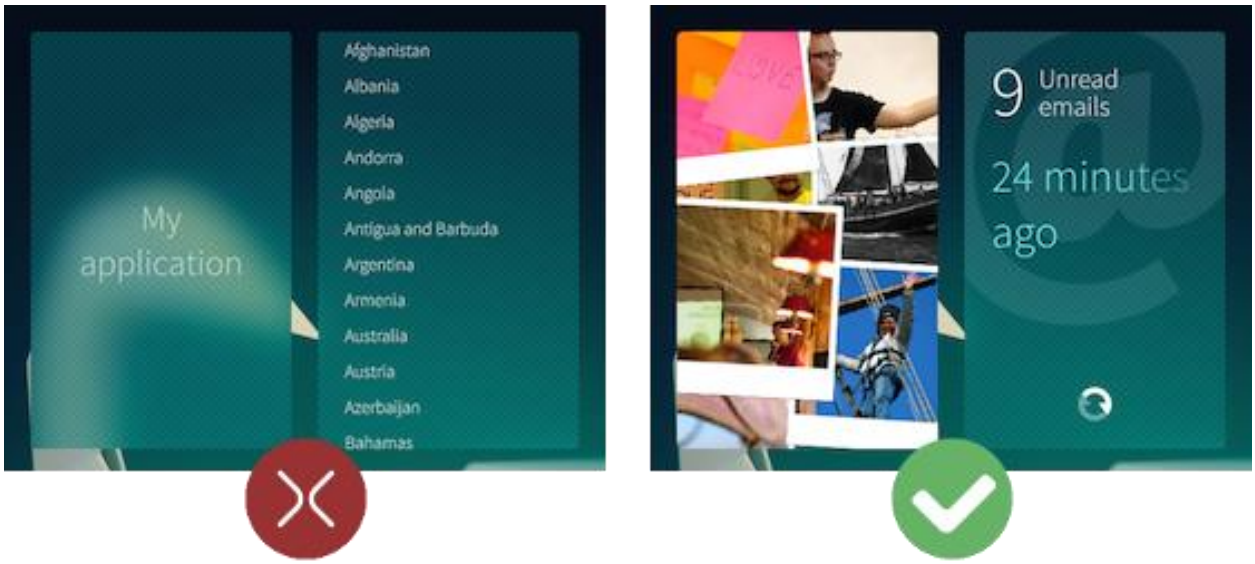
Общие проблемы при разработке приложений для операционной системы Sailfish

Оглавление

Отсутствие полезных активных обложек

Обложки активных приложений (активные обложки) с функцией CoverAction лежат в основе многозадачности операционной системы Sailfish. Однако при адаптации приложений под операционную систему Sailfish концепция активных обложек может показаться разработчикам незначительной, потому что такие системы, как Android, iOS и оригинальные приложения N9 Meego не имеют подобной функциональности.

С помощью активных обложек можно не только легко переключаться между приложениями, но и просматривать состояния экранов или совершать действия в открытых приложениях, не посещая их. Активная обложка должна предоставлять пользователю основную информацию о приложении, показывать рабочие задачи и обеспечивать быстрый доступ к основным действиям, например, сохранению нового элемента или поиску элементов в списке, когда приложение свернуто на домашний экран.



Нестандартные цвета надписей

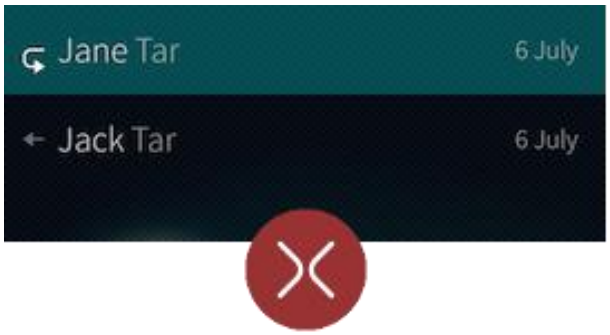
В операционной системе Sailfish для обозначения интерактивности элемента интерфейса используют цвета стилей (атмосфер) платформы. Существуют исключения, но в большинстве случаев кнопки, переключатели, пункты списков и другие элементы, которые должны реагировать на действия пользователя, окрашивают в основные цвета выбранной атмосферы. Для элементов, которые используются для описания интерфейса, например, статические надписи, заголовки страниц или разделов применяют фоновые цвета (подсветку) атмосферы.

