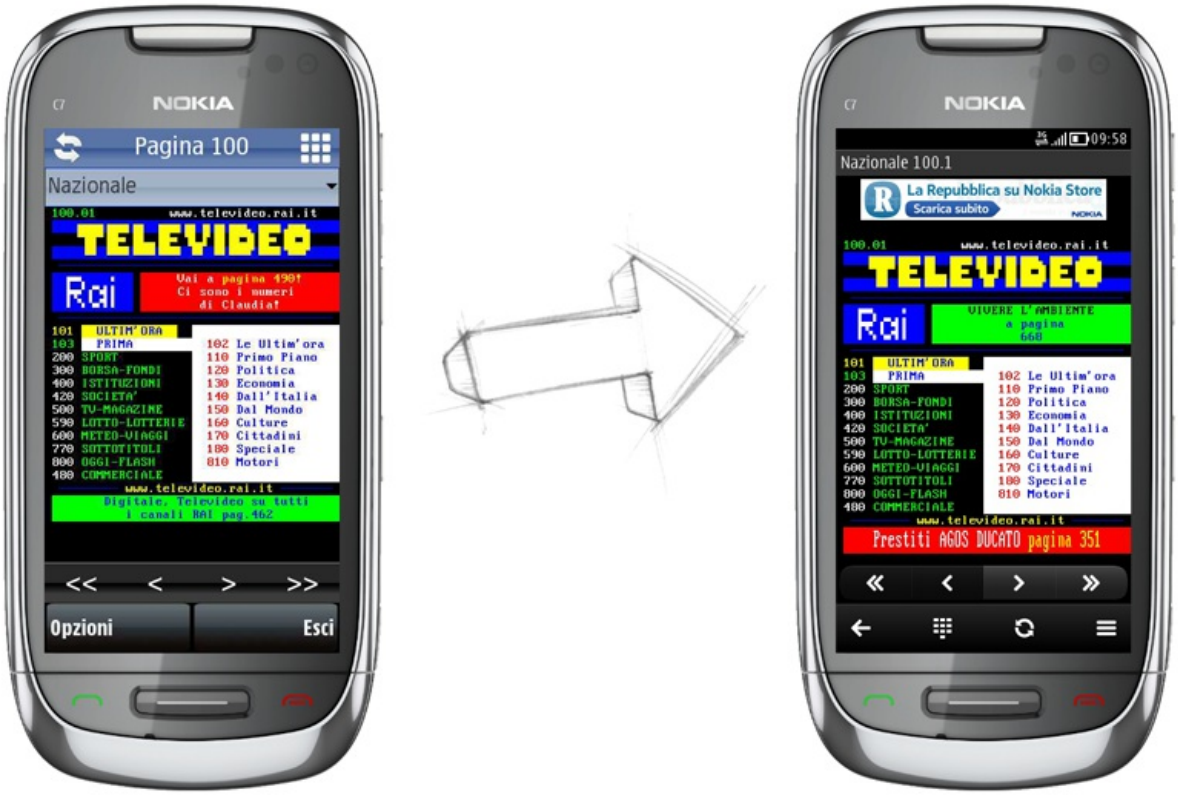
Porting the Italian Teletext app from WRT to Qt Quick

This article explains the main steps and guidelines followed to port the Italian Teletext app from the original WRT version to Qt using Symbian Qt Quick components

Introduction

Teletext Italy is a Web Runtime application that allows users to easily browse the Italian teletext pages. This article will highlight the main design and implementation choices made to port the app to Qt Quick.

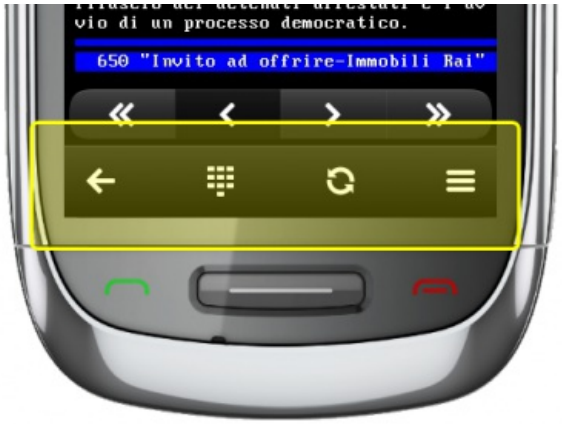


Porting the UI

The Teletext app is mainly a single-page app, presenting all the necessary information in one screen.

