

# Создание Мобильных приложений Qt Quick на примере ОС Аврора



# Модуль 1. Основы Qt Quick

В этом модуле  
рассматриваются  
следующие темы

- Знакомство с фреймворком Qt и технологией Qt Quick
- Инструменты разработки
- Структура проекта
- Знакомство с языком QML
- Основные визуальные типы
- Позиционирование элементов
- Обработка событий
- Определения новых свойств объектов
- Взаимодействие с пользователем

## Тема 1.3. Структура проекта

Обзор файлов, входящих в проект



Сборка, деплой и запуск проекта



Основы qmake/cmake





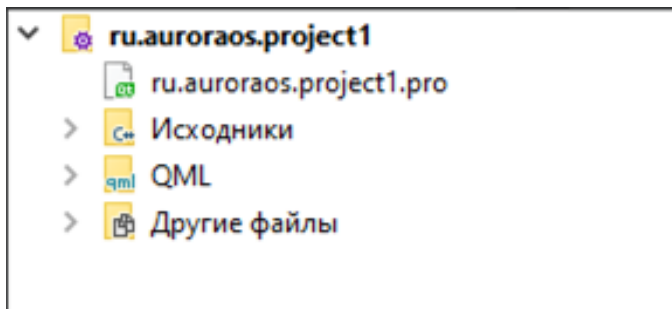
---

# 1. Обзор файлов, входящих в проект

---

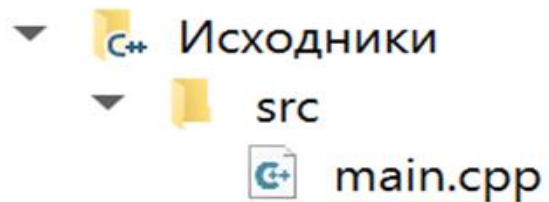


# Обзор файлов, входящих в проект



- Файл с расширением **.pro** – основной сборочный файл
- Директория **Исходники**: в ней находятся файлы с расширением **.cpp**
- Директория **QML**: в ней находятся файлы с расширением **.qml**
- В **Других файлах** находятся иконки программы, изображения, используемые в программе, аудио, видеофрагменты и прочие ресурсы

# Исходники



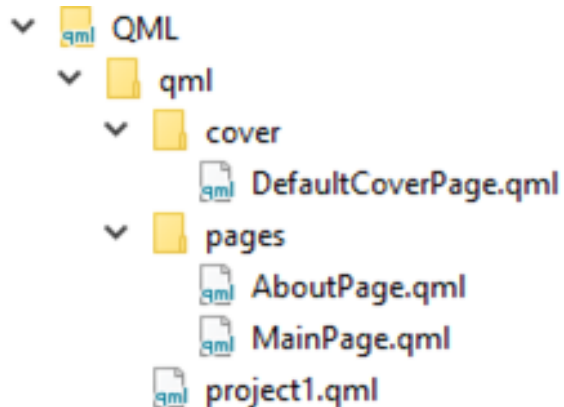
Здесь находится еще одна директория src, в которой находится файл main.cpp

# Исходники

```
/******  
#include <auroraapp.h>  
#include <QtQuick>  
  
int main(int argc, char *argv[])  
{  
    QScopedPointer<QGuiApplication> application(Aurora::Application::application(argc, argv));  
    application->setOrganizationName(QStringLiteral("ru.auroraos"));  
    application->setApplicationName(QStringLiteral("project1"));  
  
    QScopedPointer<QQuickView> view(Aurora::Application::createView());  
    view->setSource(Aurora::Application::pathTo(QStringLiteral("qml/project1.qml")));  
    view->show();  
  
    return application->exec();  
}
```

main.cpp

# QML



Здесь находятся: папка для обложки приложения (cover), файлы которой генерируют отображение запущенной программы в свернутом виде, папка для страниц приложения (pages) и основной файл приложения (**project1.qml**)





# QML

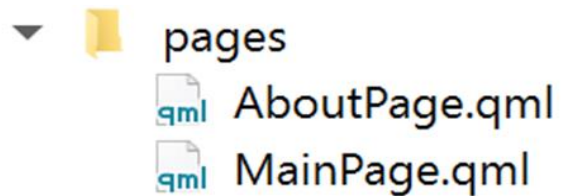
- ★ **objectName** – название приложения
- ★ **initialPage** – прописывается главная страница приложения
- ★ **cover** – прописывается обложка приложения
- ★ **allowedOrientations** – какие ориентации приложения разрешены

```
import QtQuick 2.0
import Sailfish.Silica 1.0

ApplicationWindow {
    objectName: "applicationWindow"
    initialPage: Qt.resolvedUrl("pages/MainPage.qml")
    cover: Qt.resolvedUrl("cover/DefaultCoverPage.qml")
    allowedOrientations: defaultAllowedOrientations
}
```

project1.qml

# QML. Pages



В шаблонном приложении по умолчанию есть файл основной страницы приложения (MainPage.qml) и добавлена еще одна страница – о приложении (AboutPage.qml)