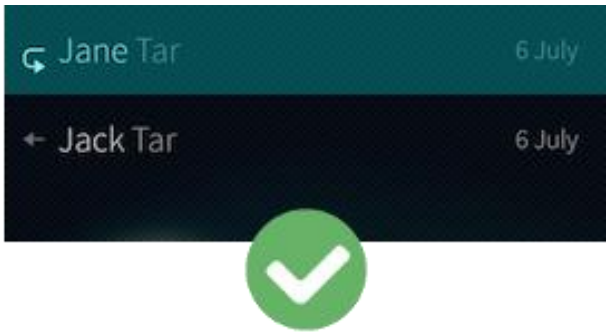
В примере ниже текст интерфейса — это не интерактивный элемент, поэтому для него используют фоновую подсветку Theme.highlightColor.

```
Dialog {
    Label {
        color: Theme.highlightColor
```

```
        text: "Terms of Use. By selecting Accept you agree
to..."
        width: parent.width
    }
}
```

Надписи в элементах интерфейса и иконки при нажатии должны окрашиваться в фоновые цвета атмосферы. Слева на снимке экрана первый пункт списка окрашен в основной цвет атмосферы, хотя пункт уже выбран. Такое поведение не соответствует стилю платформы Sailfish. На правом снимке экрана выбранный пункт подсвечен правильно.





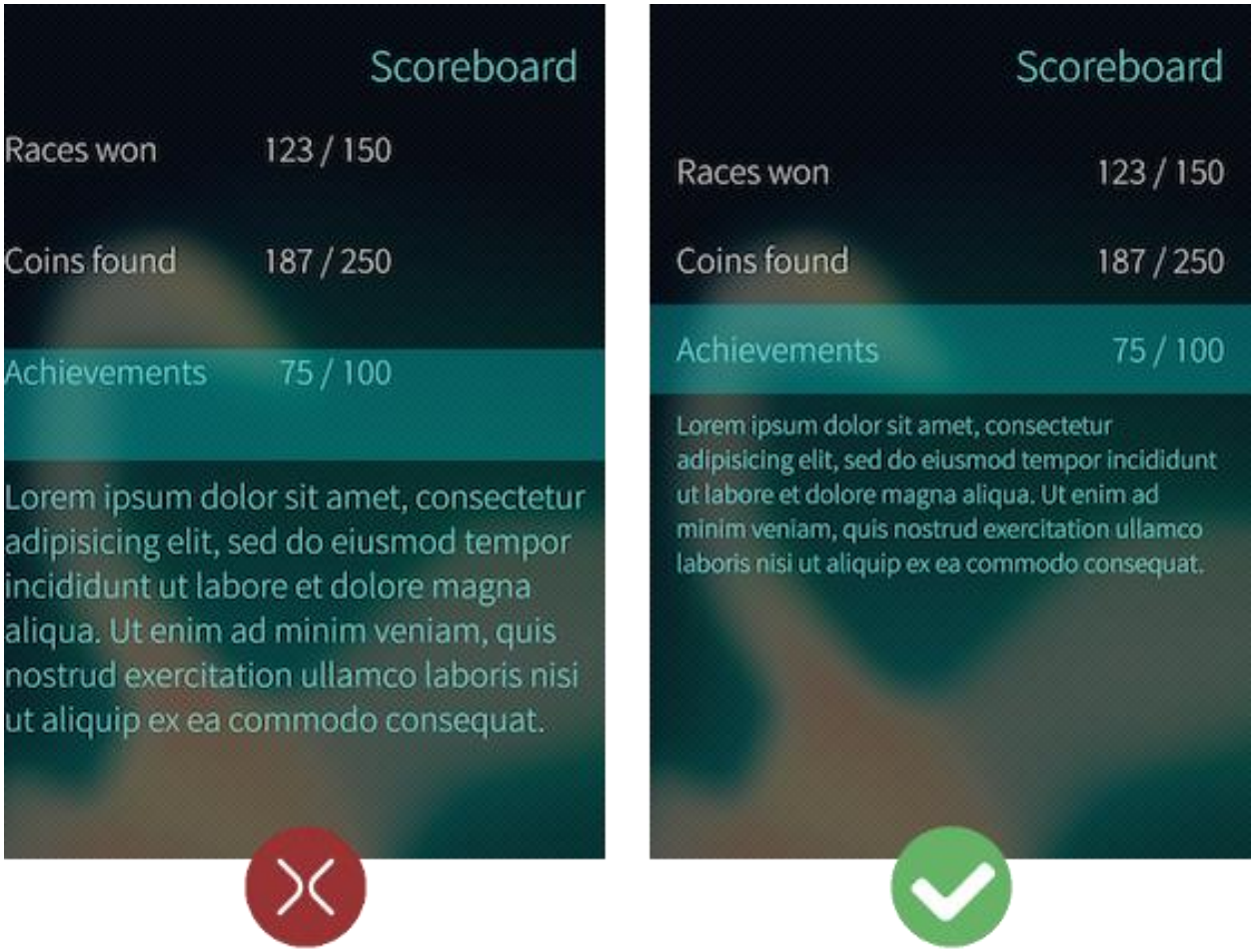
В нативных компонентах Silica со встроенным текстом, таких как кнопка, комбинированный список и контекстное меню при нажатии текст окрашивается автоматически. Для пользовательских компонентов эффект автоматической подсветки текста необходимо реализовывать дополнительно. Например:

```
ListItem {
    id: listItem
    width: parent.width

    Label {
        text: model.text
        color: listItem.highlighted ? Theme.highlightColor :
        Theme.primaryColor
    }
}
```