The toolbar

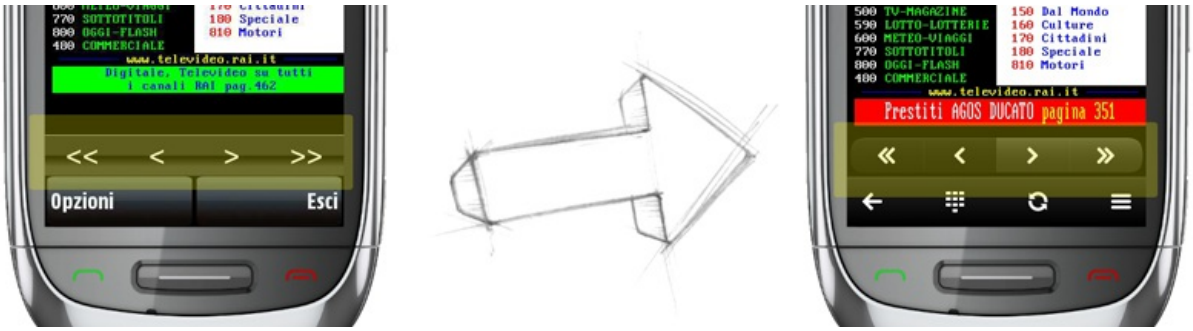
While the WRT version placed buttons on the app header, the Qt Quick UI is redesigned in order to place the more relevant actions' buttons in the page's ToolBar. Following the Symbian Design Guidelines, the left button is used to exit the app, and the right one to open a menu with extra options. The center buttons are used to allow the user to manually enter a page number or to reload the currently displayed page.



Navigation buttons

The WRT version implemented a buttom toolbar, placed just above the softkeys, with four buttons that allow user to navigate

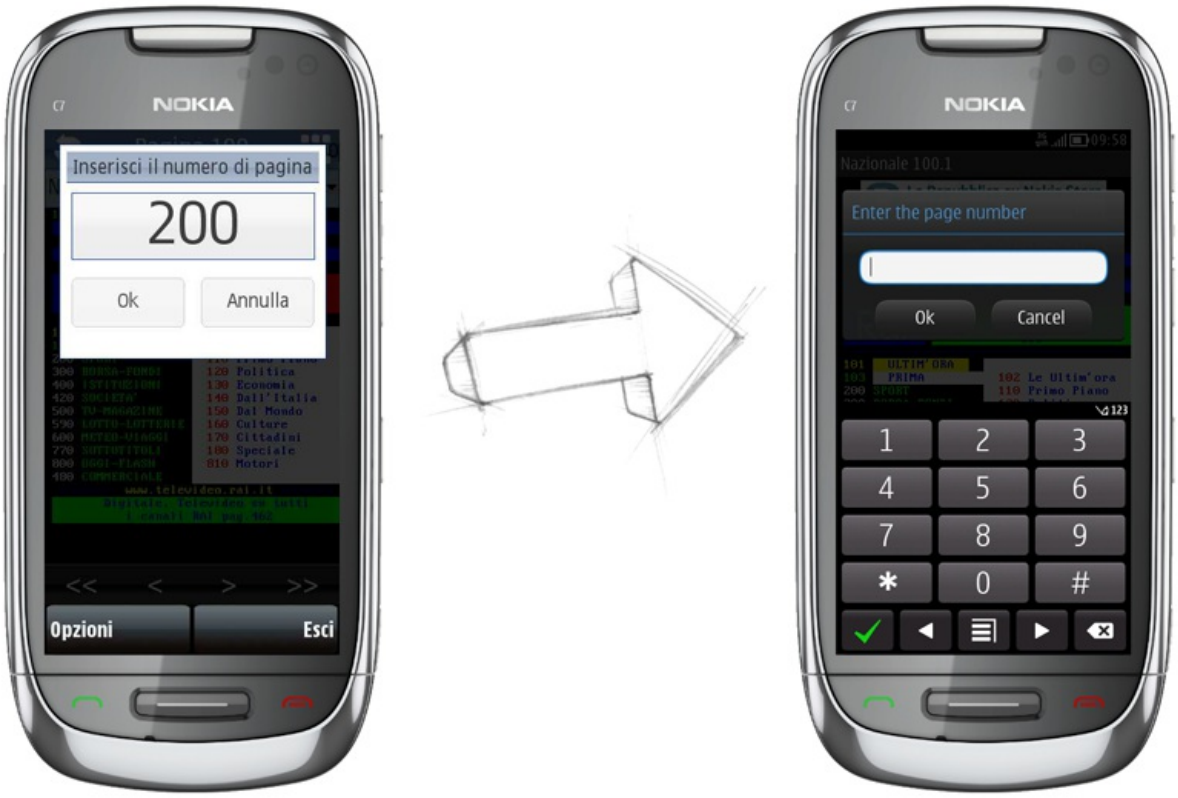
pages and subpages. This custom component is easily ported to Qt Quick by using the ButtonRow component together with four Buttons.




```
ButtonRow {
    exclusive: false
    Button {
        iconSource: "images/page_prev.png"
        onClicked: switchPage(-1)
    }
    Button {
        iconSource: "images/subpage_prev.png"
        onClicked: switchSubpage(-1)
    }
    Button {
        iconSource: "images/subpage_next.png"
        onClicked: switchSubpage(1)
    }
    Button {
        iconSource: "images/page_next.png"
        onClicked: switchPage(1)
    }
}
```