## QML. Cover



Файл, который содержит информацию  
об обложке по умолчанию



## Другие файлы

- ▼  Другие файлы
  - >  icons
  - >  qml\icons
  - ▼  rpm
    -  ru.auroraos.project1.spec
  - >  translations
    -  AUTHORS.md
    -  CODE\_OF\_CONDUCT.md
    -  CONTRIBUTING.md
    -  LICENSE.BSD-3-CLAUSE.md
    -  README.md
    -  ru.auroraos.project1.desktop

В данной директории находятся различные другие папки и файлы, которые используются в приложении



## Файл pro

- ★ Файл с расширением pro – это текстовый файл, который используется для конфигурирования qmake при сборке проекта
- ★ Генерируется автоматически
- ★ Конфигурирует весь проект



## Файл pro.user

- ★ Содержит в себе настройки проекта
- ★ Создается автоматически
- ★ Обычно не требует вмешательства программиста
- ★ При изменении пути проекта этот файл желательно удалить





## Файл spec

- ★ Используется для конфигурирования grm пакета
- ★ Указываются: лицензия, имя файла устанавливаемой программы в виде пакета и команды для сборки, установки и удаления программы





## Файл Desktop

- ★ **Файл Desktop** – это текстовый файл, который используется как ярлык в ОС Аврора для быстрого запуска приложения







---

## 2. Сборка, деплой и запуск проекта

---



## Запуск и отладка проекта

- ★ Пишется на C++/Qt с использованием QML
- ★ Осуществляется в Аврора IDE, основанной на Qt Creator
- ★ Сборка происходит в среде сборки
- ★ Запуск – в эмуляторе или на внешнем устройстве



А В Р О Р А

# Создание или открытие проекта

- ★ Создать новый проект из шаблона
- ★ Создать новый проект из примера
- ★ Открыть существующий проект



А В Р О П А

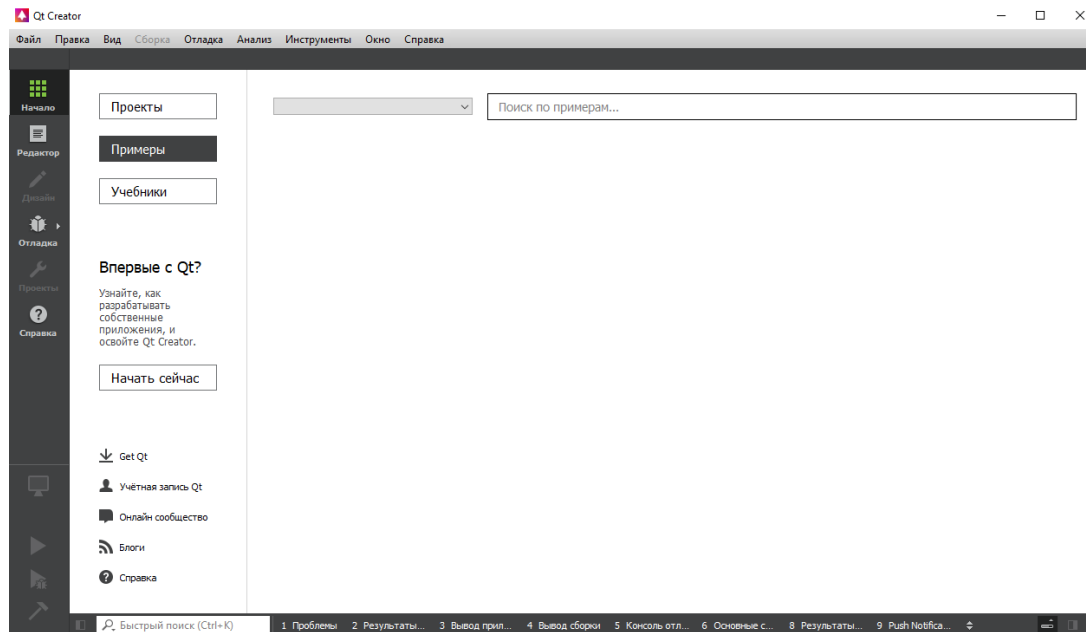
# Создание нового проекта из шаблона

Данный способ позволяет получить простое приложение с графическим интерфейсом с помощью мастера



