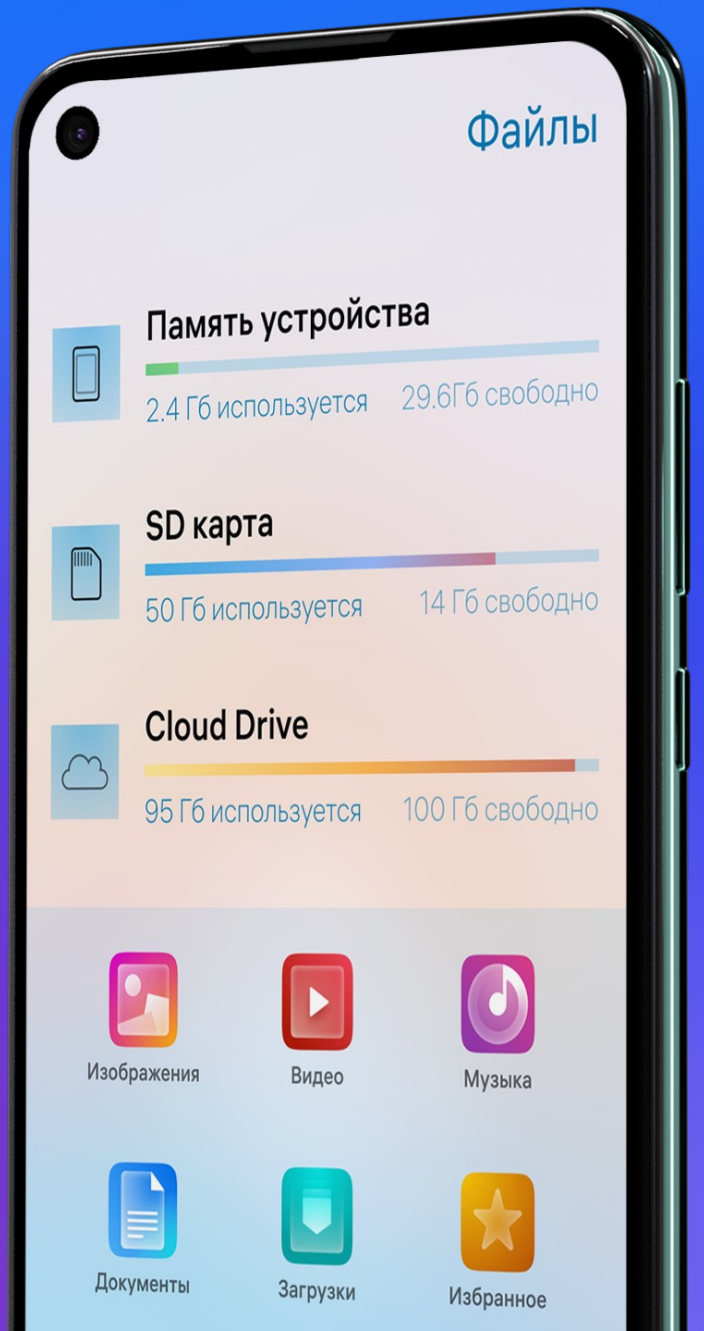
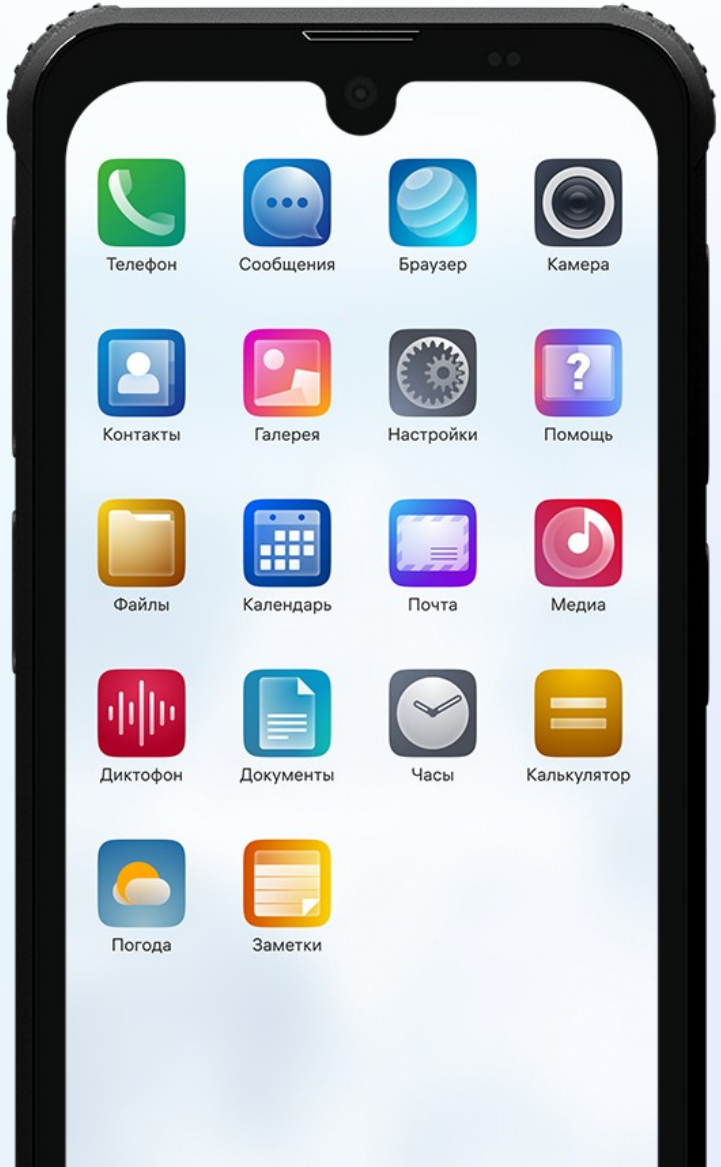
МОБИЛЬНАЯ ОПЕРАЦИОННАЯ СИСТЕМА АВРОРА



Что такое ОС Аврора?

Как писать приложения для ОС Аврора?

Программа Бета-тестирования Аврора



Открытая мобильная платформа

- Офисы разработки в Москве, Санкт-Петербурге, Нижнем Новгороде и Иннополисе
- Дочерняя компания ПАО «Ростелеком»
- Образовательные программы в ведущих ВУЗах страны, прежде всего, в Университете Иннополис!
- Участники open source проектов и организаций

Открытая мобильная платформа

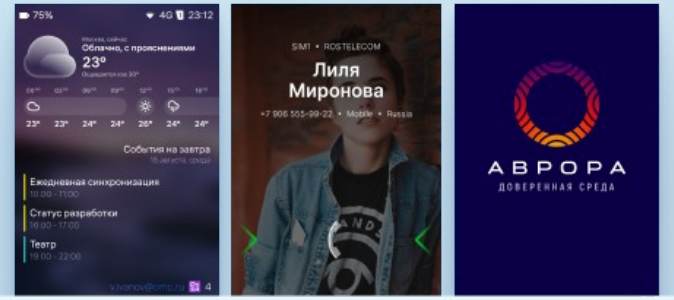
Продукты и направления:

- Мобильная операционная система Аврора
- EMM-решение Аврора Центр
- Доверенная среда исполнения Аврора TEE
- Средство доверенной загрузки Аврора СДЗ

ОПЕРАЦИОННАЯ СИСТЕМА

Четвертое поколение

Современный мобильный функционал
с фокусом на безопасности



АВРОРА
СВОЯ СИСТЕМА

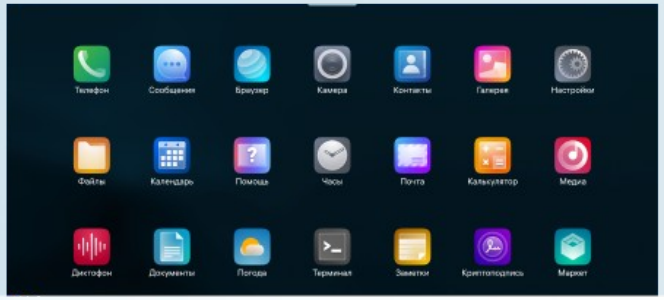
более 400 000 устройств

в промышленной эксплуатации

ЭКОСИСТЕМА ПРИЛОЖЕНИЙ

Более 100 партнеров

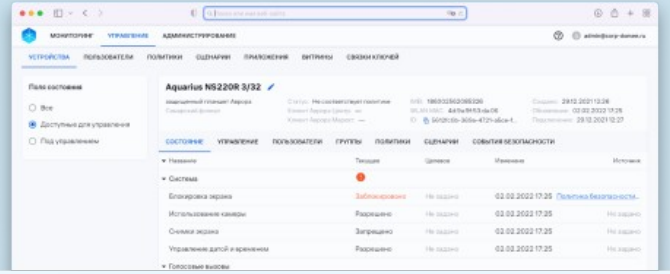
создают свои решения
для ОС Аврора



МОБИЛЬНЫЕ СЕРВИСЫ И MDM

Аврора Центр

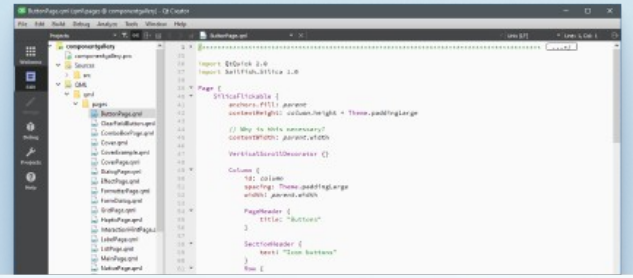
Управление парком устройств
Push-сервис, сервис обновлений,
магазин приложений