Page selection dialog

While the WRT app implemented the page selection dialog with a custom view, Qt Quick components offer the ready-to-use CommonDialog component that can be used to build dialogs with platform-style title area and custom content.



A TextField is used to allow the user to enter the page number. To improve user experience, the **inputMethodHints: Qt.ImhDigitsOnly** attribute is used, to present the user with the numeric keyboard, so that the number input is faster and easier.

```
TextField {
    id: pageNumberField
    anchors.centerIn: parent
    maximumLength: 3
    inputMethodHints: Qt.ImhDigitsOnly
    inputMask: "000"
}
```

A ToolButton is placed in the page ToolBar to allow the user to open the page number dialog.

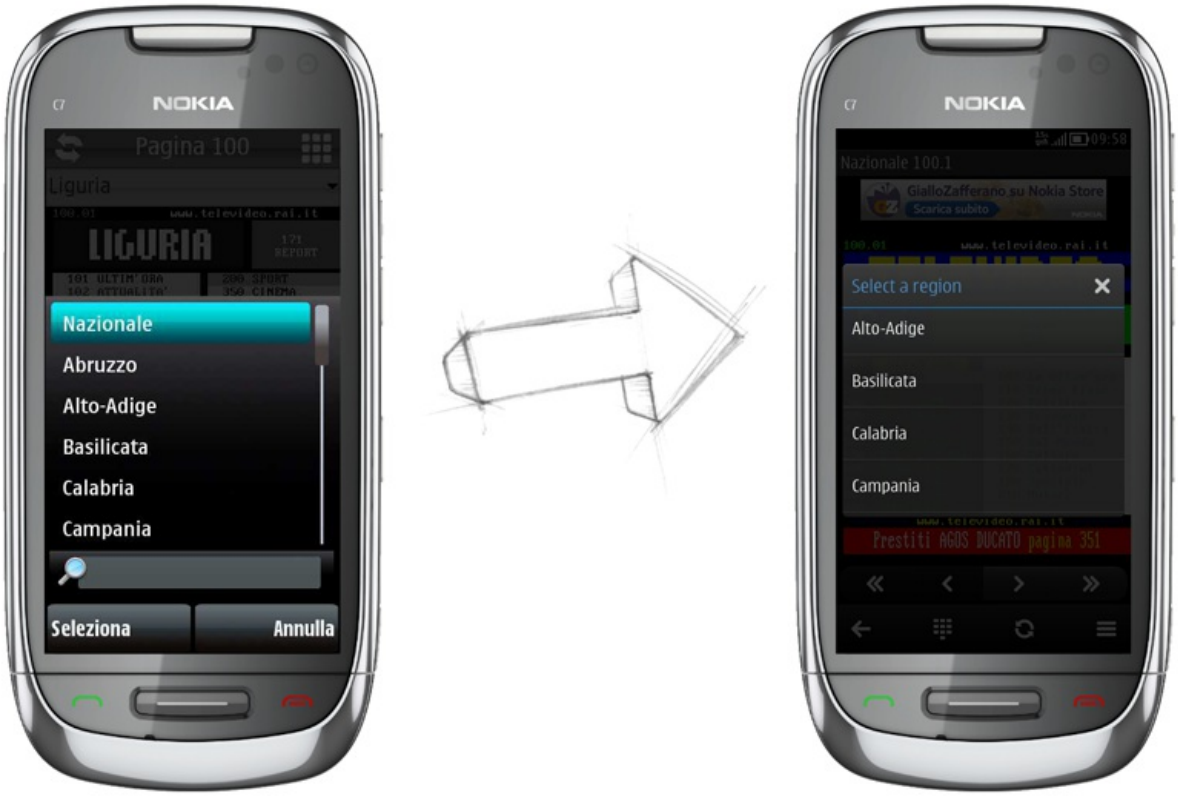
```
ToolButton {
            flat: true
            iconSource: "toolbar-dialer"
            onClicked: numberKeypadClicked()
        }
```