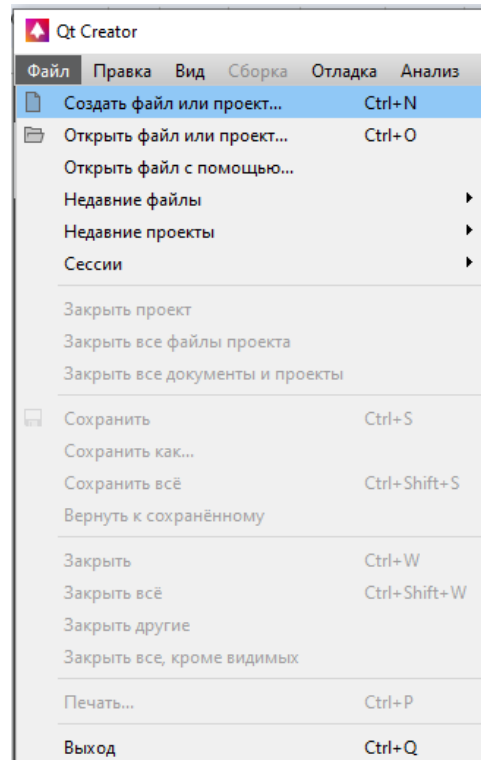
# Создание нового проекта из шаблона

## 1. Запустить Аврора IDE



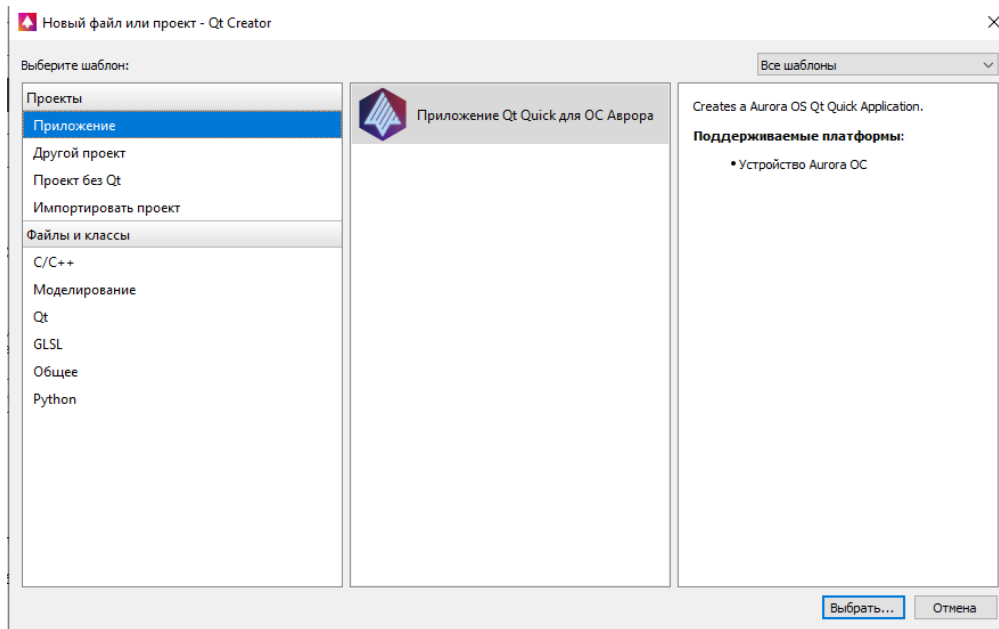
# Создание нового проекта из шаблона

2. В основном окне Аврора IDE выбрать пункт меню  
«Файл» → «Создать файл или проект...»