СРЕДСТВА РАЗРАБОТКИ

Аврора SDK

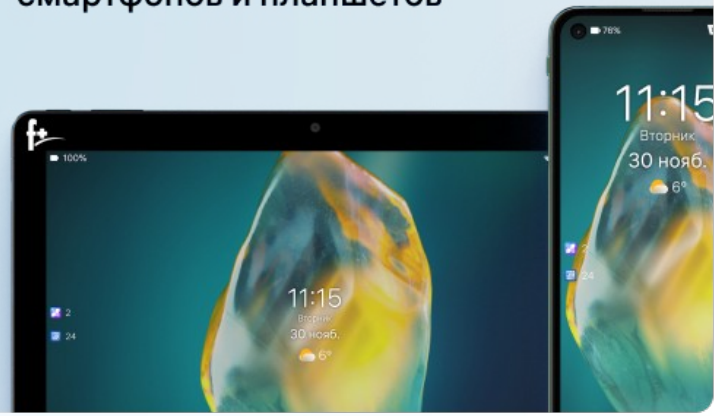
для Windows, MacOS и Linux



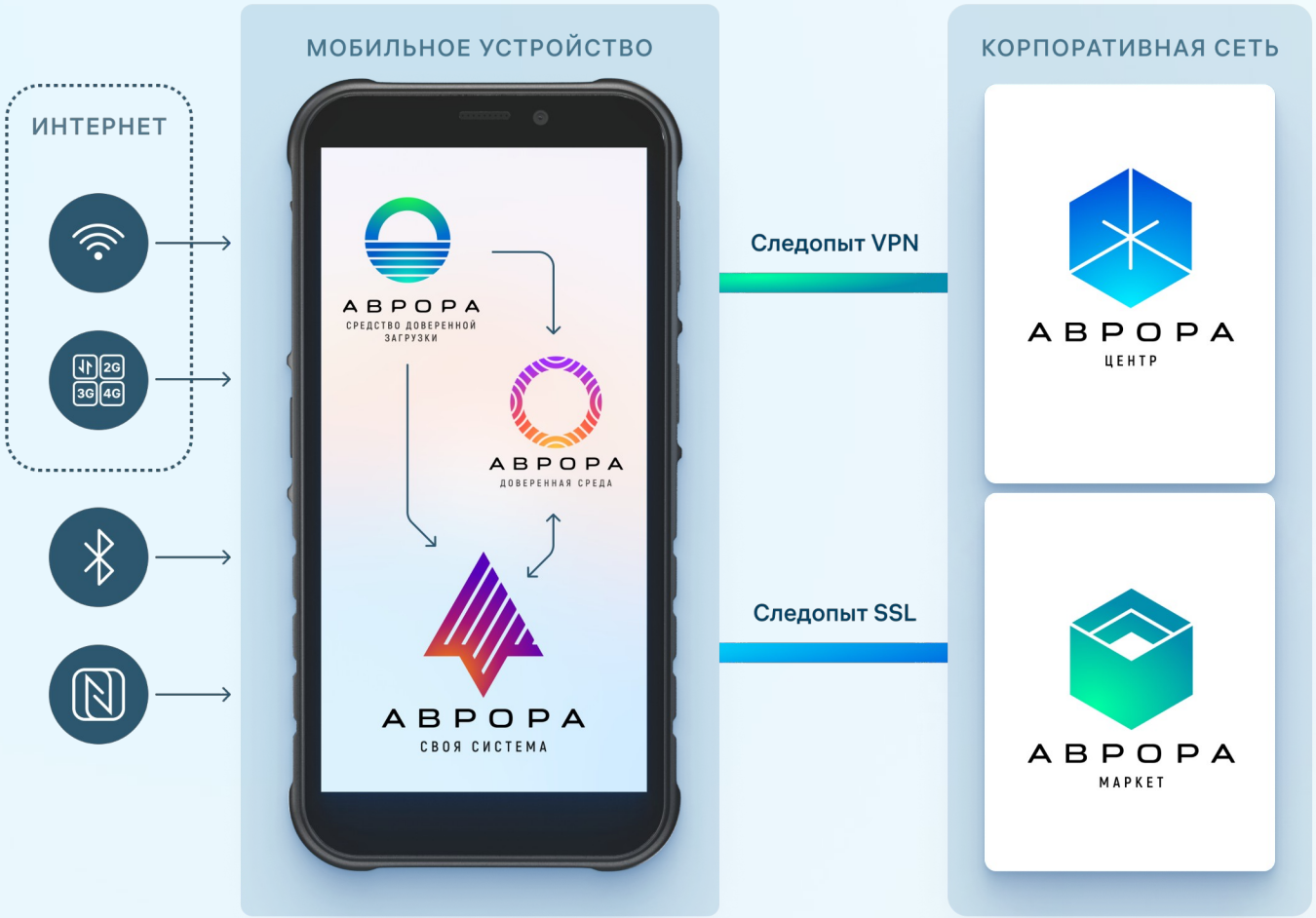
МОДЕЛЬНЫЙ РЯД

10 устройств

смартфонов и планшетов



Архитектура корпоративной мобильности



Флот устройств на ОС Аврора

+

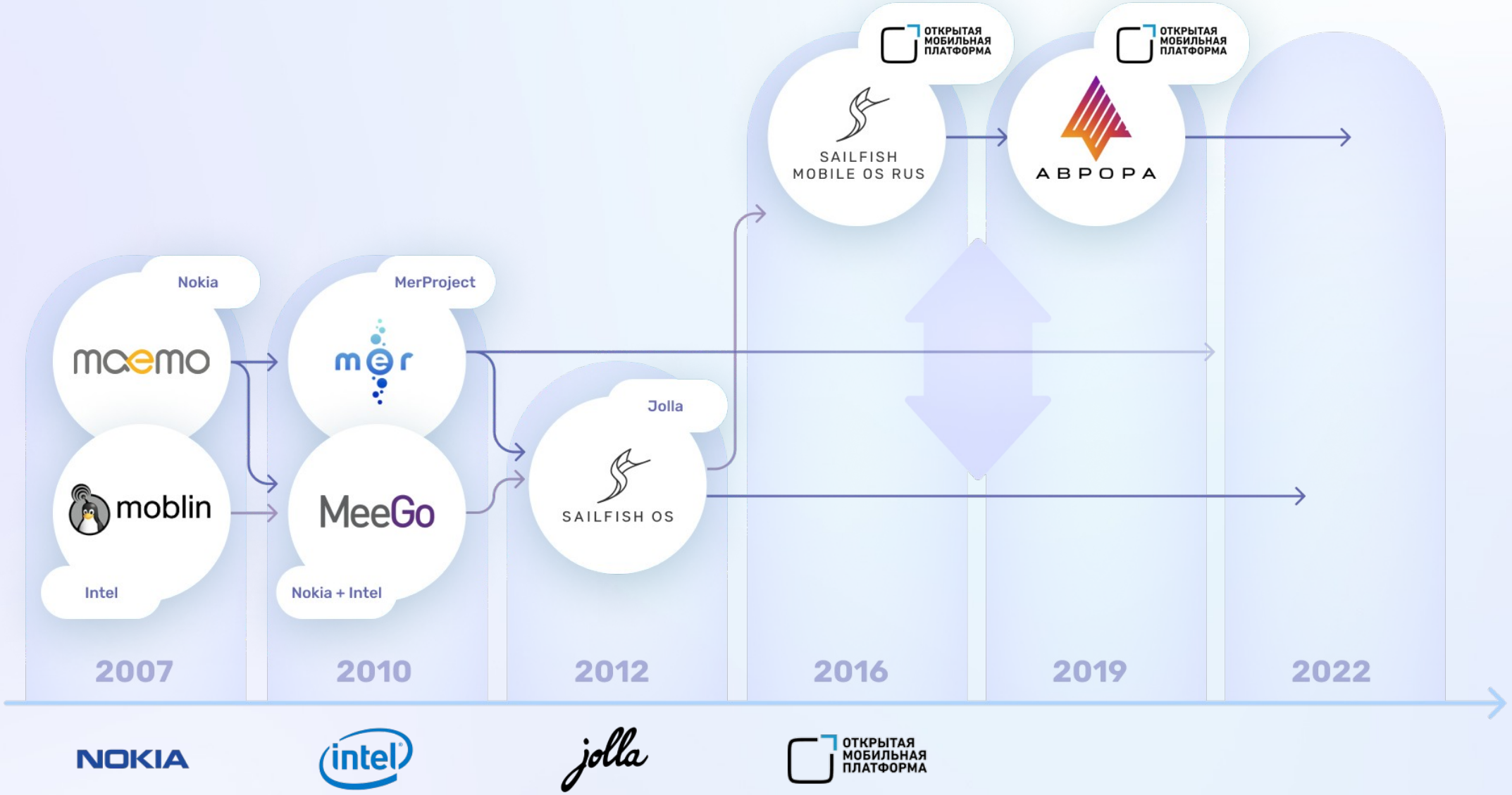
Размещение на серверах компании-эксплуатанта

- Сервис управление политиками безопасности ОС Аврора - MDM
- Сервис обновлений
- Магазин приложений
- Сервис PUSH уведомлений

АРХИТЕКТУРА
БЕЗОПАСНОСТЬ
UX
УСТРОЙСТВА
ПРОЕКТЫ

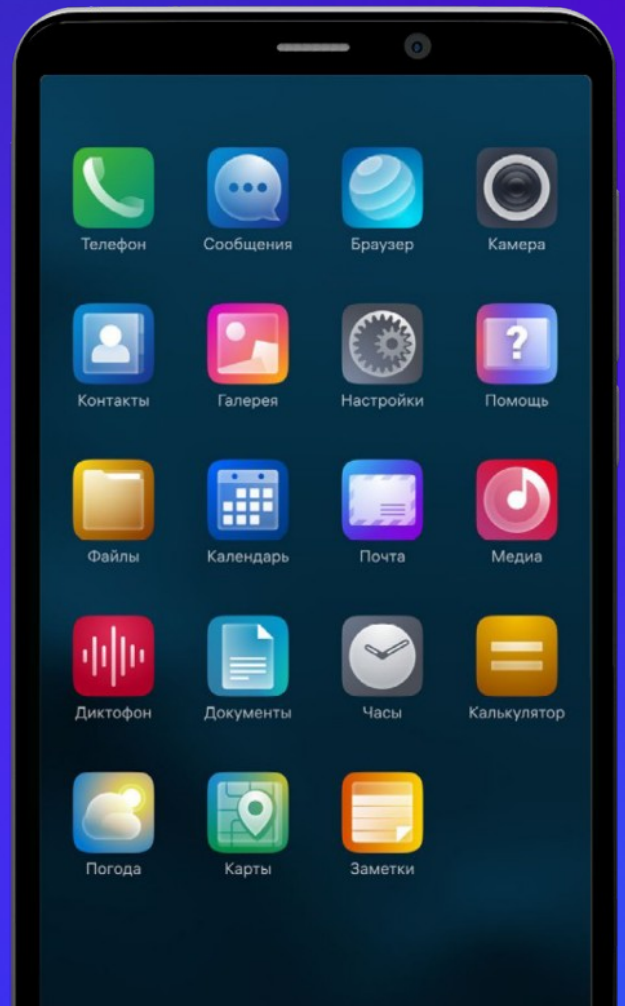
Откуда есть пошла земля...