Рекомендации по стилевым эффектам при нажатии см. в разделе Темы.

Неправильное выравнивание, размеры или интервалы



Элементы интерфейса и их содержимое должны гармонично размещаться на дисплее, для этого размер элементов должен соответствовать стандартам операционной системы Sailfish. Чтобы улучшить визуальное восприятие интерфейса и повысить читаемость, к элементам интерфейса применяют консистентные правила для выравнивания, размеров и межстрочных интервалов.

В приложениях Sailfish графика и изображения обычно выравнивают по краям страницы (например, отображение в галерее или альбомы в Media). Текст и иконки в элементах управления обычно отделяют от края страницы с помощью отступов, заданных через Theme.horizontalPageMargin (слева и справа) и Theme.paddingLarge (сверху и снизу).

Некоторые текстовые контролы, например, комбинированные списки, текстовые поля, заголовки страницы для определения горизонтальных отступов своего содержимого используют значения Theme.horizontalPageMargin по умолчанию. Эти отступы могут быть изменены с помощью значений в свойствах `leftMargin` и `rightMargin` (например, изменение отступа слева (leftMargin) для заголовка страницы (PageHeader)). Для создания собственных пользовательских элементов, необходимо добавить отступы самостоятельно. Например, текстовый элемент Label, описанный ниже, привязывается к полной ширине родительского элемента, а затем принимает свойства `anchors.leftMargin` и `anchors.rightMargin`, если это необходимо:

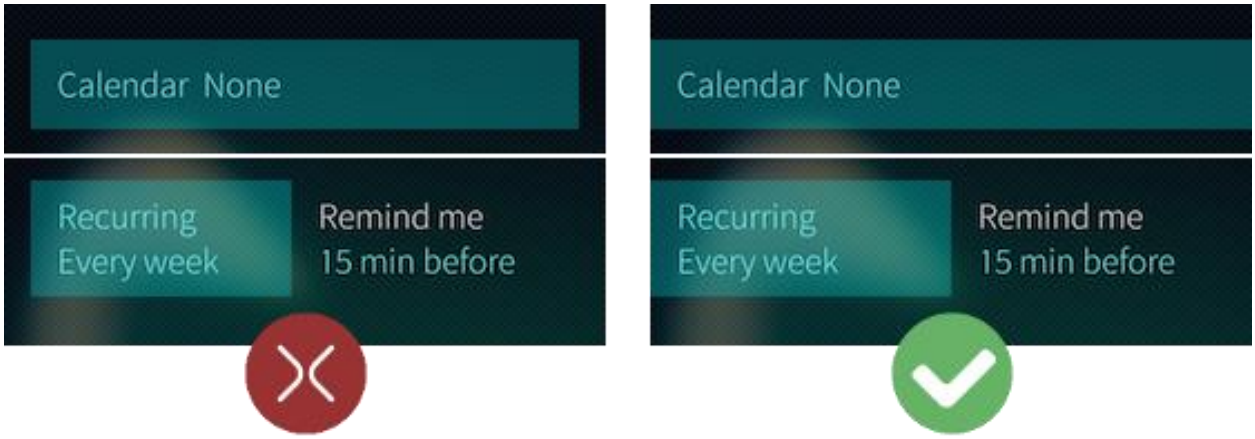
```
Page {
    Label {

        text: "A very, very, very long sentence that will extend
beyond the width of the screen."
        truncationMode: TruncationMode.Fade
        color: Theme.highlightColor

        anchors {
            left: parent.left
            leftMargin: Theme.horizontalPageMargin
            right: parent.right
            rightMargin: Theme.horizontalPageMargin
            verticalCenter: parent.verticalCenter
        }
    }
}
```

