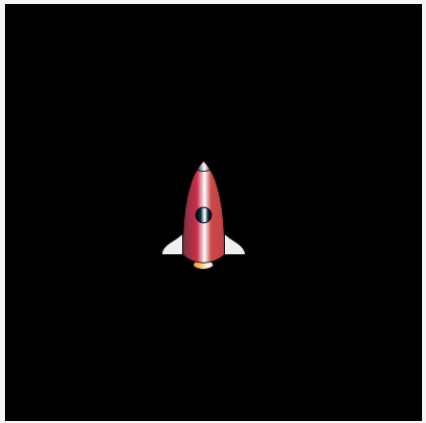
The Qt Company

- Represented by the `Image` element
- Refer to image files with the `source` property
  - Using absolute URLs
  - Or relative to the QML file
- Can be transformed
  - scaled, rotated
  - About an axis or central point

Images

The Qt Company

```
Rectangle {
    width: 400; height: 400
    color: "black”
    Image {
        x: 150; y: 150
        source: "../images/rocket.png"
    }
}
```



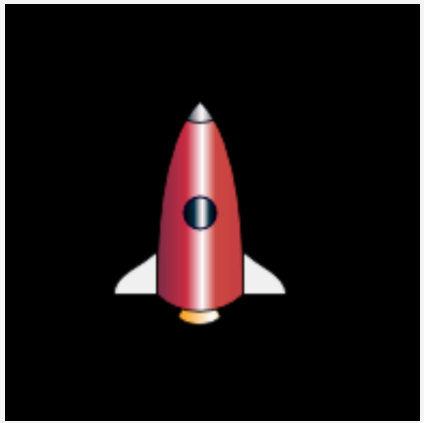
- Property `source` contains a relative path
- Properties `width` and `height` are obtained from the image file

Demo: qml-composing-uis/ex-elements/images.qml

Image Scaling

The Qt Company

```
Rectangle {
    width: 400; height: 400
    color: "black”
    Image {
        x: 150; y: 150
        source: "../images/rocket.png"
    }
}
```

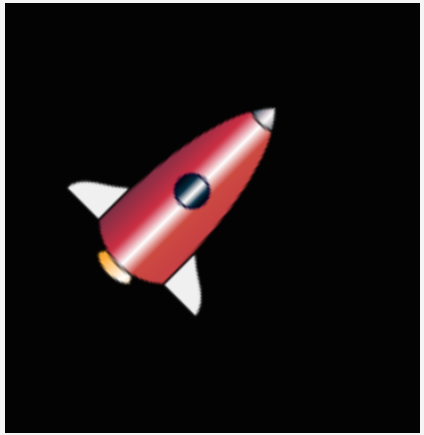


- Property `source` contains a relative path
- Properties `width` and `height` are obtained from the image file

Demo: qml-composing-uis/ex-elements/image-scaling.qml

Image Rotation

```
Rectangle {
    width: 200; height: 200
    color: "black”
    Image {
        x: 50; y: 35
        source: "../images/rocket.png"
        rotation: 45.0
    }
}
```



- Set the rotate property
- By default, the center of the item remains in the same place

Demo: qml-composing-uis/ex-elements/image-rotation.qml

Image Rotation

The Qt Company

```
Rectangle {
    width: 200; height: 200
    color: "black”
    Image {
        x: 50; y: 35
        source: "../images/rocket.png"
        rotation: 45.0
        transformOrigin: Item.Top
    }
}
```

- Set the `transformOrigin` property
- Now the image rotates about the top of the item

Gradients

The Qt Company

Define a gradient using the gradient property:

- With a `Gradient` element as the value
- Containing `GradientStop` elements, each with
  - A position: a number between 0 (startpoint) and 1 (endpoint)
  - A color
- The start and end points
  - Are on the top and bottom edges of the item
  - Cannot be repositioned
- Gradients override color definitions
- Alternative to gradients: A simple background image.

See Documentation: QML Gradient Element

Gradients

The Qt Company

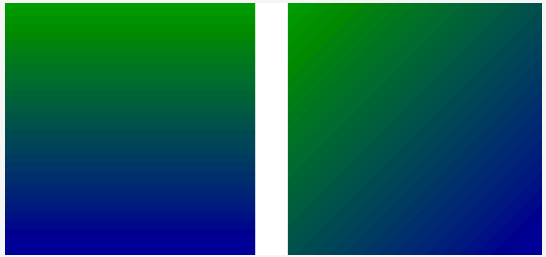
```
Rectangle {
    width: 400; height: 400
    gradient: Gradient {
        GradientStop {
            position: 0.0; color: "green"
        }
        GradientStop {
            position: 1.0; color: "blue"
        }
    }
}
```