The JavaScript **numerKeypadClicked()** opens the dialog, focuses the text field, and forces the virtual keyboard to appear, so that the user must not manually touch the text field to activate that. All these operations are implemented with the following code, and area geared at improving the overall user experience.

```
function numberKeypadClicked()
{
    pageNumberDialog.open();

    pageNumberField.forceActiveFocus();

    pageNumberField.openSoftwareInputPanel();

}
```

Region selection

The WRT version used the HTML <SELECT> element to implement the region selection dialog, allowing the user to pick one of the available regional teletext editions.

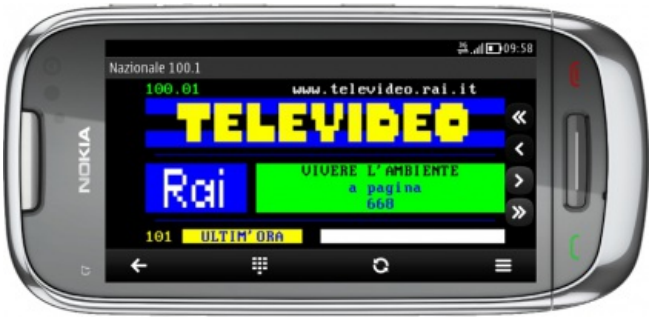


Porting this element to Qt Quick is straightforward thanks to the SelectionDialog component, that is used as shown below.

```
SelectionDialog {
    id: regionDialog
    titleText: qsTr("Select a region")
    model: regionModel
    onAccepted: {
        _region = _regionIds[selectedIndex]
        loadHomePage()
    }
}
```

Landscape support

Following the Symbian Design Guidelines, the app supports landscape mode, nicely repositioning its UI elements in order to provide an optimal user experience.



Since vertical space is smaller in landscape mode, the following UI changes are performed:

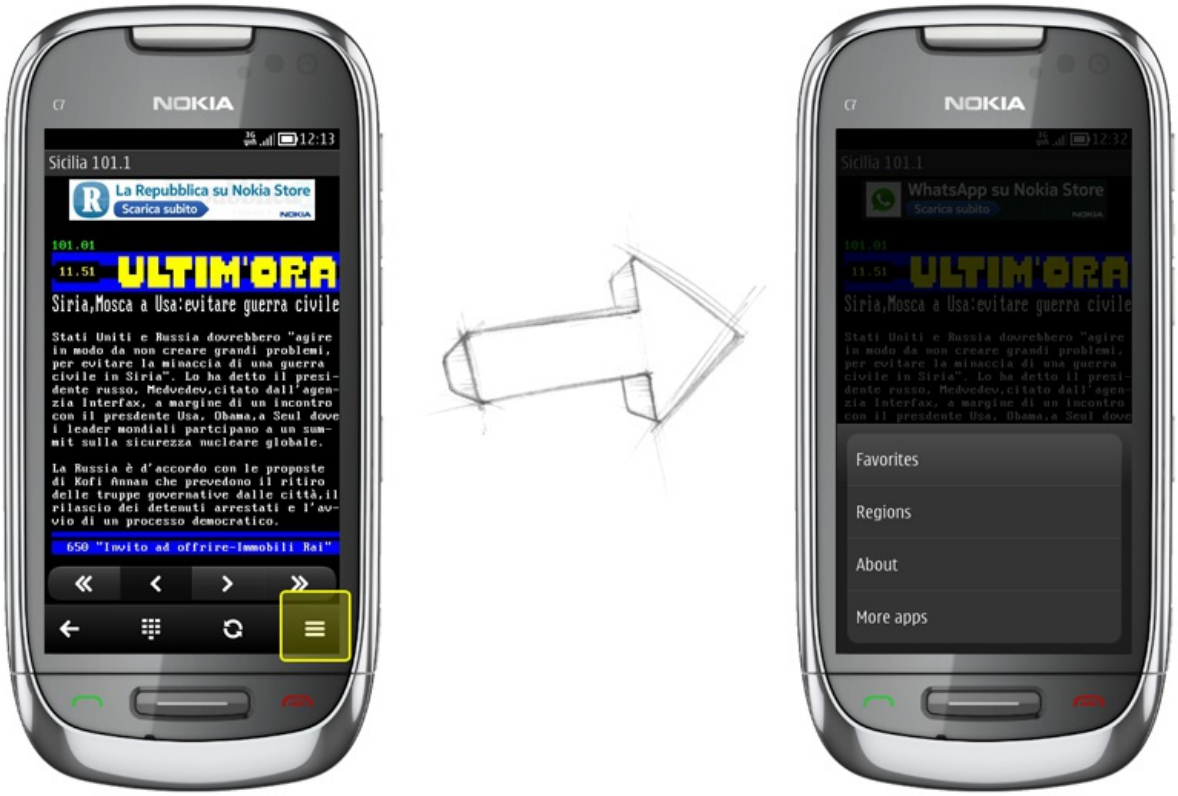
- the ad unit is hidden, since it would take up too much space, leaving little room for the relevant UI elements (mainly, the teletext page itself)
- for the page-navigation Button s, a ButtonColumn is used in place of the ButtonRow used in portrait mode. The

ButtonColumn is placed on the right side of the screen, so that the maximum available space is left free for the teletext page

- the teletext page image is zoomed up, allowing the user to scroll it up and down

From softkeys to toolbar menu

The original WRT version used softkeys to present the user with some additional options. Since softkeys are removed in the Nokia Belle user interface paradigm, those options are then moved to the menu that is opened via the toolbar "view menu" button, placed on the right of the toolbar.



App Icon

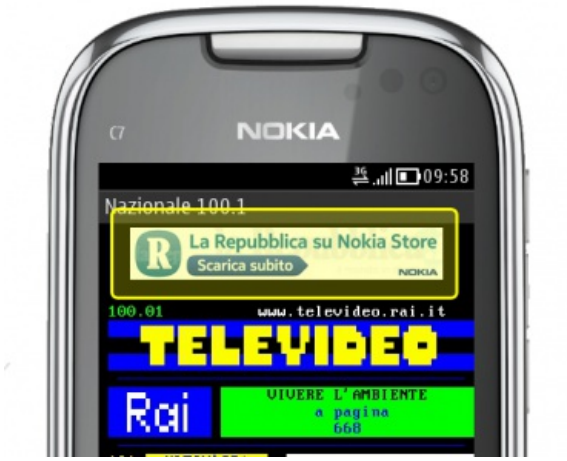
The app icon is redesigned to match the new Nokia Belle style, as visible below.

Adding new features

Thanks to the usage of Qt, many new useful features can be implemented in the new version. This section shows how some of those new features are implemented.

In-App Advertising

Thanks to the recently released InnerActive's QML SDK, integrating In-App Advertising in the app is quick and easy.