Размеры элементов пользовательского интерфейса, такие как размеры шрифтов и пункты списка, должны быть одинаковыми во всем приложении и соответствовать стандартам стилей операционной системы Sailfish. Задать размеры можно с помощью Темы, которая предоставляет набор стандартных параметров, отступов и цветов для стилизации содержимого пользовательского интерфейса Sailfish. Подробная информация и руководство описаны в документации к Theme.

Мелкие сенсорные области



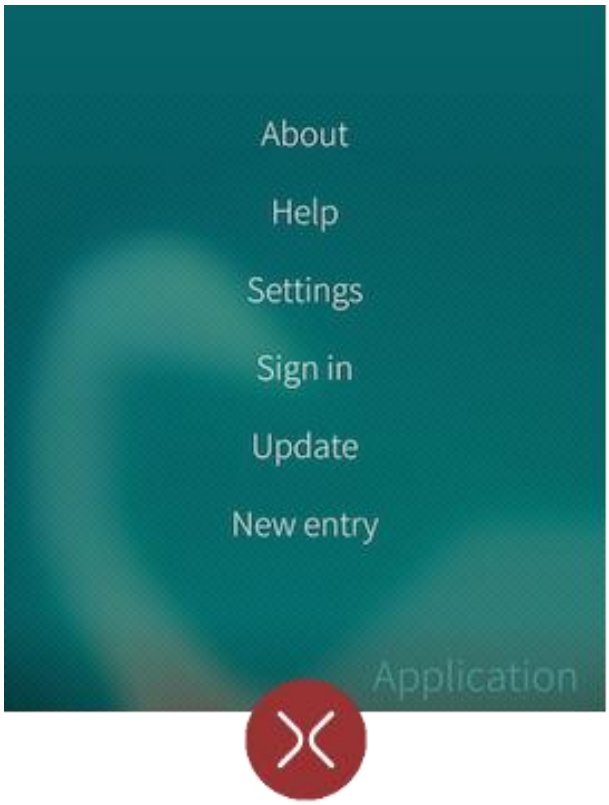
Некоторые разработчики создают приложения с мелкими интерактивными элементами пользовательского интерфейса. Работа с такими элементами вызывает проблемы у пользователей, так как успешное нажатие на элемент требует большей концентрации и точности попадания. Проблема может усугубиться тем, что большинство компонентов интерфейса Sailfish не имеют визуальных границ интерактивной области, поэтому легко ошибиться в размерах сенсорных областей.

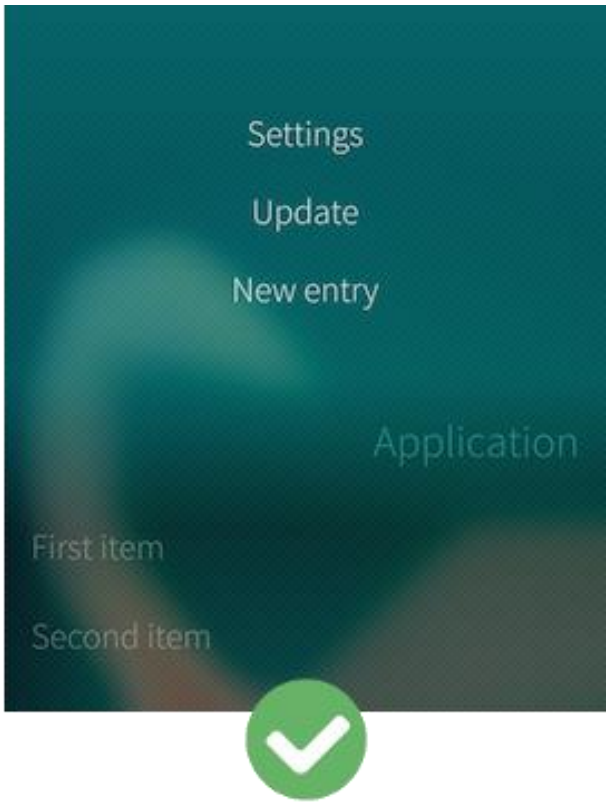
Чтобы максимально увеличить сенсорные области элементов, изменяют размер свойства MouseArea. Если элемент размещен с помощью типа позиционирования Row или Column,

необходимо перераспределить незадействованные пиксели в интервалах между соседними элементами внутрь области касания элемента. Сенсорные элементы не должны быть меньше высоты, указанной в свойствах Theme.itemSizeSmall. Предпочтительно, чтобы высота большинства сенсорных элементов была равна значению Theme.itemSizeMedium или больше, в зависимости от их сложности.