История ОС Аврора

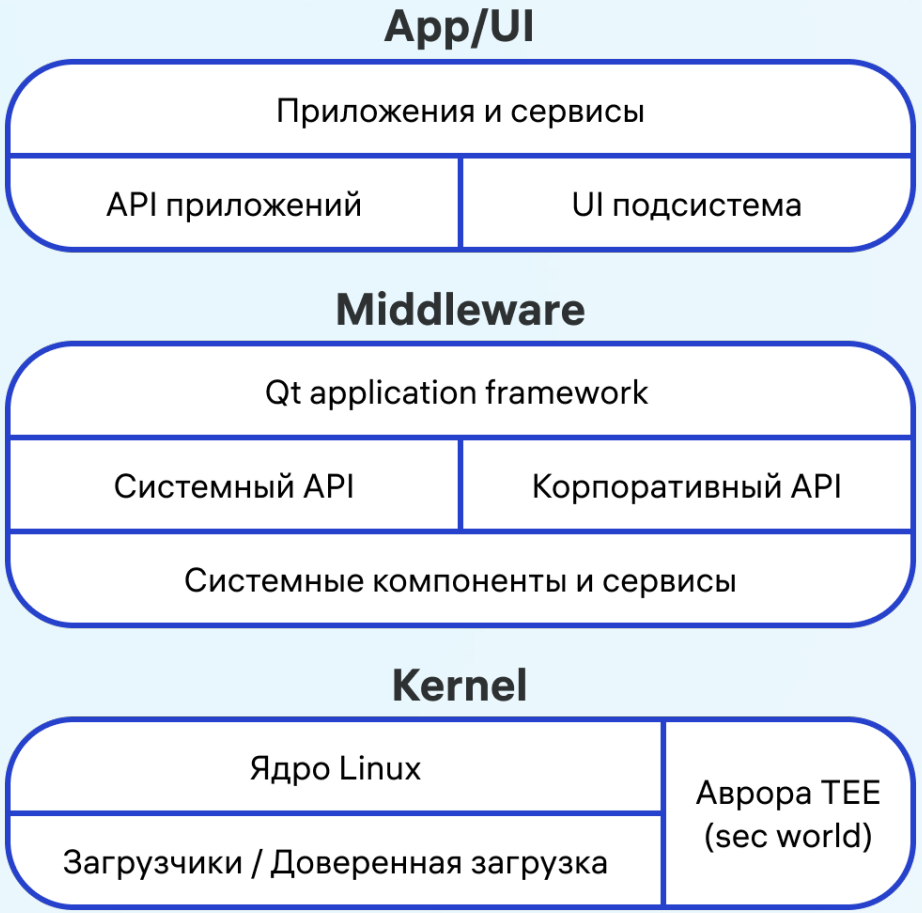


ОС Аврора

- Полнофункциональная мобильная ОС с современным интерфейсом
- Поддержка и предоставление API для сенсоров и периферийных устройств (Bluetooth, NFC, сеть, датчики, GNSS (+Яндекс.Локатор) и т.д.)
- Изоляция приложений, разрешения на API, песочницы, криптохранилище
- Запуск только подписанных вендором и эксплуатантом приложений
- Доверенная загрузка ОС и динамический контроль целостности ядра и компонент
- MDM API для корпоративного применения
- Встроенный браузер с ГОСТ-TLS, возможность КЭП
- Сертифицирована ФСТЭК и ФСБ России



ОС Аврора. Архитектура



- Linux ядро
- GNU/Linux userland
- Wayland
- systemd
- Qt Framework
- RPM пакетный менеджер
- 1600+ пакетов open-source и закрытых

Аврора TEE

- Специализированная ОС обеспечения безопасности устройств
- Использует аппаратные возможности ARM Trustzone

Основные функции

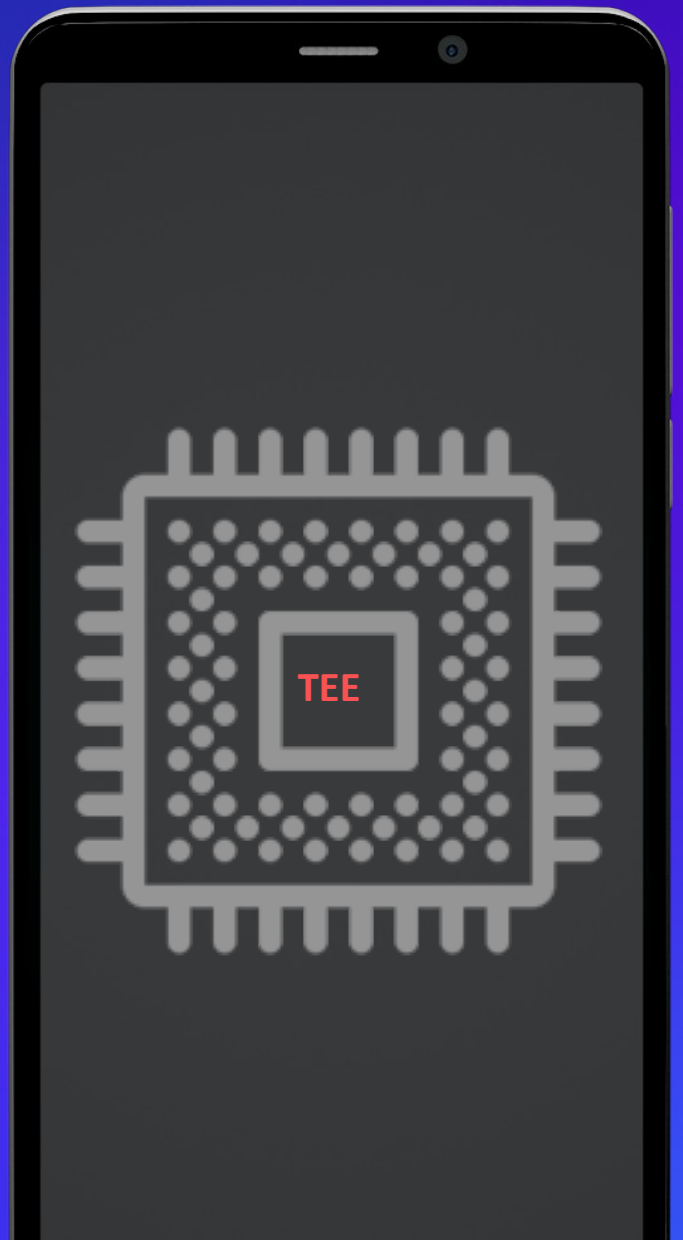
- Динамический контроль целостности основной ОС
- Доверенное хранилище ключей и данных
- Возможность реализации DRM

Ценность:

- Значительно затрудняет взлом устройства и кражу данных
- Необходимость для безопасных платежных решений

Аналогичные продукты:

- Google Trusty
- Samsung TEEgris
- Trustonic TEE

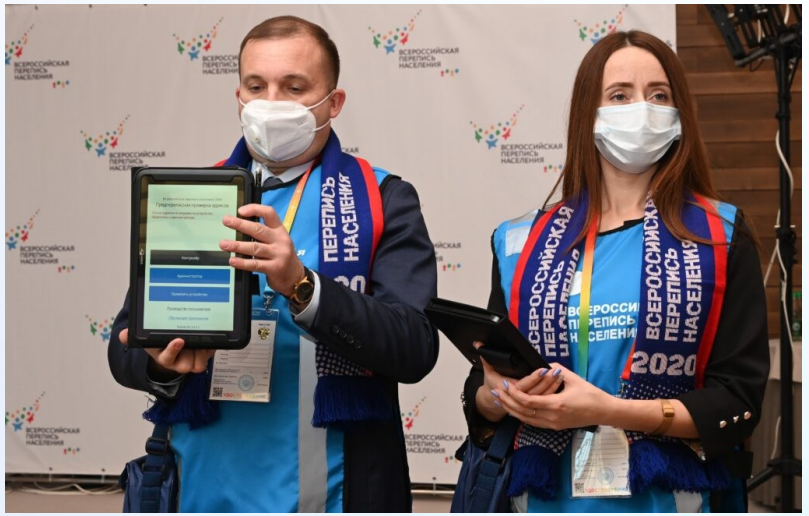


Модель угроз: от кого защищаемся?

Устройство под управлением ОС Аврора в корпоративных сценариях –
это рабочий терминал сотрудника организации

Источники угроз:

- Кража/потеря устройства
- Jailbreak/перепрошивка устройства
- Доступ посторонних к данным на устройстве
- Недоверенные сторонние приложения и данные
- Перехват данных, передаваемых по сети



Отличие от десктопа:

- Мобильность, нахождение устройства в различных окружениях

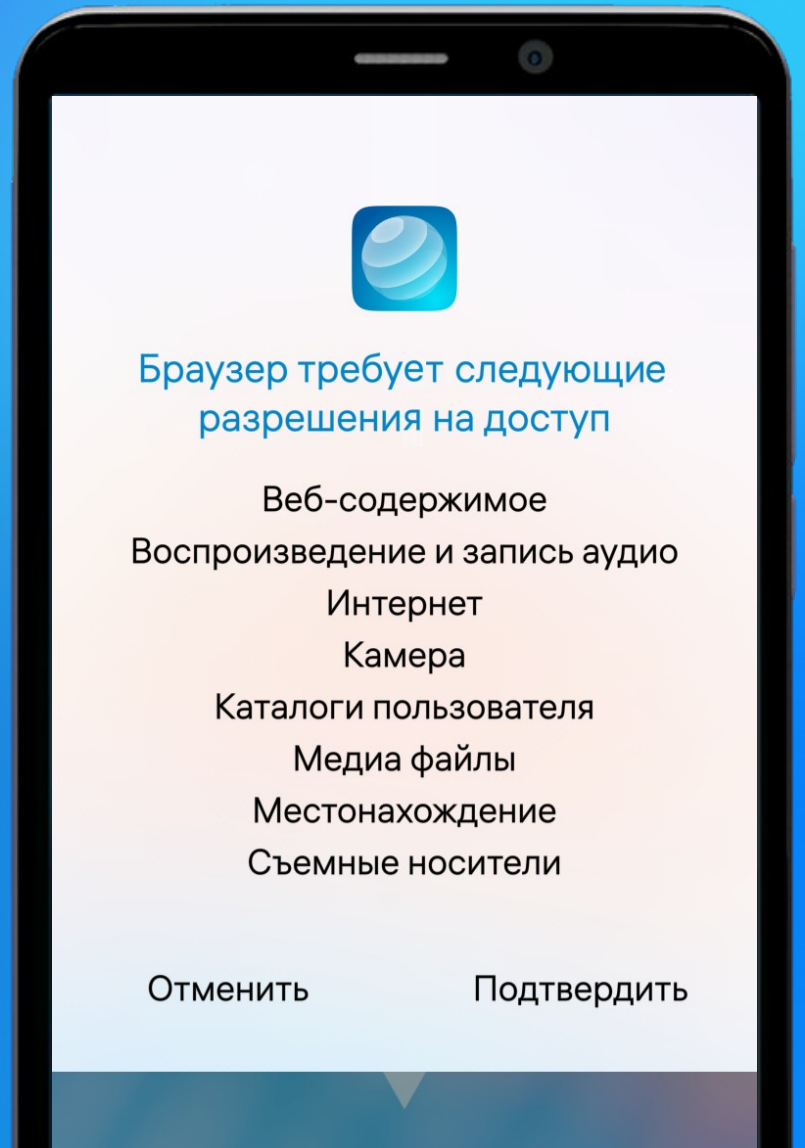
Механизмы безопасности ОС Аврора

ОС Аврора обеспечивает концепцию прикладной эшелонированной безопасности

- прикладная – все механизмы сконфигурированы и работают изначально, не требуют дорогостоящей настройки
- эшелонированная – используется комплекс различных мер и методов, которые пересекаются и дополняют друг друга

Для разработчиков приложений

- Подпись разработчика
  - Проверяется при установке
  - Проверяется при каждом запуске бинарника (IMA)
- Подпись Клиента/Эксплуатанта
  - Проверяется при установке
- Все приложения стартуют в контейнере
- Необходимо декларировать используемое API
  - Существуют различные профили
- Есть системное криптохранилище



Современный интерфейс

- **Общесистемная поддержка жестов**
  используются как для управления системой, так и внутри приложений
- Автоматическая адаптация под **светлую/темную тему**
- **Богатые возможности персонализации внешнего вида приложений**
  цветовые палитры, фоны, звуки, шейдерные эффекты
- **Набор высокопроизводительных элементов управления**
  на любой вкус и пользовательский сценарий, постоянно пополняется
- Универсальные приложения для смартфона и планшета,
  с **возможностью отображения адаптированных интерфейсов**
  для каждого из устройств
- **Продвинутая многозадачность**
  с функционалом быстрых действий прямо с экрана переключения задач
- **Высокая скорость работы интерфейса**
  даже на очень слабых устройствах
- **Расширенный функционал системных уведомлений:**
  несколько видов отображения, кастомная графика, быстрые действия