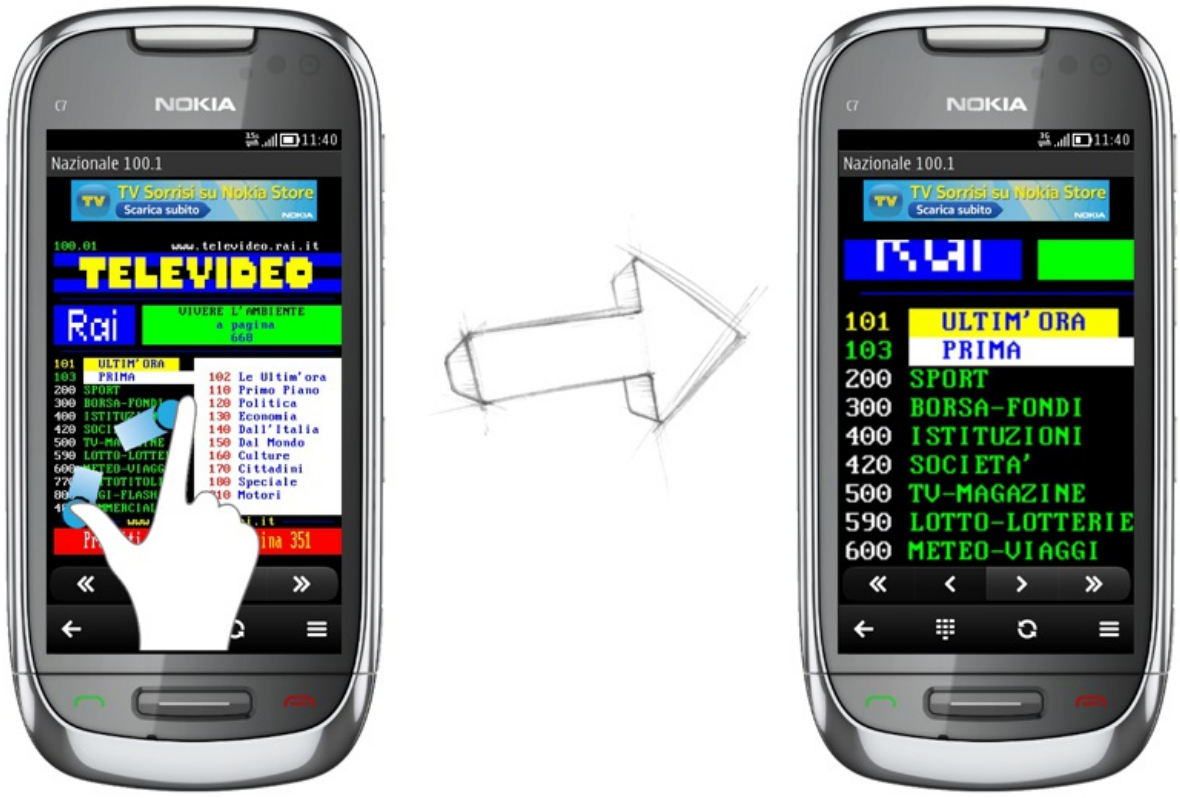


To show ads within the app, it is enough to use and place a **QML AdItem** element, as shown below, specifying the App ID provided by InnerActive.

```
AdItem {
    id: adItem
    parameters: AdParameters {
    applicationId: "My_Application_Id"
    }

    anchors.horizontalCenter: parent.horizontalCenter
}
```

Pinch-zooming

Qt Quick allows to easily implement the pinch-to-zoom feature by using the PinchArea element together with the Flickable element, so that the user can freely navigate the zoomed-up page. The main implementation code is shown below.