Избыточные пункты в меню Pulley

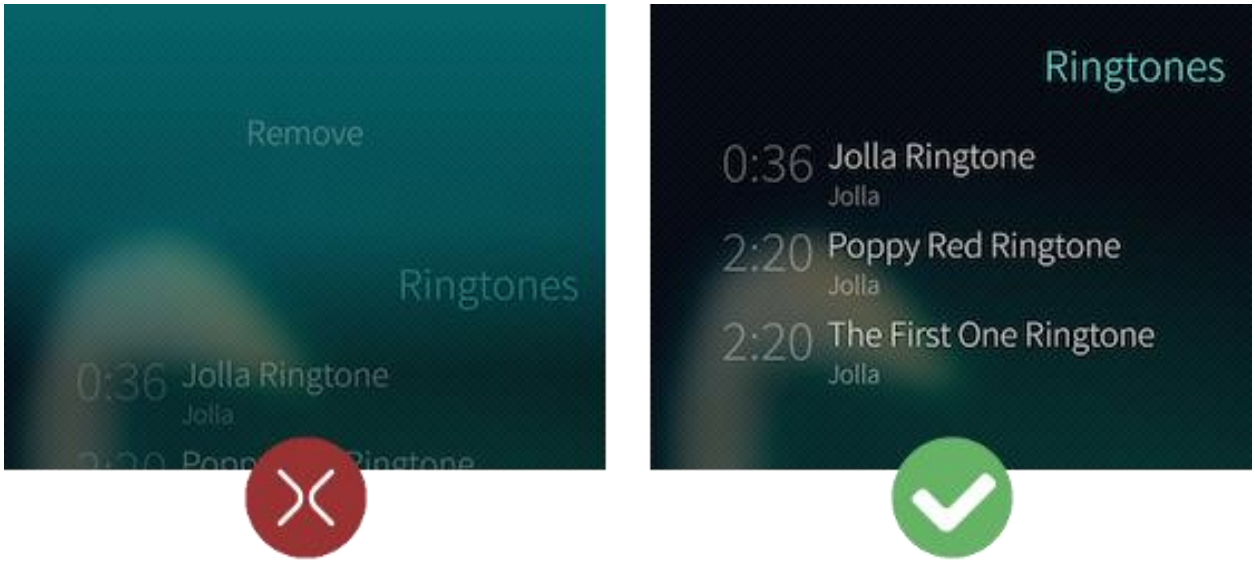
Меню Pulley должно иметь минимальное количество пунктов. Здесь действуют те же правила, что и для традиционной панели инструментов мобильного приложения: меню не должно состоять из 6 и более пунктов.





Большинство приложений платформы Sailfish предоставляют от одного до трех возможных действий в меню Pulley на странице. Чтобы не перегружать меню и облегчить доступ к пунктам, в PullDownMenu или PushUpMenu не показывают более четырех действий.

Отображение неактивного меню Pulley



Некоторые пункты меню Pulley становятся доступными только после выполнения определенных условий. Меню Pulley не показывают, если все его пункты неактивны и их нельзя использовать.

```
PullDownMenu {
    MenuItem { text: "Remove" }

    visible: playList.selectionCount > 0
}
```