- Note the definition of an element as a property value

Demo: qml-composing-uis/ex-elements/gradients.qml

Gradient Images

```
Rectangle {
    width: 425; height: 200
    Image {
        x: 0; y: 0
        source: "../images/vertical-gradient.png"
    }
    Image {
        x: 225; y: 0; source: "../images/diagonal-gradient.png"
    }
}
```



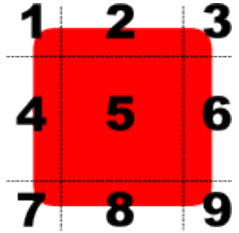
- It is often faster to use images instead of real gradients
- Artists can create the desired gradients

Demo: qml-composing-uis/ex-elements/image-gradients.qml

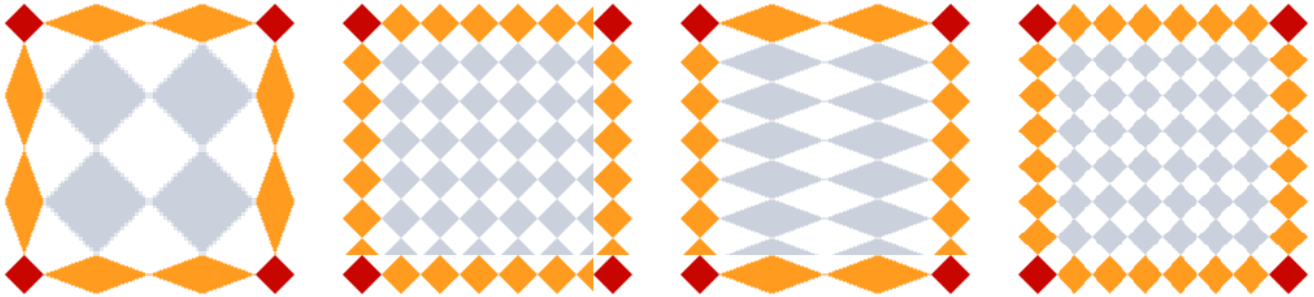
Border Images

The Qt Company

- Create border using part of an image:
  - Corners (region 1,3,7,9) are not scaled
  - Horizontal borders (2 and 8) are scaled according to `horizontalTileMode`
  - Vertical borders (4 and 6) are scaled according to `verticalTileMode`
  - Middle region (5) is scaled according to both modes
- There are 3 different scale modes
  - `Stretch`: scale the image to fit to the available area.
  - `Repeat`: tile the image until there is no more space.
  - `Round`: like `Repeat`, but scales the images down to ensure that the last image is not cropped



Border Images



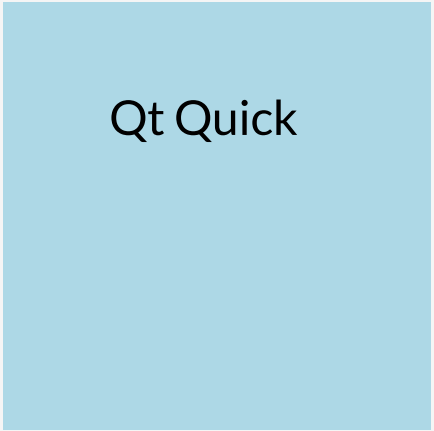
```
BorderImage {
    source: "content/colors.png"
    border { left: 30; top: 30; right: 30; bottom: 30; }
    horizontalTileMode: BorderImage.Stretch
    verticalTileMode: BorderImage.Repeat
    // ...
}
```

Demo: <Qt Examples>/declarative/imageelements/borderimage

Text Elements

Text Elements

```
Rectangle {
    width: 400; height: 400
    color: "lightblue”
    Text {
        x: 100; y: 100
        text: "Qt Quick"
        font.family: "Helvetica”; font.pixelSize: 32
    }
}
```



- Width and height determined by the font metrics and text
- Can also use HTML tags in the text:
  - `"<html><b>Qt Quick</b></html>"`

Demo: qml-composing-uis/ex-elements/text.qml

TextInput

The Qt Company

```
TextInput {
    x: 50; y: 100; width: 300
    text: "Editable text"
    font.family: "Helvetica"; font.pixelSize: 32
}
```