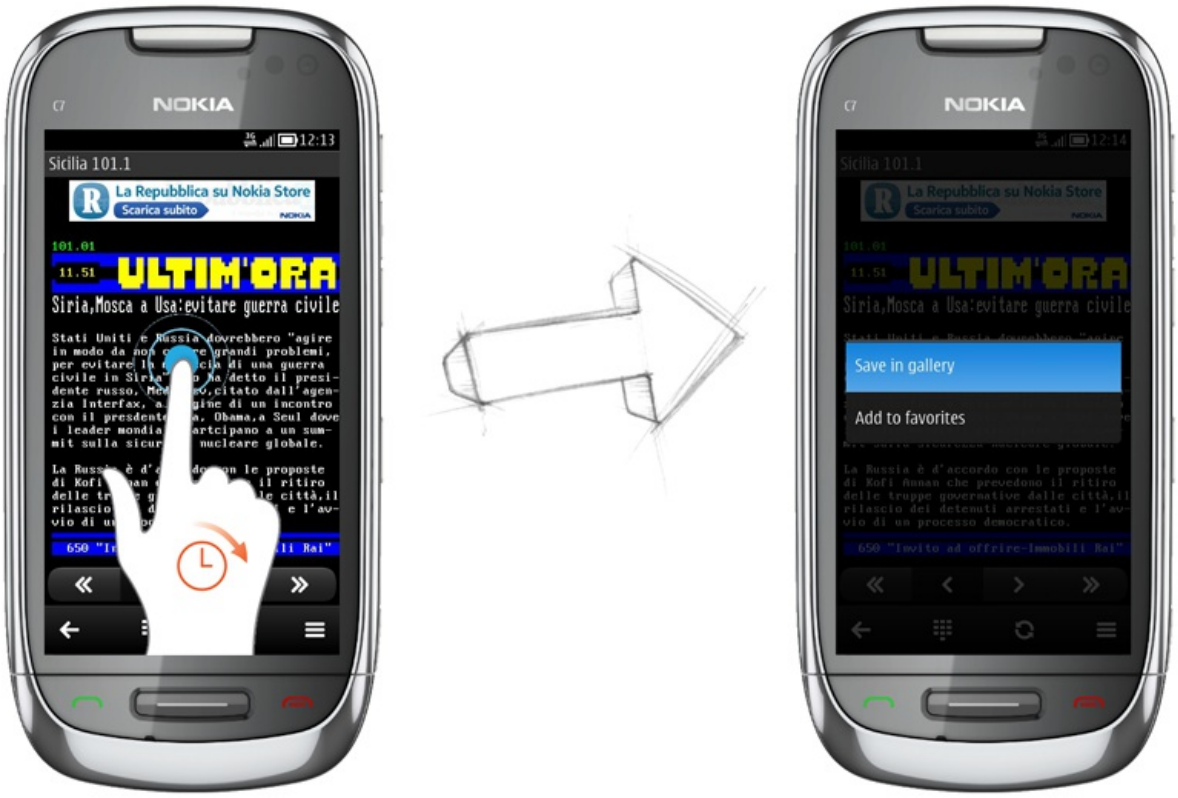



```
Flickable {
    id: flick
    anchors.fill: parent
    contentWidth: width
```

```
    contentHeight: height

    PinchArea {
        width: Math.max(flick.contentWidth, flick.width)
        height: Math.max(flick.contentHeight, flick.height)

        property real initialWidth
        property real initialHeight
        onPinchStarted: {
            initialWidth = flick.contentWidth
            initialHeight = flick.contentHeight
        }

        onPinchUpdated: {
            flick.contentX += pinch.previousCenter.x - pinch.center.x
            flick.contentY += pinch.previousCenter.y - pinch.center.y
            flick.resizeContent(Math.max(flick.width, initialWidth * pinch.scale),
Math.max(flick.height, initialHeight * pinch.scale), pinch.center)
        }

        onPinchFinished: {
            flick.returnToBounds()
        }
    }
    Item {
        id: pageImageContainer
        width: flick.contentWidth
        height: flick.contentHeight
    }
}
```

Save teletext page

The user is given the possibility to save a teletext page to the device gallery.