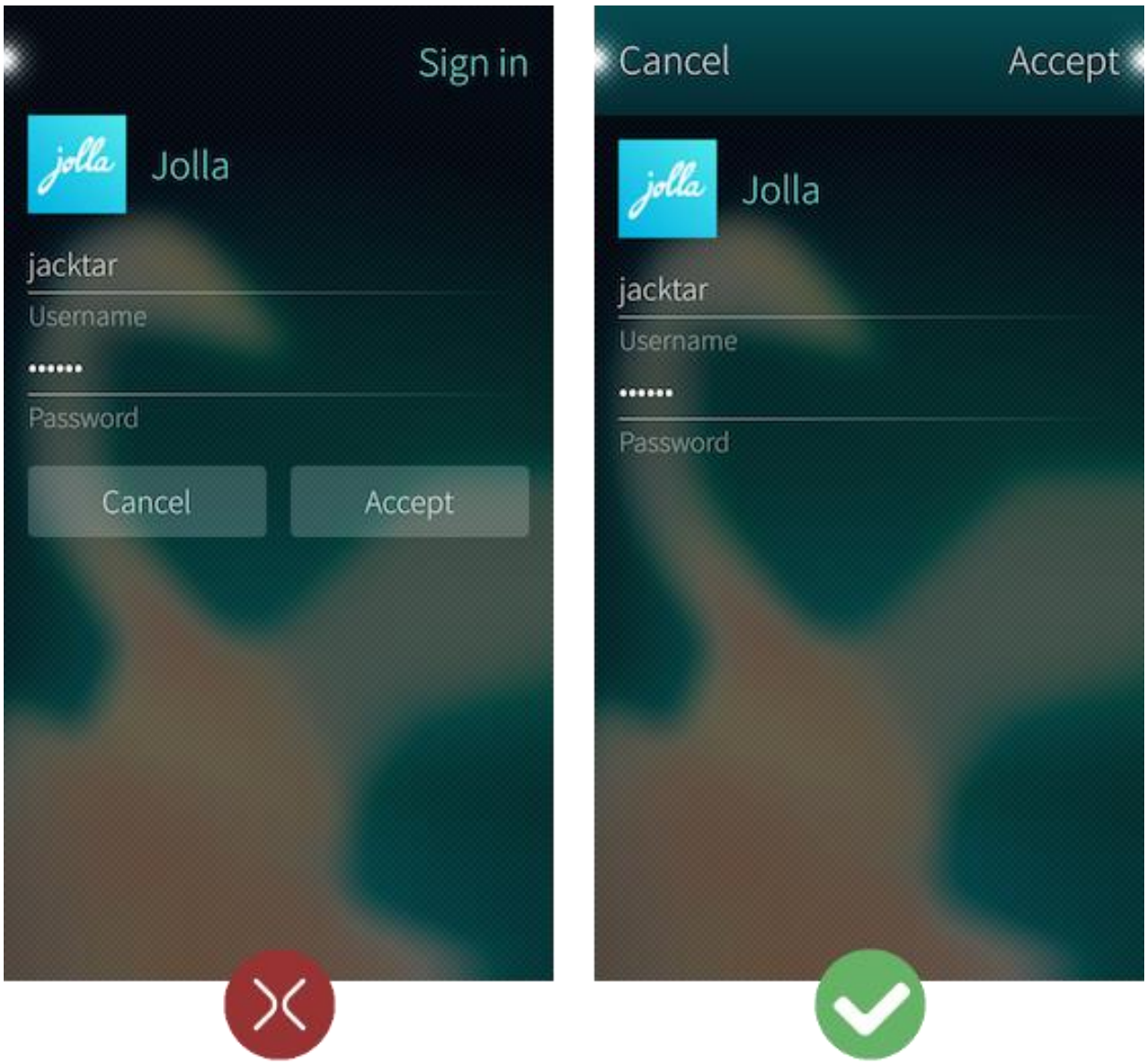
Отсутствие значка прокрутки

Любая страница, которая может накапливать контент, выходящий за пределы экрана, должна иметь значок прокрутки. Он показывает текущее положение просмотра относительно остального доступного содержимого вне экрана.

```
SilicaListView {
    anchors.fill: parent
    VerticalScrollDecorator {}
}
```