АВРОРА
НАБОР СИСТЕМНЫХ КОМПОНЕНТОВ
Colors
Light theme
Dark theme
Primary
Secondary
Highlight
Highlight Secondary
Highlight Background
Highlight Dimmer
Font sizes
fontSizeTiny 20pt
fontSizeExtraSmall - 70px
Вторник, 30 ноября 2021 г
Москва
Переменная облачность
Сейчас
День
Вечер
Ночь
Подробный прогноз
Gismeteo
Телефон
Сообщения
Браузер
Камера
Контакты
Галерея
Настройки
Помощь
Медиа
Почта
Календарь
Калькулятор

Набор компонентов в Figma

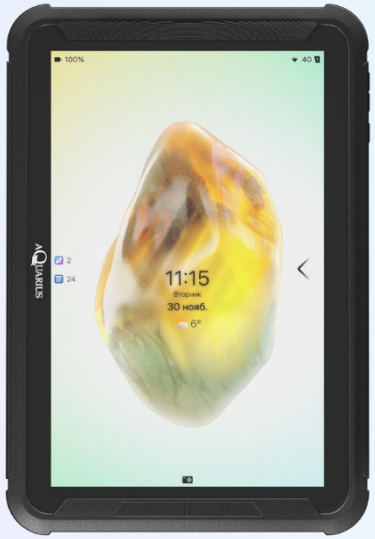
- Всегда синхронизирован с актуальной версией ОС
- Полный набор элементов, от виджетов до цветовой палитры и иконок
- Всегда использует самые последние «фишки» Фигмы: авто-позиционирование, версии компонентов, динамические слои...
- Все элементы доступны в светлом и темном вариантах исполнения
- Содержит в себе целый ряд рекомендаций по верстке интерфейсов под Аврору

https://www.figma.com/community/file/1070268277075531514

Планшеты на ОС Аврора

F+ Life Tab

- MediaTek MT 8766 WA 2ГГц
- Дисплей: 10.1” 1920*1200
- Память: 4/32 ГБ
- Камеры: 8/5МП
- Аккумулятор: 10000 mAh, Li-ion
- NFC



Aquarius NS220 v5.2

- MediaTek MT 8766V 2ГГц
- Дисплей: 10.1” 1920*1200
- Память: 4/32 ГБ
- Камеры: 8/5МП
- Аккумулятор: 10000 mAh, Li-ion
- NFC, POGO, IP54

Aquarius CMP NS208

- MediaTek MT6763V 2ГГц
- Дисплей: 8.0” 1280п800
- Память: 4/64 ГБ
- Камеры: 13/5МП
- Аккумулятор: 9000 mAh, Li-ion
- NFC, POGO, IP68

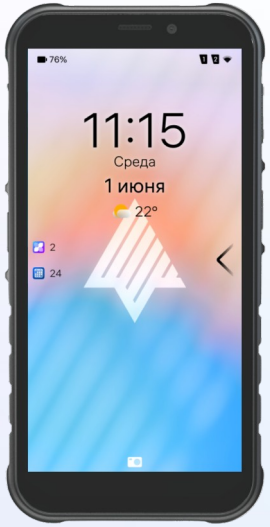
Смартфоны на ОС Аврора

Масштаб Trustphone T1

- MediaTek MT6771V 2,1ГГц
- Дисплей: 6.55” 1600*720
- Память: 4/64 ГБ
- Камеры: 13/12МП
- Аккумулятор: 4000 mAh
- NFC

Qtech QMP-M1-N IP

- MediaTek MT 6739WA 1,3ГГц
- Дисплей: 5.71” 1520*720
- Память: 3/32 ГБ
- Камеры: 13/5МП
- Аккумулятор: 4000 mAh
- NFC
- IP 68



F+ R570

- MediaTek MT6765 2.3-1.8 ГГц
- Дисплей: 5.7” 1440*720
- Память: 4/64 Гб
- Камеры: 13/8МП
- Аккумулятор : 5080 mAh
- NFC
- P68

Aquarius NS M11

- MediaTek MT6765V 2.3-1.8 ГГц
- Дисплей: 7” 2400*1080
- Память: 4/64 Гб
- Камеры: 16/8МП
- Аккумулятор : 5000 mAh
- NFC
- IP68

Инфокиоск на базе процессора Байкал-М

Примеры реализованных проектов

Всероссийская перепись населения

360 000 планшетов
на мобильной ОС Аврора

Ростелеком

Цифровой монтажник
связного оборудования

Почта России

Цифровой почтальон

РЖД Российские железные дороги

Эксплуатация инфраструктуры

ОКБМ Африкантов Росатом

Доверенная мобильная среда

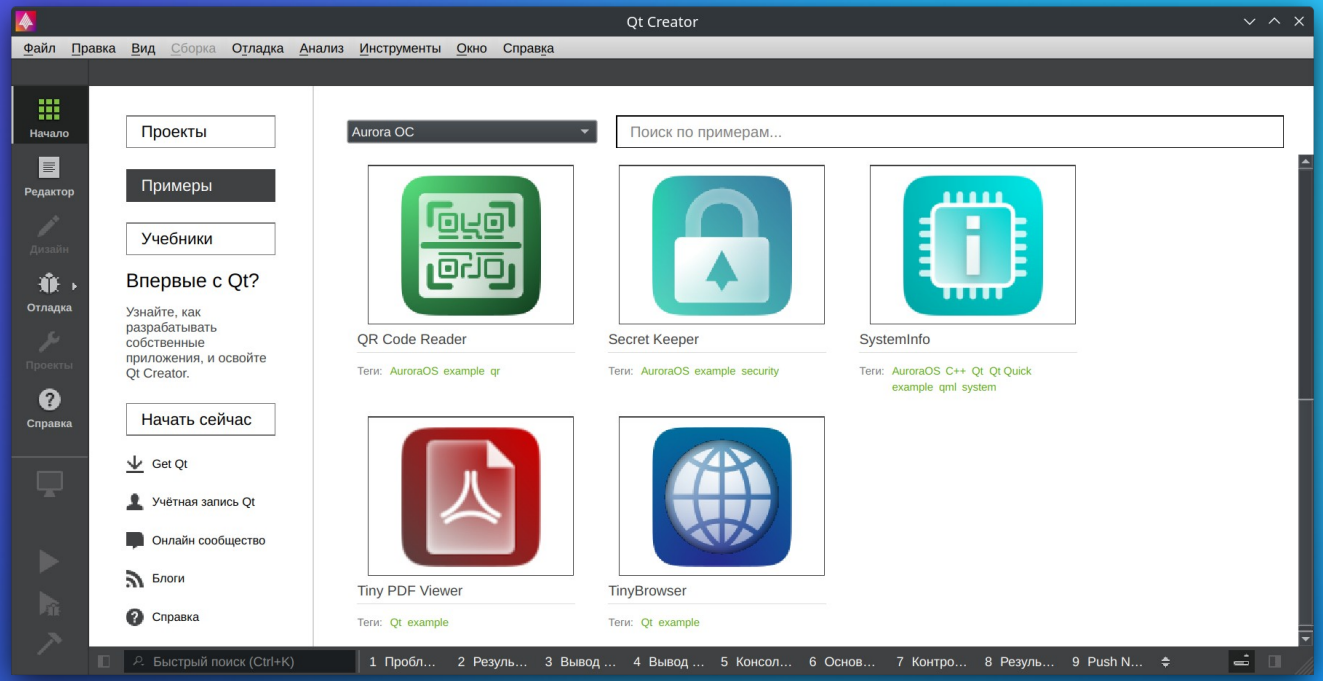
Интер РАО

Контроль исполнения
осмотров и обслуживания

Технологии разработки в ОС Аврора и фреймворк Qt

Аврора SDK

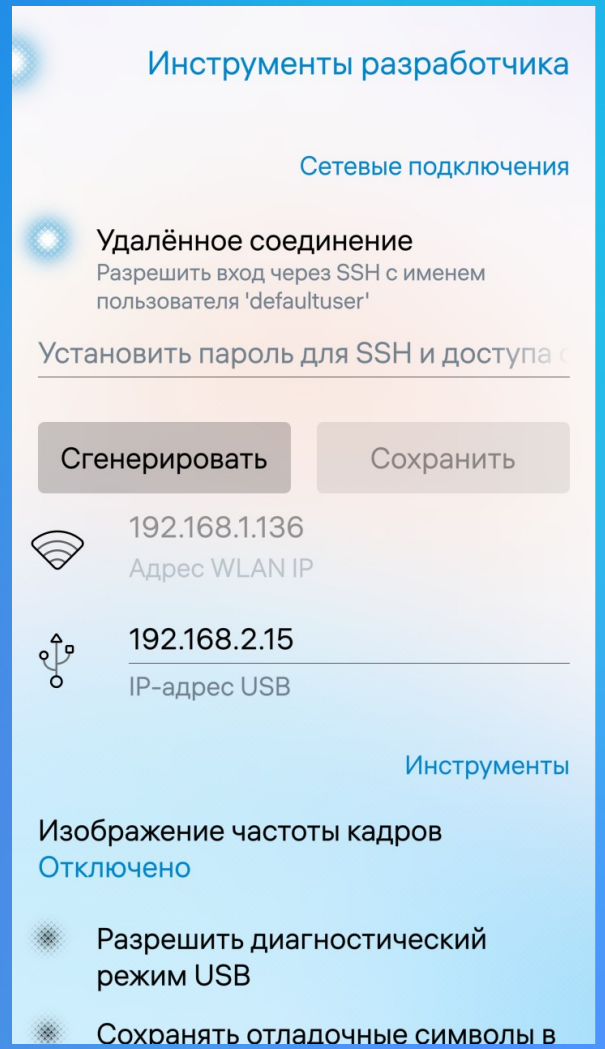
- Среда разработки на основе Qt Creator
- Инструменты сборки
- Эмулятор
- Инструменты подписи и валидации
- Эмуляция Push-сообщений
- Документация
- Примеры