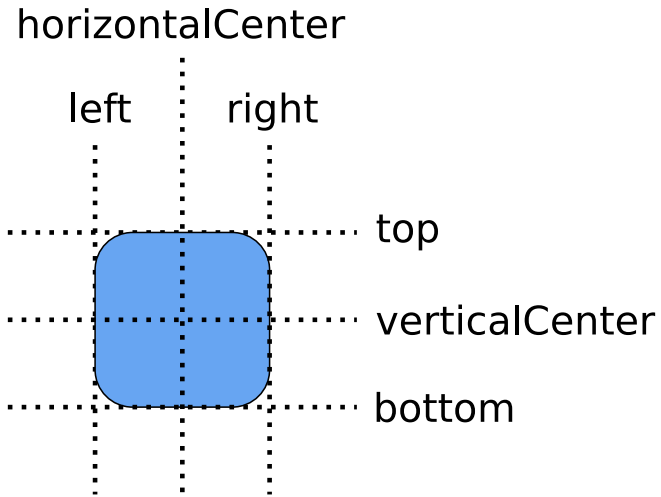
- No decoration (not a QLineEdit widget)
- Gets the focus when clicked
  - Need something to click on
- Property `text` changes as the user types

Demo: qml-composing-uis/ex-elements/textInput.qml

Anchor Layout

Anchors

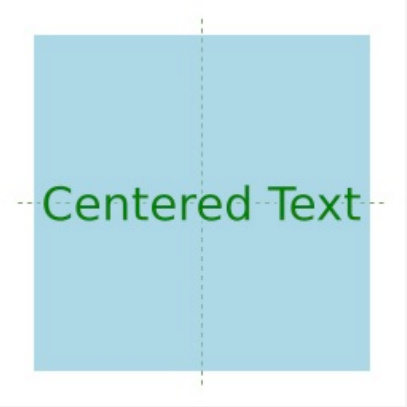
- Used to position and align items
- Line up the edges or central lines of items
- Anchors refer to
  - Other items (`centerIn`, `fill`)
  - Anchors of other items (`left`, `top`)



See Documentation: Anchor Positioning and Anchros

Anchors

```
Rectangle {
    width: 400; height: 400
    color: "lightblue"
    id: rectangle1
    Text {
        text: "Centered text"; color: "green"
        font.family: "Helvetica"; font.pixelSize: 32
        anchors.centerIn: rectangle1
    }
}
```



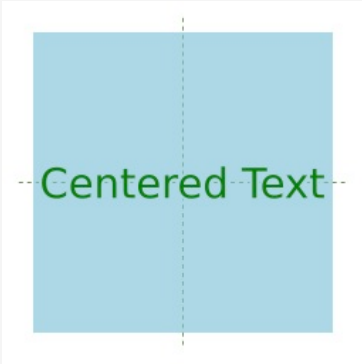
- `anchors.centerIn` centers the `Text` element in the `Rectangle`
  - Refers to an item not an anchor

Demo: qml-composing-uis/ex-anchor-layout/anchors.qml

Anchors

The Qt Company

```
Text {
    text: "Centered text";
    color: "green"
    font.family: "Helvetica";
    font.pixelSize: 32
    anchors.centerIn: parent
    }
}
```

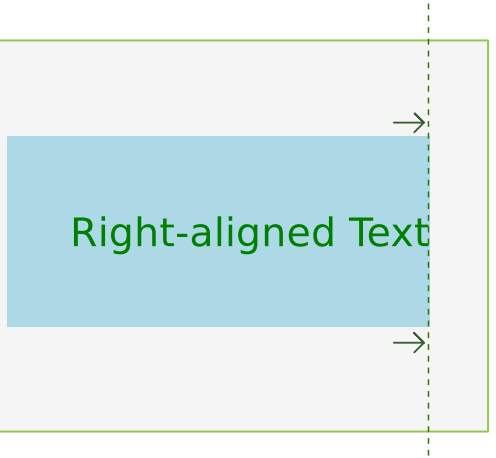


- Each element can refer to its parent element
  - Using the parent ID
- Can refer to ancestors and named children of ancestors

Demo: qml-composing-uis/ex-anchor-layout/anchors2.qml

Anchors

```
Text {
    y: 34
    text: "Right-aligned text"; color: "green"
    font.family: "Helvetica"; font.pixelSize: 32
    anchors.right: parent.right
}
```

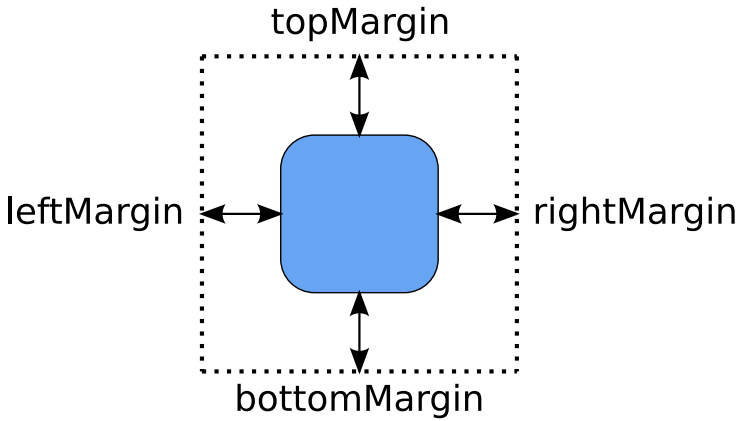


- Connecting anchors together
- Anchors of other items are referred to directly
  - Use `parent.right`
  - Not `parent.anchors.right`