Использование кнопок вместо жестов платформы

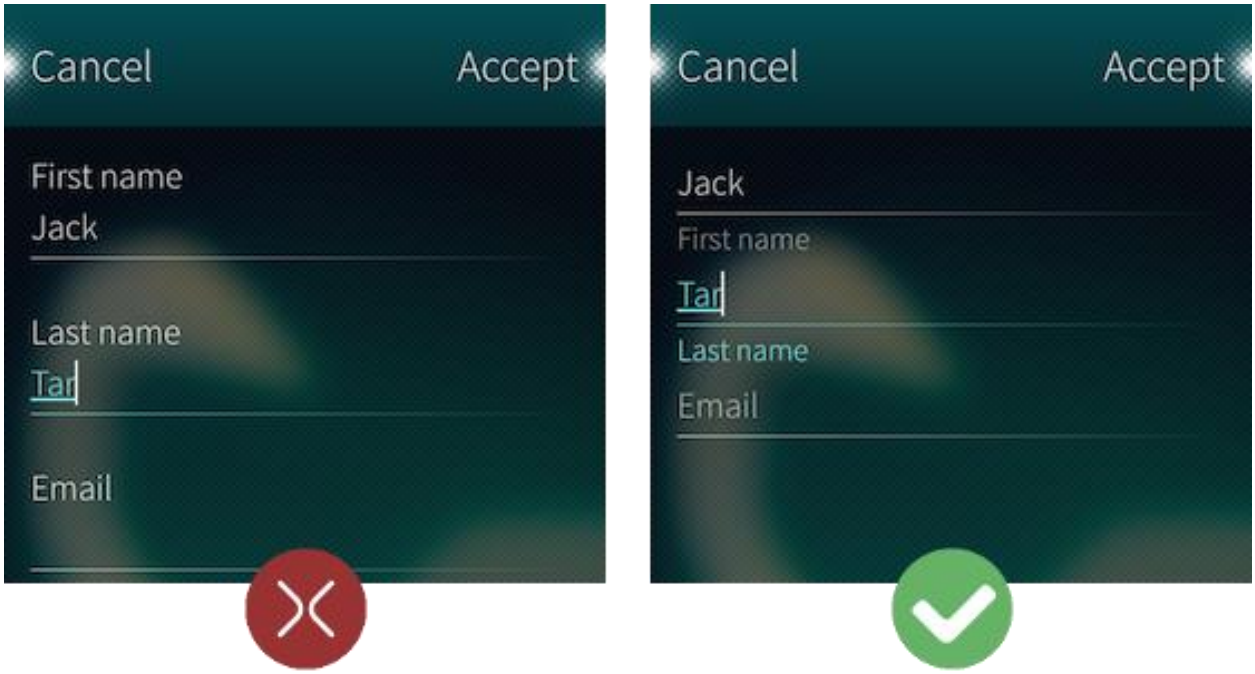
В Sailfish управление устройством основано на жестах. Жесты «применить» и «отменить» заменяют традиционные кнопки подтверждения или отмены. Жест «назад» используют вместо кнопки «назад». Кнопки «выход» и «домой» заменяют жестом, при котором нужно потянуть от левого или правого края экрана к центру и обратно. Кроме того, меню Pulley заменяет традиционные панели инструментов в приложениях.



Чтобы создать пользовательский интерфейс, совместимый с приложениями платформы Sailfish, используйте управление с помощью жестов везде, где это возможно. Например, на правом снимке экрана в диалоговом окне кнопки «Accept» и «Cancel» расположены в заголовке страницы с помощью опции DialogHeader, вместо отдельных кнопок «Accept» и «Cancel». Действия кнопки могут быть реализованы с помощью обработчиков сигналов в диалоге onAccepted и onRejected:

```
Dialog {
    onAccepted: account.logIn()
    onRejected: accountCreationCanceled()
}
```

