```
import QtQuick 2.15
import QtQuick.Controls 2.15 as Controls
import QtQuick.Layouts 1.15
import org.kde.kirigami 2.20 as Kirigami
```

essential

```
Kirigami.ApplicationWindow {
    id: root

    title: i18nc("@title:window", "Day Kountdown")

    globalDrawer: Kirigami.GlobalDrawer {
        isMenu: true
        actions: [
            Kirigami.Action {
                text: i18n("Quit")
                icon.name: "gtk-quit"
                shortcut: StandardKey.Quit
                onTriggered: Qt.quit()
            }
        ]
    }
```

Pre-defined Kirigami / QML types appropriate for model. What others are there?

```
    ListModel {
        id: kountdownModel
    }
```

other components will communicate with this through this id

```
    Component {
        id: kountdownDelegate
        Kirigami.AbstractCard {
            contentItem: Item {
```

Not sure if essential, but possibly good practice?

```
                implicitWidth:
                delegateLayout.implicitWidth
                implicitHeight:
                delegateLayout.implicitHeight
                GridLayout {
```

forward reference to this?

One of too layout used by this card. The other is Column, later.

```
                    id: delegateLayout
                    anchors {
                        left: parent.left
```

```
    top: parent.top
    right: parent.right
}
rowSpacing:
Kirigami.Units.largeSpacing
columnSpacing:
Kirigami.Units.largeSpacing
columns: root.wideScreen ? 4 : 2

Kirigami.Heading {
    Layout.fillHeight: true
    level: 1
    text: i18n("%1 days",
    Math.round((date-Date.now())/
    86400000))
}

ColumnLayout {
    Kirigami.Heading {
        Layout.fillWidth: true
        level: 2
        text: name
    }
    Kirigami.Separator {
        Layout.fillWidth: true
        visible:
        description.length > 0
    }
    Controls.Label {
        Layout.fillWidth: true
        wrapMode: Text.WordWrap
        text: description
        visible:
        description.length > 0
    }
}
Controls.Button {
    Layout.alignment:
```

probably a good idea to always have a heading?

```
                    Qt.AlignRight
                    Layout.columnSpan: 2
                    text: i18n("Edit")
                }
            }
        }
    }
}

// Overlay sheets appear over a part of the
window
Kirigami.OverlaySheet {
    id: addSheet
    header: Kirigami.Heading {
        text: i18nc("@title:window", "Add
        kountdown")
    }
    // Form layouts help align and structure a
    layout with several inputs
    Kirigami.FormLayout {
        // Textfields let you input text in a
        thin textbox
        Controls.TextField {
            id: nameField
            // Provides label attached to the
            textfield
            Kirigami.FormData.label:
            i18nc("@label:textbox", "Name:")
            // Placeholder text is visible before
            you enter anything
            placeholderText: i18n("Event name
            (required)")
            // What to do after input is accepted
            (i.e. pressed enter)
            // In this case, it moves the focus
            to the next field
            onAccepted: ← "event handler" for this TextField.
            descriptionField.forceActiveFocus()
```

```
}
Controls.TextField {
    id: descriptionField
    Kirigami.FormData.label:
    i18nc("@label:textbox",
    "Description:")
    placeholderText: i18n("Optional")
    onAccepted:
    dateField.forceActiveFocus()
}
Controls.TextField {
    id: dateField
    Kirigami.FormData.label:
    i18nc("@label:textbox", "Date:")
    placeholderText: i18n("YYYY-MM-DD")
    inputMask: "0000-00-00"
}
Controls.Button {
    id: doneButton
    Layout.fillWidth: true
    text: i18nc("@action:button", "Done")
    // Button is only enabled if the user
    has entered something into the
    nameField
    enabled: nameField.text.length > 0
    onClicked: {
        // Add a listelement to the
        kountdownModel ListModel
        kountdownModel.append({
            name: nameField.text,
            description:
            descriptionField.text,
            date:
            Date.parse(dateField.text)
        });
        nameField.text = ""
        descriptionField.text = ""
        dateField.text = ""
```

Interact with model when this button clicked.

appears to be a newly constructed anonymous object ?

lookup against examples from ListModel API

ids of components mentioned above, so their fields are in scope ?

```
                    addSheet.close();
                }
            }
        }
    }
```

essential to close the overlaySheet.

I think required, for app to know which page to start on

```
    pageStack.initialPage: Kirigami.ScrollablePage {
        title: i18nc("@title", "Kountdown")

        // Kirigami.Action encapsulates a UI action. Inherits from Controls.Action
        actions.main: Kirigami.Action {
            id: addAction
            // Name of icon associated with the action
            icon.name: "list-add"
            // Action text, i18n function returns translated string
            text: i18nc("@action:button", "Add kountdown")
            // What to do when triggering the action
            onTriggered: addSheet.open()
        }
```

specifies type of view on the ScrollablePage.

```
        Kirigami.CardsListView {
            id: layout
            model: kountdownModel
            delegate: kountdownDelegate
        }
    }
}
```

view?

model?

delegate

Assuming these two fields used inside a View type object link it to its model and delegate.