The UI uses the ContextMenu Qt Quick component to show a menu just above the teletext page.

```
ContextMenu {
    id: pageContextMenu

    MenuLayout {
        MenuItem {
            text: qsTr("Save in gallery")
            onClicked: savePageImage()
        }
        [ ... ]
    }
}
```

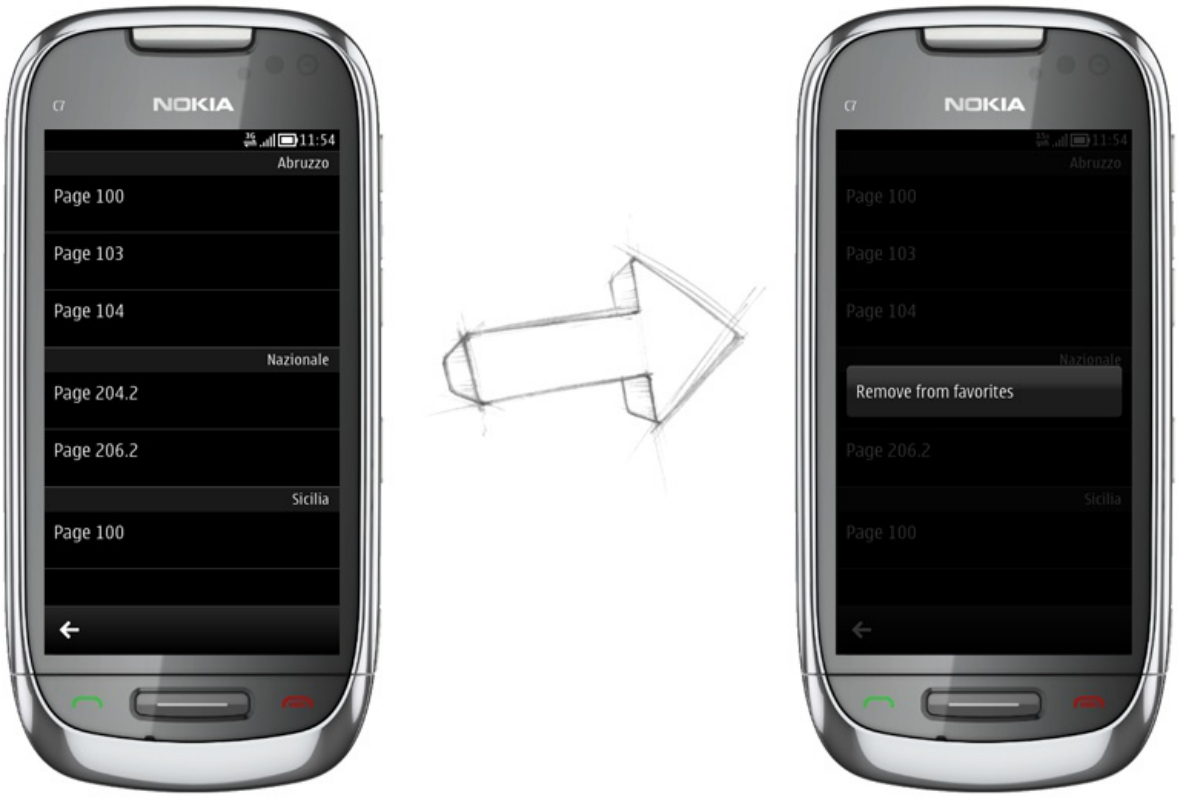
The context menu is opened when the user long-presses the teletext page, feature implemented with the code snippet below.

```
Item {
    id: pageImageContainer
    width: flick.contentWidth
    height: flick.contentHeight

    [...]

    MouseArea {
        anchors.fill: parent
        onPressAndHold: pageContextMenu.open()
    }
}
```

Favorites management

The app allows the user to save and manage his favorite teletext pages, so that he can easily browse those. While the core functionality is implemented using the SQLite support provided by Qt Quick, the UI uses a second Page element used to display

the favorites section.



This page is implemented by using the ListItem and ListHeading Qt Quick components to populate a standard QML ListView element. Following the Symbian Design Guidelines, the page header scroll together with the content, so leaving up the maximum available space for the content.

Long-pressing an item in the favorites list shows a ContextMenu that allows the user to remove a page from the favorites.

Download

A self-signed version of the Qt Quick porting can be downloaded here: TeletextItaly.sis.

Summary

The benefits of porting the Teletext app from WRT to Qt Quick range from UI to functionality, and include:

- a UI style more coherent and integrated with the device UI
- access to more functionality, that allow to build and integrate new features in the app
- a smoother, more responsive user interface
- a fast and easy porting process

Note: This is an entry in the Symbian Qt Quick Components Competition 2012Q1