Несоответствие названия поля и подсказки



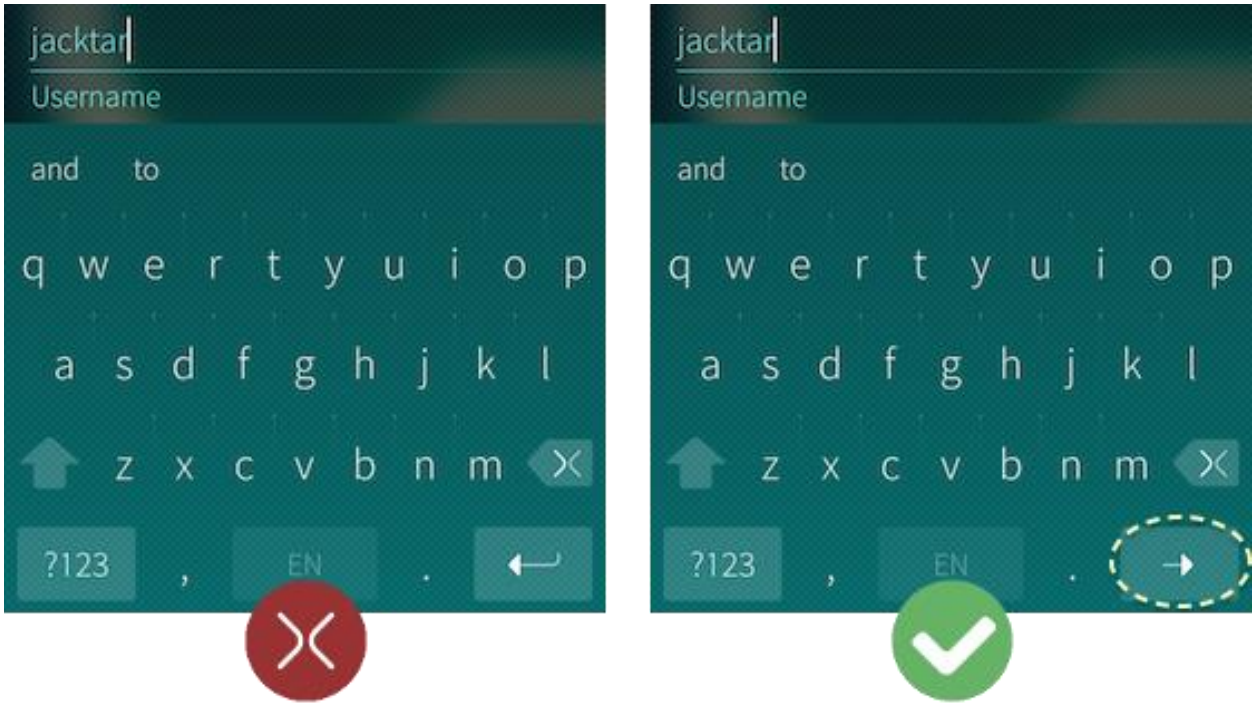
Каждое текстовое поле в Sailfish должно определять значения для свойств `placeholderText` и `label`. Текст подсказки отображается в пустом поле ввода. Он описывает, какие данные должен указать пользователь. Название поля имеет ту же цель, но отображается уже после того, как пользователь ввел текст. Оба свойства могут использовать одну и ту же строку для описания, например: «Имя», «Фамилия», «Электронная почта» и т. д.

```
Column {
    width: parent.width

    TextField {
        placeholderText: "First name"
        label: "First name"
        width: parent.width
    }

    TextArea {
        placeholderText: "Last name"
        label: "Last name"
        width: parent.width
    }
}
```

Отсутствие конфигурации для клавиши Enter



В однострочном текстовом поле клавиша `Enter` не может создать дополнительную строку, поэтому ее функциональность может быть перенастроена с помощью свойства EnterKey. В подобной ситуации клавишу `Enter` обычно используют для перемещения фокуса клавиатуры между последовательными текстовыми полями:

```
TextField {
    label: "Username"
    placeholderText: "Username"
    width: parent.width

    // Нажатие разрешено только после ввода текста
    EnterKey.enabled: text.length > 0

    // Показывает иконку next, чтобы дать понять, что нажатие на
    // клавишу Enter будет перемещать фокус клавиатуры между
    // последовательными текстовыми полями на странице
    EnterKey.iconSource: "image://theme/icon-m-enter-next"

    // После нажатия на клавишу Enter, переместить фокус
    // клавиатуры в следующее поле
    EnterKey.onClicked: passwordField.focus = true
}
```

Если поле на странице последнее, клавишу `Enter` используют для сохранения или отправки введенных данных.

```
TextField {
    label: "Password"
    placeholderText: "Password"
```

```
    width: parent.width

    EnterKey.enabled: text.length > 0
    EnterKey.iconSource: "image://theme/icon-m-enter-accept"
    EnterKey.onClicked: account.login()
}
```

Если описанные выше действия нельзя применить, клавиша `Enter` может закрывать экранную клавиатуру.

```
TextField {
    label: "Password"
    placeholderText: "Password"
    width: parent.width

    EnterKey.iconSource: "image://theme/icon-m-enter-close"
    EnterKey.onClicked: focus = false
}
```

Глубокая вложенность страниц

Стековая модель навигации по страницам приложения позволяет пользователям легко перемещаться между различными экранами. Однако глубокая вложенность страниц все равно может запутать пользователя в том, где находится текущий экран приложения. Вместо того, чтобы постоянно добавлять всё больше страниц в стек, используйте возможность замены страниц с помощью метода replace(). Например, поиск по приложению в платформе Maps заменяет существующие искомые страницы, а не добавляет новые, чтобы избежать громоздкой (и в теории бесконечно растущей) иерархии стека страниц.