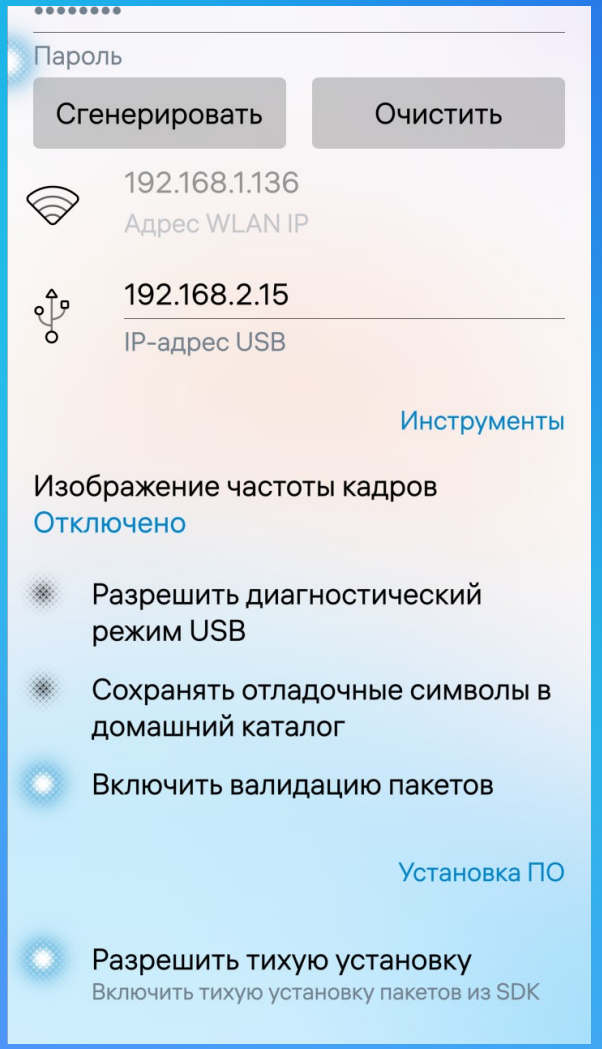


community.omprussia.ru/documentation/software_development/sdk.html

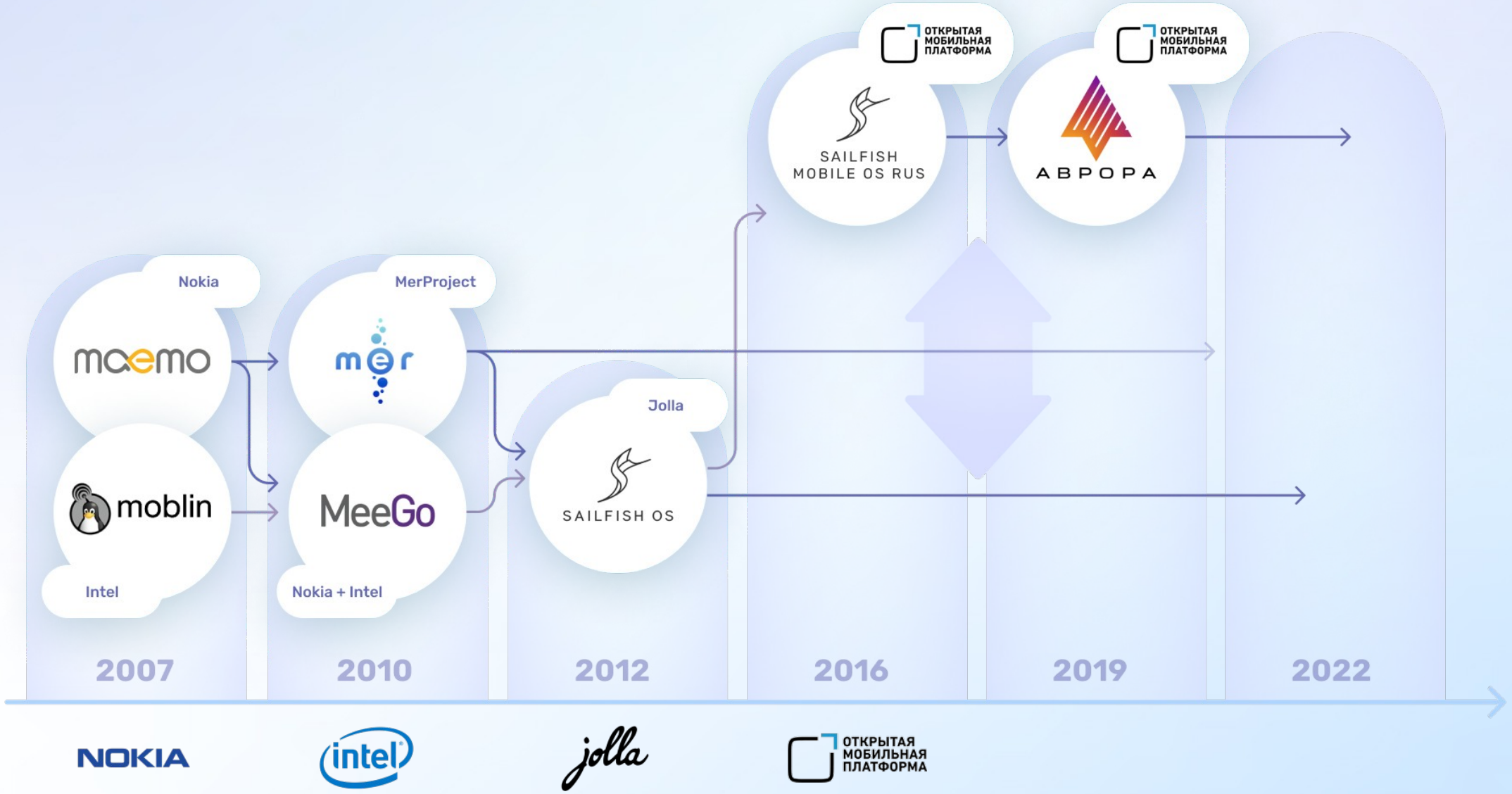
Режим разработчика на устройствах

- Подключение к SDK
- Установка пароля
- Отключение валидатора
- Активация установки без подтверждения
- Включение терминала

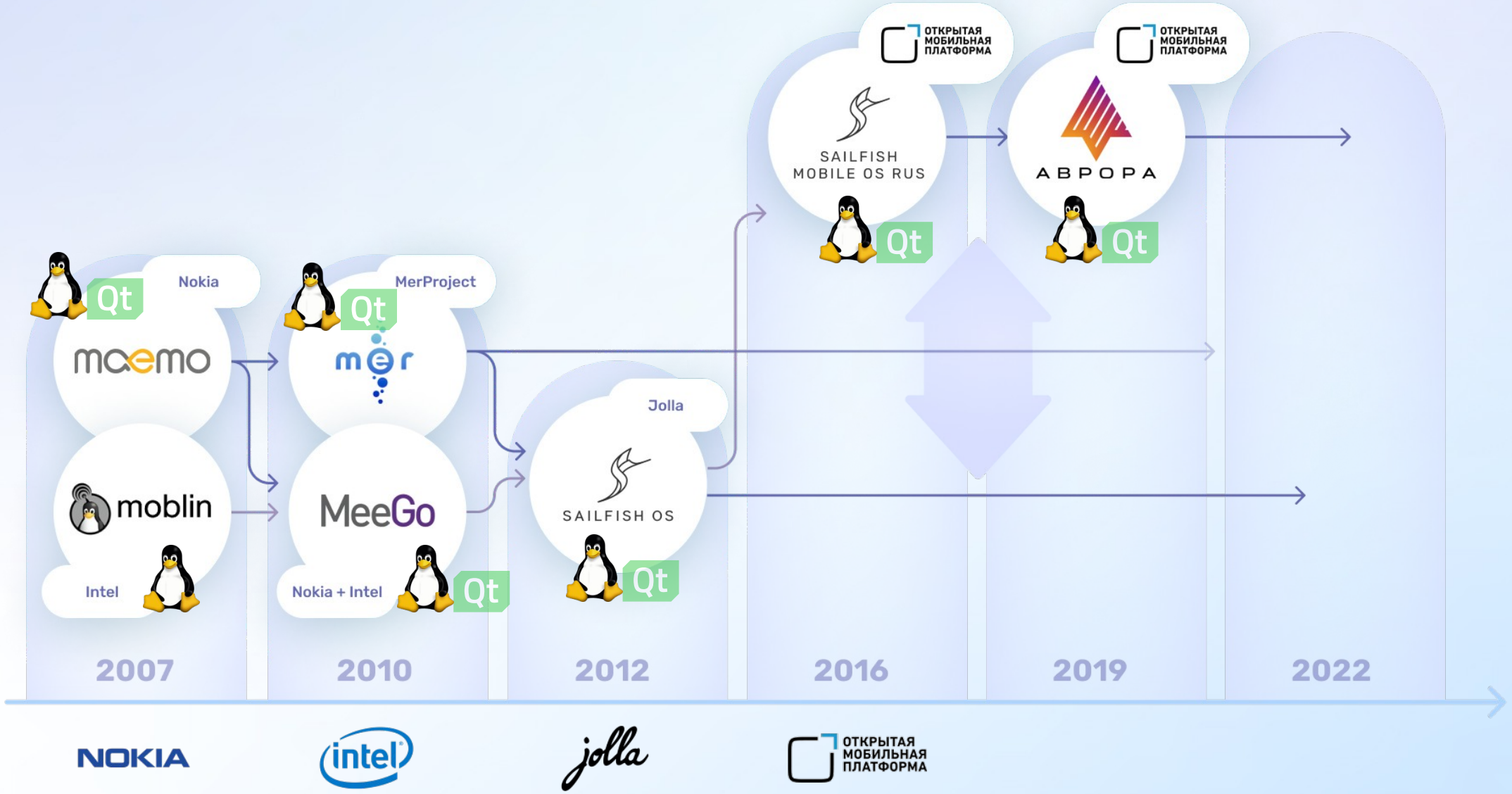




История ОС Аврора



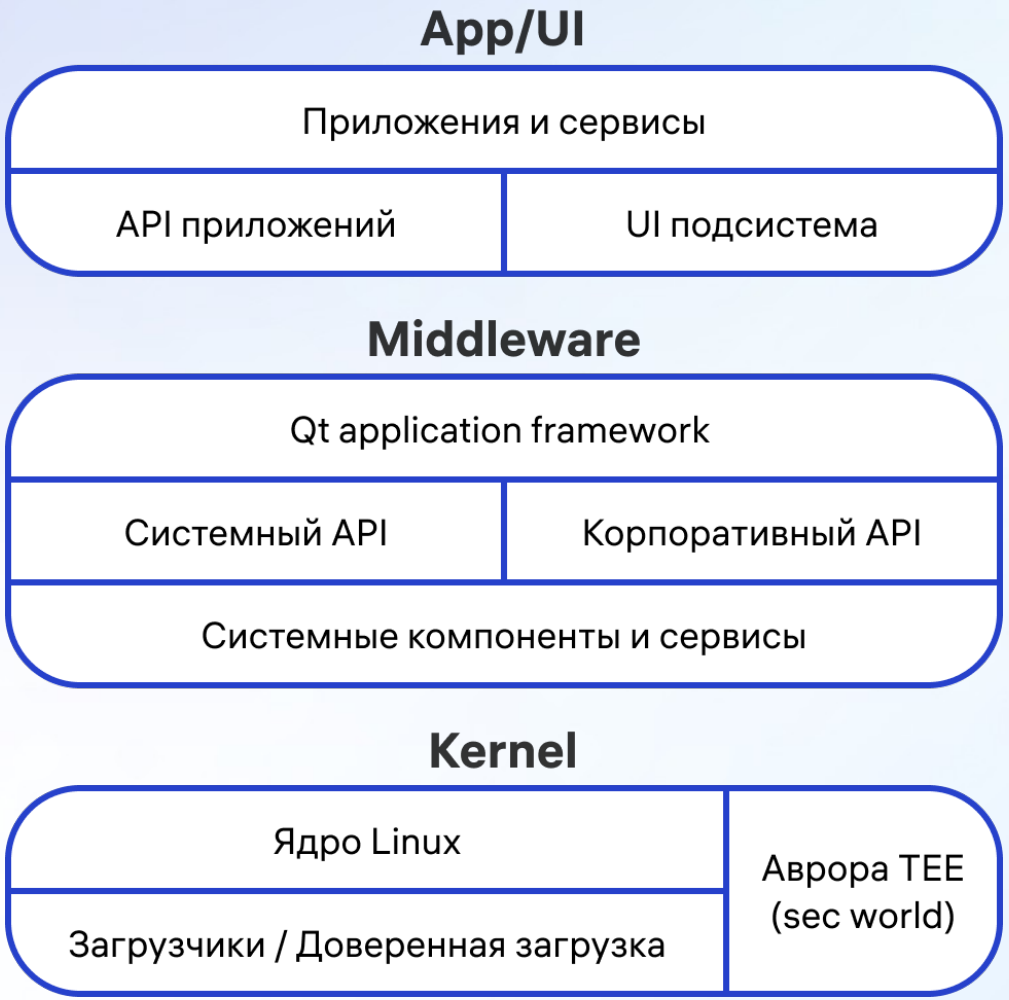
История ОС Аврора



Технологии разработки

- Полноценный GNU/LINUX
- Фреймворк Qt
- Аврора API

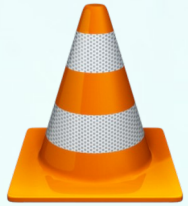
POSIX-совместимая ОС Аврора



- Ядро Linux
- GNU/Linux userland
- systemd
- D-Bus
- Система пакетов RPM
- bash, terminal, ssh и др.