Demo: qml-composing-uis/ex-anchor-layout/anchor-to-anchor.qml

Margins

- Used with anchors to add space
- Specify distances
  - In pixels
  - Between elements connected with anchors



Margins

```
Rectangle {
    width: 400; height: 200; color: "lightblue"
    Image {
        id: book; source: "../images/book.svg"
        anchors.left: parent.left
        anchors.leftMargin: parent.width/16
        anchors.verticalCenter: parent.verticalCenter
    }
    Text {
        text: "Writing"; font.pixelSize: 32
        anchors.left: book.right anchors.leftMargin: 32
        anchors.baseline: book.verticalCenter
    }
}
```

Demo: qml-composing-uis/ex-anchor-layout/alignment.qml

Hints and Tips

The Qt Company

- Anchors can only be used with parent and sibling items
- Anchors work on constraints
  - Some items need to have well-defined positions and sizes
  - Items without default sizes should be anchored to fixed or well-defined Items
- Anchors create dependencies on geometries of other items
  - Creates an order in which geometries are calculated
  - Avoid creating circular dependencies
    - e.g.,parent → child→parent
- Margins are only used if the corresponding anchors are used
  - e.g., `leftMargin` needs `left` to be defined

Strategies for Use

The Qt Company

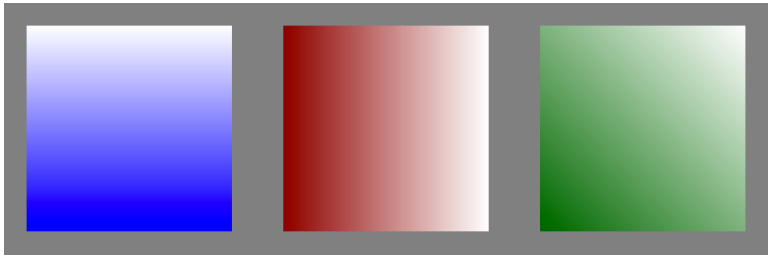
Identify item with different roles in the user interface:

- Fixed items
  - Make sure these have id properties defined
  - Unless these items can easily be referenced as parent items
- Items that dominate the user interface
  - Make sure these have id properties defined
  - Items that react to size changes of the dominant items
  - Give these anchors that refer to the dominator fixed items

Lab – Color and Gradients

The Qt Company

1. How else can you write these colors?
   - `"blue"`
   - `"#ff0000"`
   - `Qt.rgba(0,0.5,0,1)`
2. How would you create these items using the gradient property?



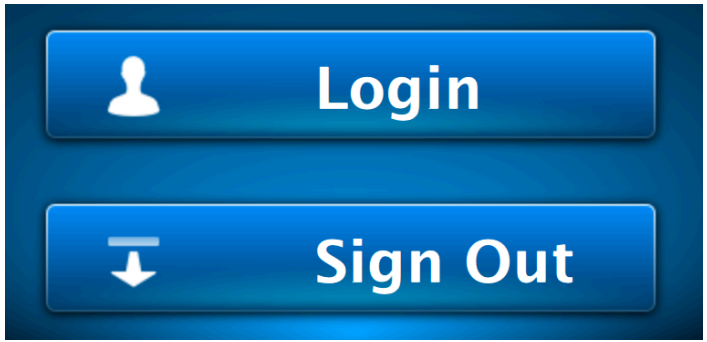
3. Describe another way to create these gradients?

Lab – Images and Text

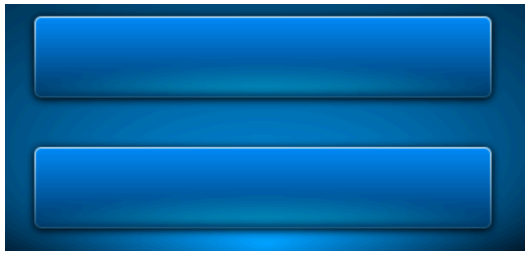
The Qt Company

1. When creating an `Image`, how do you specify the location of the image file?
2. By default, images are rotated about a point inside the image. Where is this point?
3. How do you change the text in a `Text` element?

Lab – Images, Text, and Anchors



- Create a user interface similar to the one shown above.
- Hint: Use the background image supplied in the common images directory.


Lab: qml-composing-uis/lab-text-images-anchors