Кроссплатформенный фреймворк Qt

- Среда разработки ПО для различных типов устройств
  - ПК (Linux, Windows, macOS)
  - Мобильные (Аврора и др. Linux, Android, iOS)
  - Встраиваемые
- Сообщество разработчиков более 1 500 000 человек
- Можно бесплатно использовать в коммерческом ПО

Концепции Qt

- Абстракция GUI
  - UI/UX ориентирован на платформу
  - Общая бизнес-логика

- Сигналы и слоты
  - Удобный способ отправки и обработки информации о событиях
  - Возможность отслеживать значения свойств
- Метаобъектный компилятор
  - Обрабатывает макросы
  - Создает добавленный исходный код с метаинформацией
  - Расширяет синтаксис языка C++

Code less, create more, deploy everywhere

www.qt.io/qt-for-application-development

Мобильные приложения на Qt

Декларативное описание UI

- Язык QML
- Стандартный модуль QtQuick
- Специфичный модуль Silica
- Доступ к некоторым API
- «Лёгкая» логика на «JavaScript»

Нативная бизнес-логика

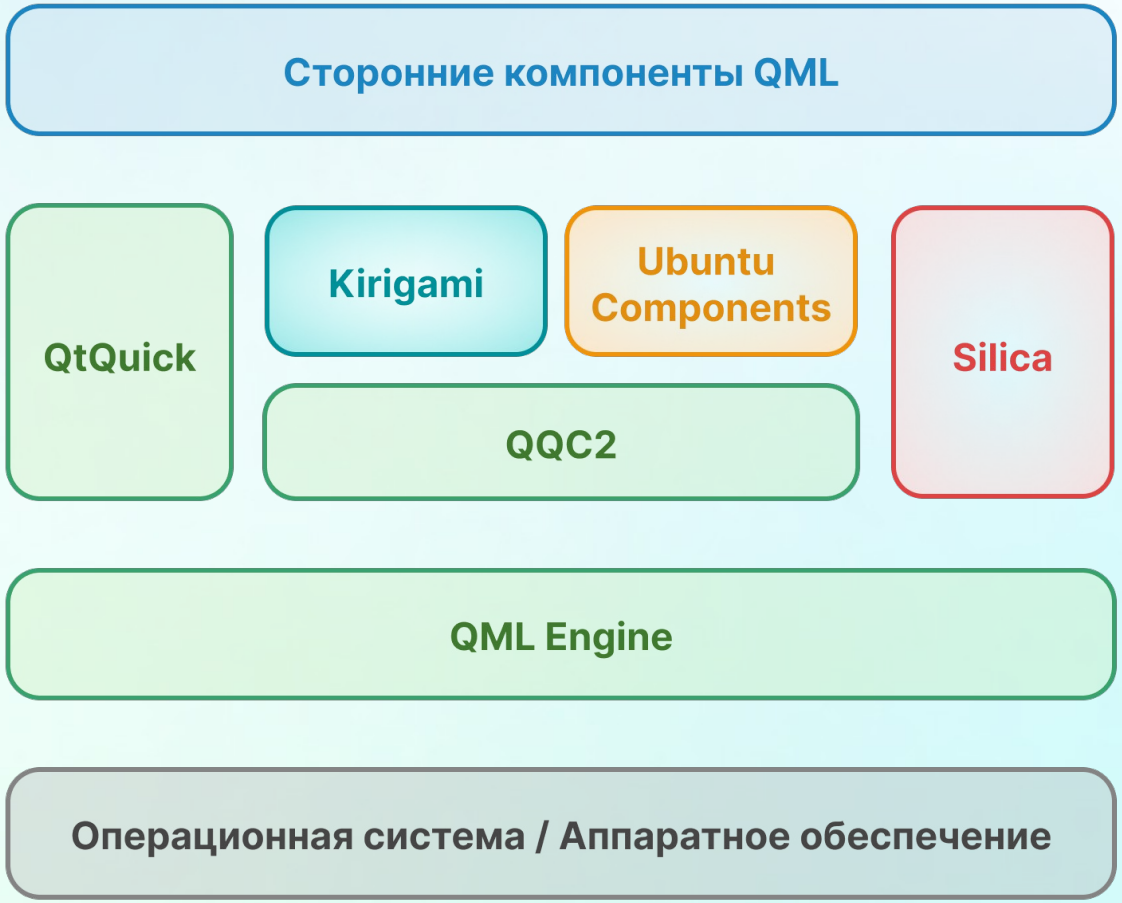
- Язык C++
- QQuickView для UI
- Доступ к библиотекам Qt и C++
- Низкоуровневый доступ к API устройства
- «Тяжёлые» вычисления

Qt Quick

Технология быстрой разработки

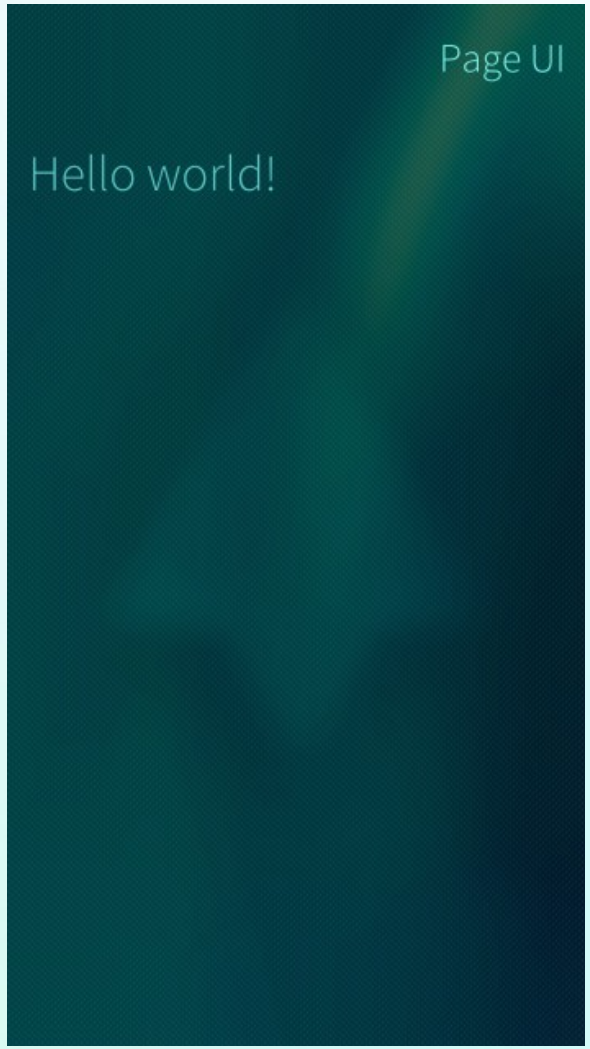
- QML — язык разметки
  - Декларативный
  - Поддерживает вставки на «JavaScript»
  - Доступны свойства объектов C++
- QML Engine — интерпретатор QML
- QtQuick — общие компоненты QML, поставляемые Qt

```
import QtQuick 2.0
```



QML: объекты, свойства, привязки

```
import QtQuick 2.6
import Sailfish.Silica 1.0

Page {
    Column {
        id: column
        width: parent.width
        spacing: Theme.paddingLarge

        PageHeader { title: qsTr("Page UI") }

        Label {
            x: Theme.horizontalPageMargin
            text: qsTr("Hello world!")
            color: Theme.highlightColor
            font.pixelSize: Theme.fontSizeHuge
        }
    }
}
```



Элементы компонента QML

- `id`
  Идентификатор для обращения к объекту
- `Properties`
  Свойства заданных типов, обладающие названиями и значениями
- *`Methods`*
  Исполняемый код на «JavaScript»
- `Signals`
  Уведомления от объекта QML
- `Signal Handlers`
  Выражения или функции, инициируемые сигналами
- `Nested objects`
  Вложенные объекты

Примеры форматированного текста

```
Text {
    y: 0; width: parent.width
    text: "<b>Hello</b> <i>World!</i>"
    color: "red"
    font.pointSize: 48
}
Text {
    y: 200; width: parent.width
    text: "<b>Hello</b> <i>World!</i>"
    color: "green"
    font { pointSize: 48; underline: true }
    textFormat: Text.RichText
}
Text {
    y: 400; width: parent.width
    text: "<b>Hello</b> <i>World!</i>"
    color: "blue"
    font { pointSize: 32; bold: true }
    textFormat: Text.PlainText
}
```

Пример image

```
Image {
    width: parent.width; height: parent.height
    source: "avrora.svg"
}
```

- fillMode : enumeration
  - Image.Stretch — масштабировать по элементу
  - Image.PreserveAspectFit — масштабировать без обрезки
  - Image.PreserveAspectCrop — масштабировать с обрезкой
  - Image.Tile — дублировать горизонтально и вертикально
  - Image.TileHorizontally — растянуть вертикально, дублировать
  - Image.TileVertically — растянуть горизонтально, дублировать
  - Image.Pad — не масштабировать

Пример нажимаемой кнопки

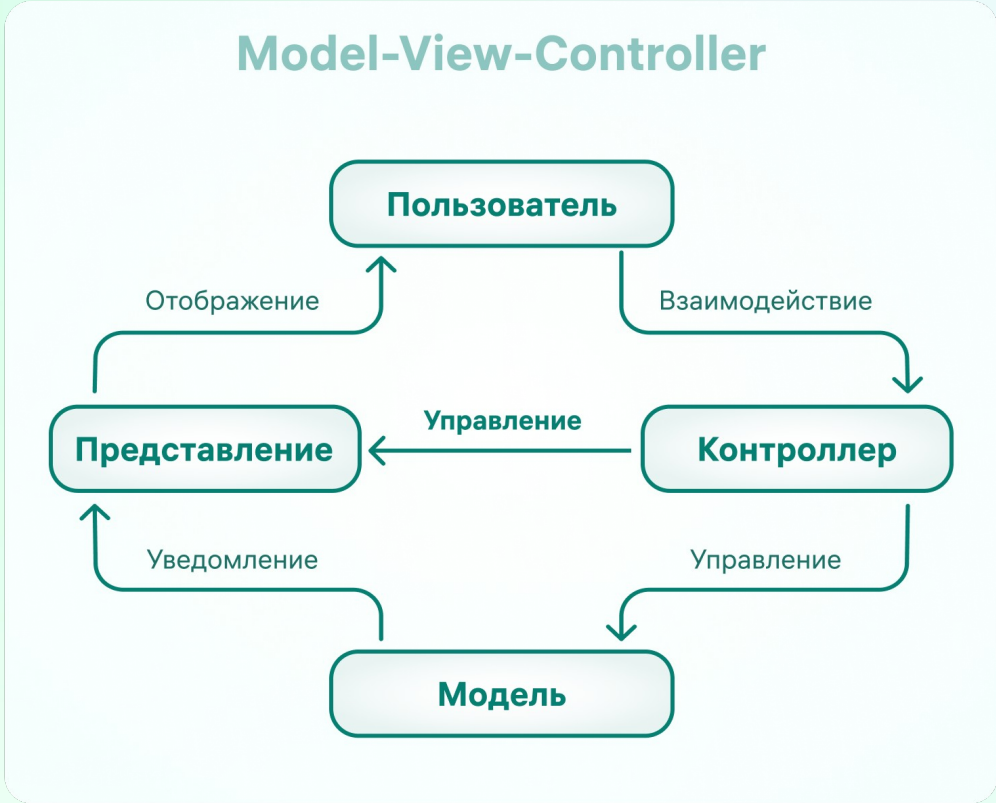
```
Rectangle {
    color: mouseArea.pressed ? "green" : "blue"

    Text {
        anchors.centerIn: parent
        text: "Click Me"
        color: "red"
        font { bold: true; pixelSize: 48 }
    }
    MouseArea {
        id: mouseArea
        anchors.fill: parent
        onClicked: console.log("clicked")
    }
}
```

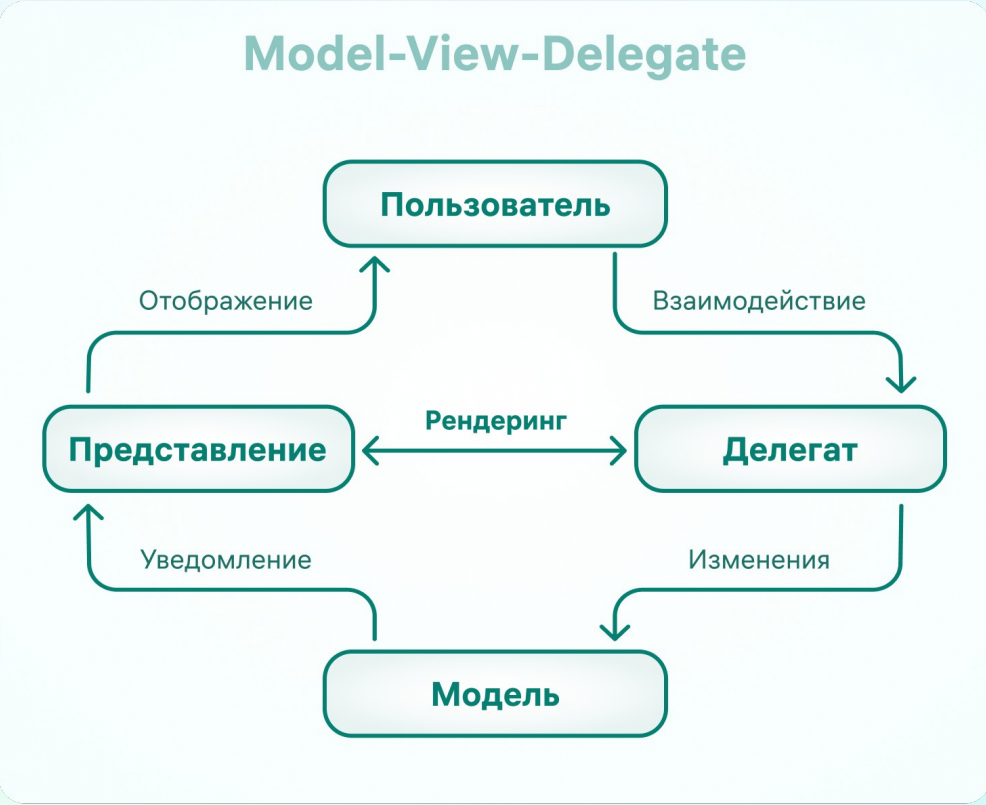
Пример состояний и переходов

```
MouseArea { id: mouseArea }
Rectangle {
    id: rectangle
    color: "red"
    states: State {
        name: "blue"; when: mouseArea.pressed

        PropertyChanges  { target: rectangle; color: "blue" }
    }
    transitions: [
        Transition {
            to: "blue"; ColorAnimation { duration: 2000 }
        },
        Transition {
            from: "blue"; ColorAnimation { duration: 500 }
        }
    ]
}
```

Модели и представления

- Отображение и взаимодействия с однородным контентом
- Можно реализовывать свои

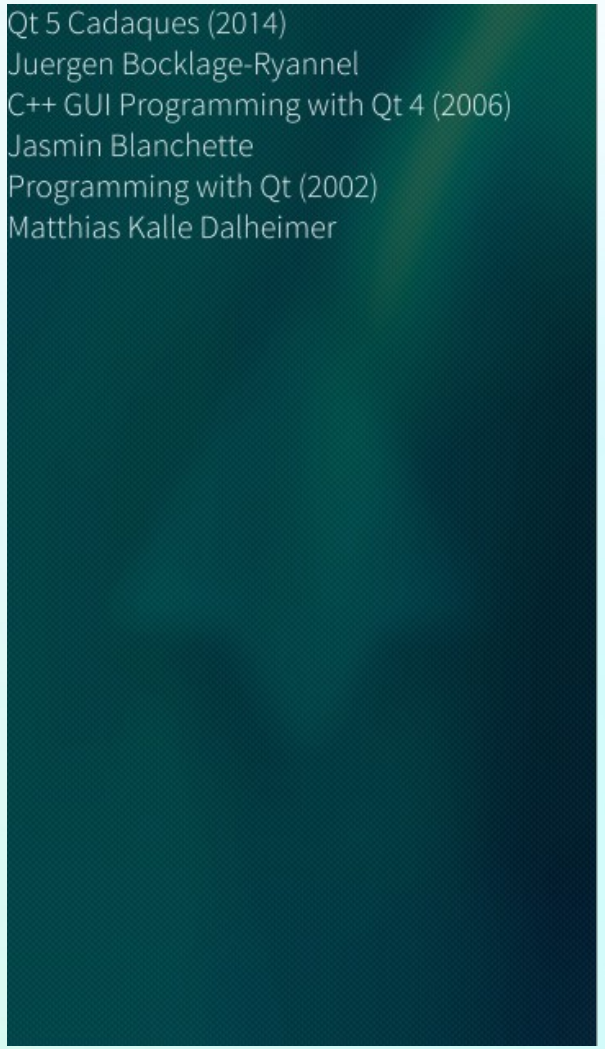




doc.qt.io/qt-5/qtquick-modelviewsdata-modelview.html

Пример модели xml

```
import QtQuick.XmlListModel 2.0
Page {
    XmlListModel {
        id: xmlListModel
        source: "books.xml"
        query: "/catalog/book"
        XmlRole { name: "title"; query: "title/string()" }
        XmlRole { name: "year"; query: "year/number()" }
        XmlRole { name: "author"; query: "author/string()" }
    }
    ListView {
        anchors.fill: parent
        model: xmlListModel
        delegate: Column {
            Text { text: title + " (" + year + ")" }
            Text { text: author }
        }
    }
}
```



Silica.ApplicationWindow — точка входа

- initialPage : var — какая страница будет загружена при запуске
- background.* — настройки фона (цвет, изображение и т.п.)
- cover : var — определяет обложку приложения
- pageStack : PageStack — стек отображаемых страниц,
- allowedOrientations : enumeration — набор доступных ориентаций экрана
  - Orientation.All
  - Orientation.PortraitMask, Orientation.Portrait, Orientation.PortraitInverted (не для смартфонов)
  - Orientation.LandscapeMask, Orientation.Landscape, Orientation.LandscapeInverted
- activate() — вывести приложение в полноэкранный режим
- deactivate()— свернуть приложение

ApplicationWindow — точка входа

```
import QtQuick 2.6
import Sailfish.Silica 1.0

ApplicationWindow {
    initialPage: Component { Page { } }
    cover: Component { CoverBackground { } }
    allowedOrientations:
        defaultAllowedOrientations
}
```

UI Kit в Figma

- Для актуальной версии ОС
- Полный набор элементов:
  от виджетов до цветовой палитры
  и иконок
- Для светлой и тёмной тем
- Рекомендации по вёрстке

www.figma.com/community/file/1070268277075531514

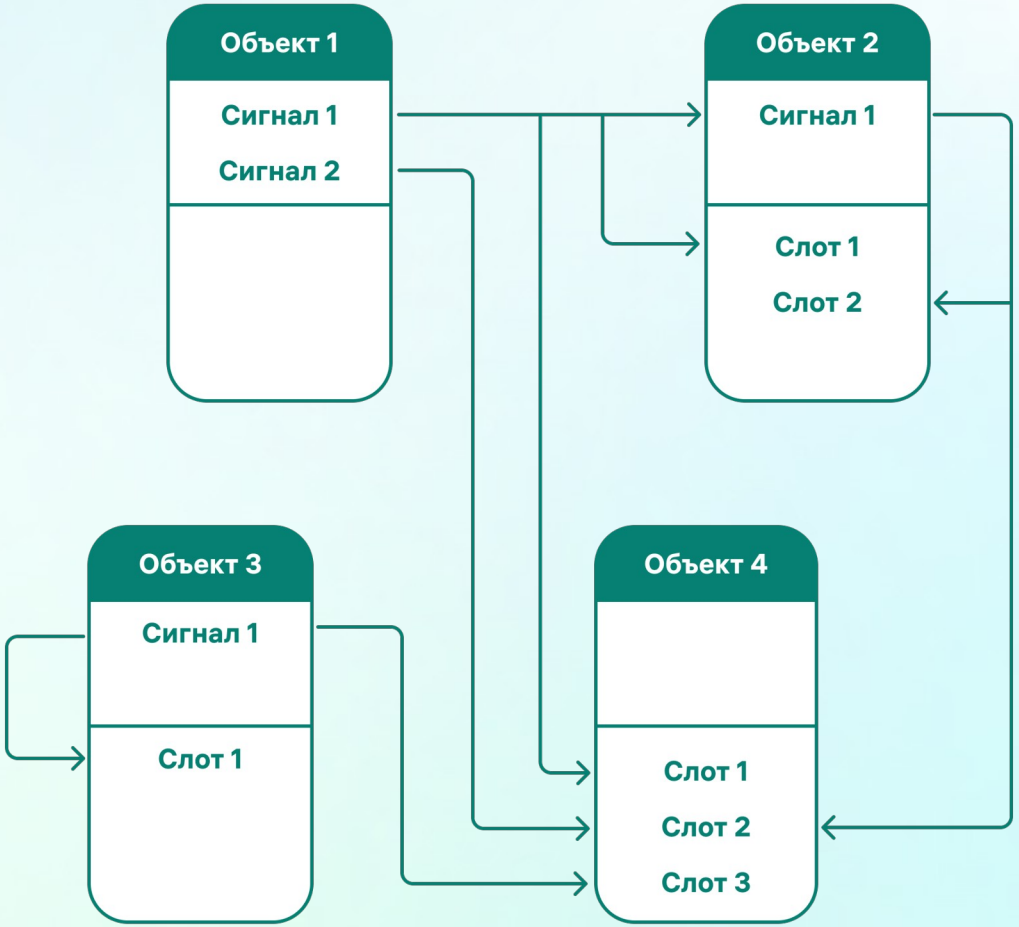
Расширение C++ классов

```
class Counter
{



public:
    Counter() { m_value = 0; }
    int value() const { return m_value; }


    void setValue(int value);




private:
    int m_value;
};
```

```
#include <QObject>

class Counter : public QObject
{
    Q_OBJECT
    Q_PROPERTY (int value READ value
        WRITE setValue NOTIFY valueChanged)

public:
    Counter() { m_value = 0; }
    int value() const { return m_value; }

public slots:
    void setValue(int value);

signals:
    void valueChanged(int newValue);

private:
    int m_value;
};
```

Сигналы и слоты

- Сигналы
  - Испускаются объектом при изменении его состояния
  - Публичные методы, доступны отовсюду (рекомендуется испускать только владельцем)
  - Не возвращают значений
  - Могут принимать аргументы
  - Реализация генерируется с помощью moc
- Слоты
  - Методы, к которым можно подключать сигналы
  - Не должны возвращать значение
  - Могут быть приватными и публичными
  - Могут быть вызваны как обычные методы
  - Вызываются при испускании присоединённых сигналов (даже приватные)



doc.qt.io/qt-5/signalsandslots.html

Регистрация класса для QML

- Регистрация типа в C++

```
#include <QtQuick>
#include "ClassName.h"
…

int main(int argc, char *argv[])
{
    qmlRegisterType<ClassName>("module.name", 1, 0, "TypeName");
    …
```

- Использование зарегистрированного типа в QML

```
import module.name 1.0
…
TypeName { … }
…
```

doc.qt.io/qt-5/qqmlengine.html#qmlRegisterType

Объявление визуального элемента

```
#include <QQuickItem>

class ClassName : public QQuickItem
{
    Q_OBJECT
    … // declaration of properties
public:
    explicit ClassName(QQuickItem *parent = nullptr);
    …

    QSGNode *QQuickItem::updatePaintNode(QSGNode *oldNode,
        QQuickItem::UpdatePaintNodeData *updatePaintNodeData);
signals: …
public slots: …
private: …
private slots: …
};
```

doc.qt.io/qt-5/qquickitem.html

Пример создания модели

```
#include <QAbstractListModel>

class DemoModel : public QAbstractListModel
{
    Q_OBJECT

public:
    enum DemoRoles {
        NameRole = Qt::UserRole + 1,
    };
    explicit DemoModel(QObject *parent = 0);
    virtual int rowCount(const QModelIndex&) const { return backing.size(); }
    virtual QVariant data(const QModelIndex &index, int role) const;
    QHash<int, QByteArray> roleNames() const;
    Q_INVOKABLE void activate(const int i);

private:
    QVector<QString> backing;
};
```

doc.qt.io/qt-5/qabstractlistmodel.html

Контейнеры Qt

- Готовые к использованию структуры данных
- Range-based for
- Итераторы
  - Java-Style
  - STL-Style
- Совместимы с STL-алгоритмами
- Работа с потоками
- Implicit Sharing

- `QList<T>`
- `QLinkedList<T>`
- `QVector<T>`
- `QVarLengthArray<T, Prealloc>`
- `QStack<T>`
- `QQueue<T>`
- `QSet<T>`
- `QMap<Key, T>`
- `QMultiMap<Key, T>`
- `QHash<Key, T>`
- `QMultiHash<Key, T>`

Модули Qt 5

Qt Essentials		Qt Add-Ons		
Qt Core	Qt Quick Controls 2	Active Qt	Qt Network Authorization	Qt Speech
Qt GUI	Qt Quick Dialogs	Qt 3D	Qt NFC	Qt SVG
Qt Multimedia	Qt Quick Layouts	Qt Android Extras	Qt Platform Headers	Qt UI Tools
Qt Multimedia Widgets	Qt Quick Test	Qt Bluetooth	Qt Positioning	Qt WebChannel
Qt Network	Qt SQL	Qt Canvas 3D	Qt Print Support	Qt WebEngine
Qt QML	Qt Test	Qt Concurrent	Qt Purchasing	Qt WebSockets
Qt Quick	Qt Widgets	Qt D-Bus	Qt Quick Controls	Qt WebView
		Qt Gamepad	Qt Quick Extras	Qt Windows Extras
		Qt Graphical Effects	Qt Quick Widgets	Qt X11 Extras
		Qt Help	Qt SCXML	Qt XML
		Qt Image Formats	Qt Sensors	Qt XML Patterns
		Qt Location	Qt Serial Bus	Qt Wayland Compositor
		Qt Mac Extras	Qt Serial Port	

doc.qt.io/qt-5/qtmodules.html

Что даёт опыт использования Qt

Знания

- Императивный и декларативный подходы
- Мета-объектная парадигма
- Понимание контейнеров и действий с ними
- Потоки
- Паттерны
  - Implicit Sharing
  - MVC, MVD
  - PIMPL
- События
  - Очередь
  - Обработчики
  - Сигналы и слоты как альтернатива обработчикам

Практические навыки

- Написание кода на QML, C++ и JS
- Проектирование ПО
  - Гибридная связка (декларатив + императив)
  - Модели-представления
- Работа с API
  - Мультимедиа
  - Сети
  - БД
  - Файлы
  - Датчики
  - Картография
  - ...

ОС Аврора для разработчиков

- Механизмы безопасности
  - Подпись установочных пакетов
  - Валидация установочных пакетов
  - Изоляция приложений
  - Многопользовательский режим
  - Шифрование домашних директорий
  - API контроля целостности
- Прикладные API
  - Push
  - MDM
  - WebView
  - Мультимедиа
  - Криптография
  - ...

Развитие разработчиков

- Материалы по разработке
  - Документация
  - Учебные материалы
- Взаимодействие с образовательными организациями
  - Открытые лекции
  - Обучение студентов
  - Курсы повышения квалификации (edu@omp.ru)
- Проведение мероприятий для разработчиков
  - Вебинары
  - Митапы
  - Хакатоны

Примеры ПО для ОС Аврора 4.0

1. Самое простое приложение — Application Template
   - Содержит только необходимое
   - Показывает правильную структуру
   - Позволяет увидеть изменения при переходе на новую версию ОС
2. Best practice
   - Позволяют в целом улучшить качество кода
   - Уменьшают количество вопросов по использованию API
3. О чём спрашивают (dev-support@omp.ru)
   - Основные проблемы, с которыми сталкиваются на практике

- Результаты R&D и примеры интеграций

gitlab.com/omprussia/examples

Семестровый курс для вузов

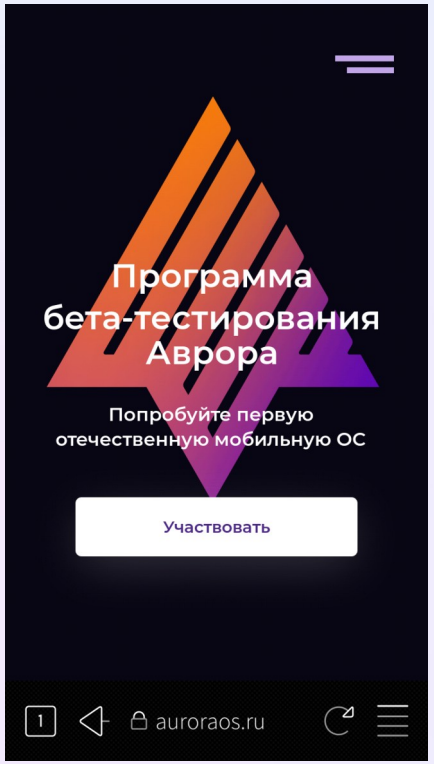
1. Инструменты и технологии
2. Основы QML
3. Продвинутый QML
4. Нативная разработка, экспорт C++ классов
5. Компоненты UI
6. Файлы и базы данных
7. Мультимедиа
8. Карты
9. Сеть
10. Датчики
11. Многопоточность
12. Межпроцессное взаимодействие
13. Создание библиотек
14. Сборка пакета, подготовка к публикации

- Всего 60 ак. часов очных занятий
  - 30 часов лекций
  - 30 часов практики, включая задания и проекты
- Входные требования
  - Навыки разработки (бонус: C++)
  - Понимание алгоритмов и структур данных

АВРОРА

БЕТА ТЕСТИРОВАНИЕ

Программа бета-тестирования ОС Аврора



Программа бета-тестирования ОС Аврора познакомит вас с новыми продуктами компании «Открытая мобильная платформа», позволит провести собственное пользовательское тестирование и даже принять участие в разработке ОС Аврора. Пользователям предоставляется во временное использование мобильное устройство с новой версией ОС, набором предустановленных приложений, доступом в Маркет, SDK и документации.

INOI R7

Масштаб TrustPhone T1

Aquarius NS220

Программа бета-тестирования ОС Аврора

Beta for developers - познакомит вас с SDK и продуктами компании «Открытая мобильная платформа», и предоставит возможность прокачивать свои навыки в разработке под Аврору.

Частным пользователям, энтузиастам и разработчикам предоставляется во временное пользование:

- ✓ Мобильное устройство на ОС Аврора новейшей версии и набором базовых приложений
  - ОС Аврора с режимом разработчика
- ✓ Доступ к Аврора Маркет, куда сможете выкладывать свои приложения
  - Вашими приложениями будет пользоваться сообщество, вы сможете получать обратную связь и обсуждать ее в чатах
- ✓ Platform SDK, и вся необходимая документация
- ✓ Чат с поддержкой
- ✓ А так же сообщество и новые знакомства.

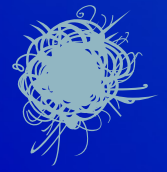
Как принять участие

1. Написать на beta@omp.ru
2. Получить анкету и заполнить ее
3. Дождаться приглашения
4. Подписать документы и вступить в команду

О программе https://auroraos.ru/beta

Ресурсы в интернете

- Портал сообщества разработчиков
  community.omprussia.ru
- Приложения с открытым исходным кодом
  community.omprussia.ru/open_source
- Материалы для разработчиков
  auroraos.ru/developer
- Сервисная поддержка
  разработчиков партнёров
  dev-support@omp.ru


habr.com/ru/company/omprussia


stepik.org/users79671378


www.youtube.com/c/
AuroraOSCommunity


www.youtube.com/cAuroraOS


t.me/omp_ru


vk.com/omp_ru