## Создание нового проекта из шаблона

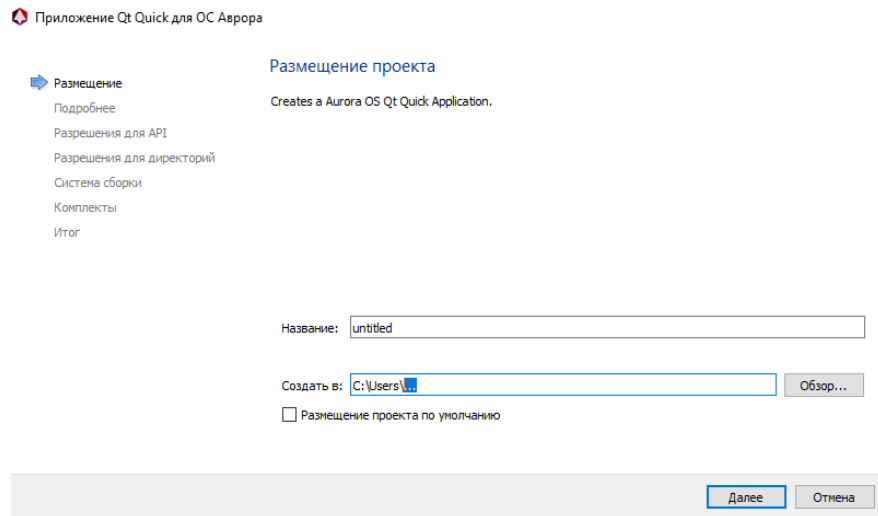
3. В открывшемся окне «Новый файл или проект» выбрать вкладку «Проекты» → «Приложение» и отметить «ОС Аврора SDK Qt Quick Application», после чего нажать кнопку «Выбрать...»



## Создание нового проекта из шаблона

4. В появившемся окне «Введение и размещение проекта» указать имя проекта, директорию и нажать кнопку «Далее»

Проект должен находиться или в домашней директории пользователя, или в альтернативной директории, указанной при установке Аврора SDK



Приложение Qt Quick для ОС Аврора

Размещение проекта  
Creates a Aurora OS Qt Quick Application.

Размещение  
Подробнее  
Разрешения для API  
Разрешения для директорий  
Система сборки  
Комплекты  
Итог

Название: untitled

Создать в: C:\Users\... Обзор...

☐ Размещение проекта по умолчанию

Далее Отмена



# Создание нового проекта из шаблона

5. В следующем окне «Application Details» ввести необходимые данные о приложении и нажать кнопку «Далее»

Приложение Qt Quick для ОС Аврора

Размещение

Подробнее

Разрешения для API

Разрешения для директорий

Система сборки

Комплекты

Итог

Описание приложения

Название организации:

Название приложения:

Название приложения на русском языке:

Краткое описание:

Версия:

Описание:

ru.auroraos

Template

Шаблон

Моё приложение для ОС Аврора

0.1

Короткое описание моего приложения для ОС Аврора

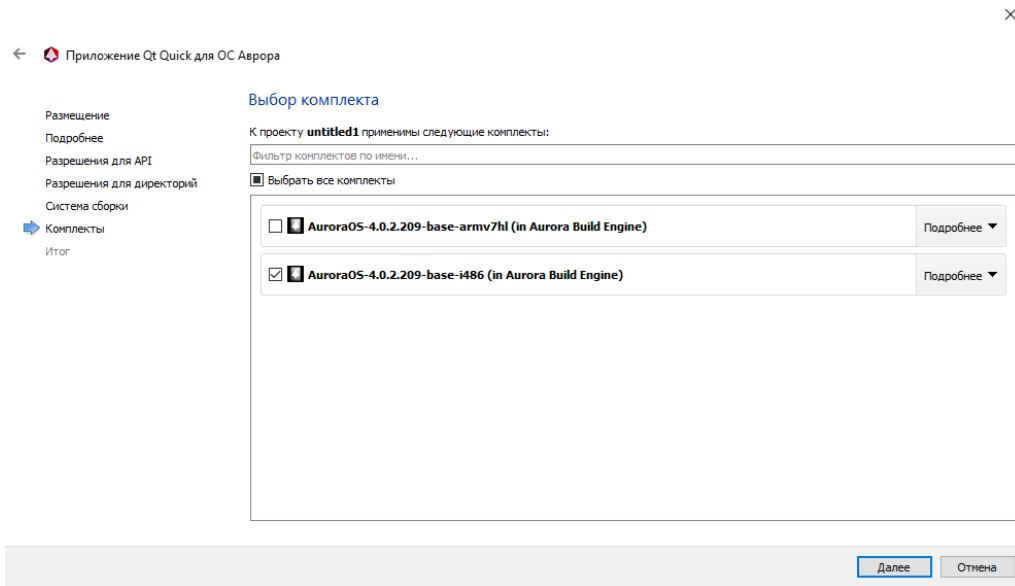
Далее

Отмена

# Создание нового проекта из шаблона

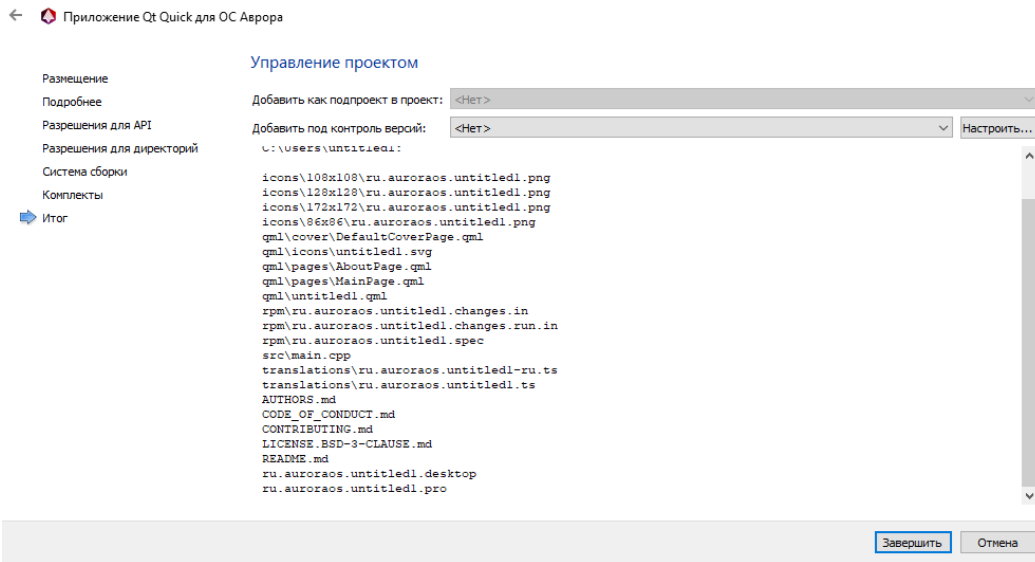
6. В открывшемся окне «Выбор комплекта» выбрать необходимые комплекты для сборки и нажать кнопку «Далее»

- ★ arm7hl – мобильных устройств
- ★ i486 – для эмулятора



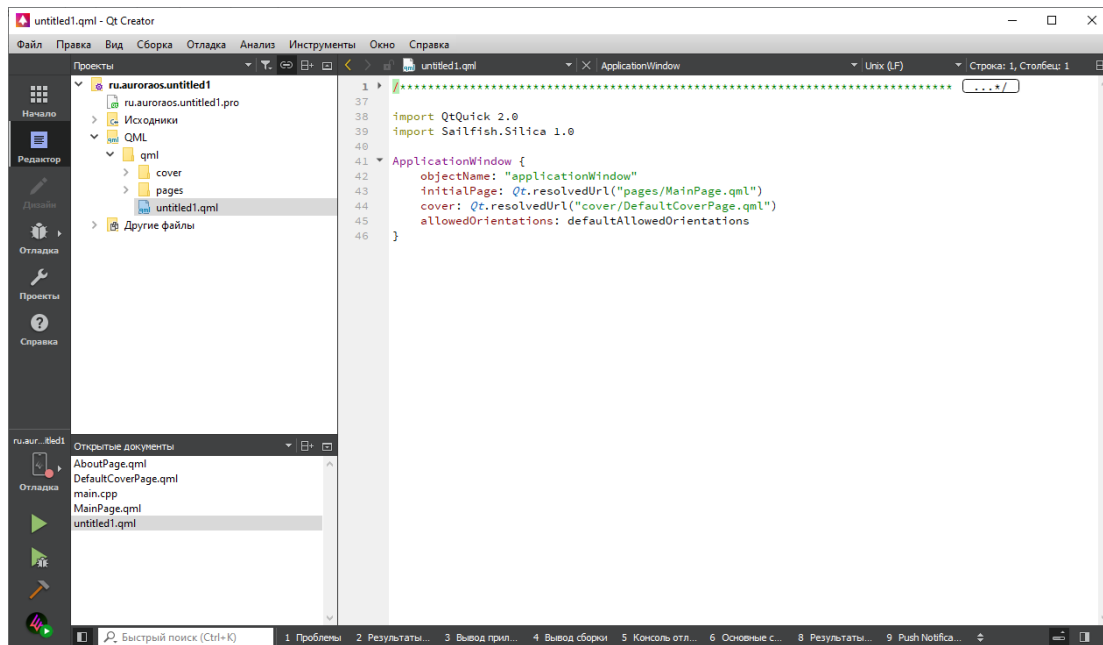
# Создание нового проекта из шаблона

7. В появившемся окне  
«Управление проектом»  
выбрать необходимые данные  
и нажать кнопку «Завершить»



# Создание нового проекта из шаблона

8. В открывшемся редакторе исходного кода можно приступить к работе над проектом



## Создание нового проекта из примера

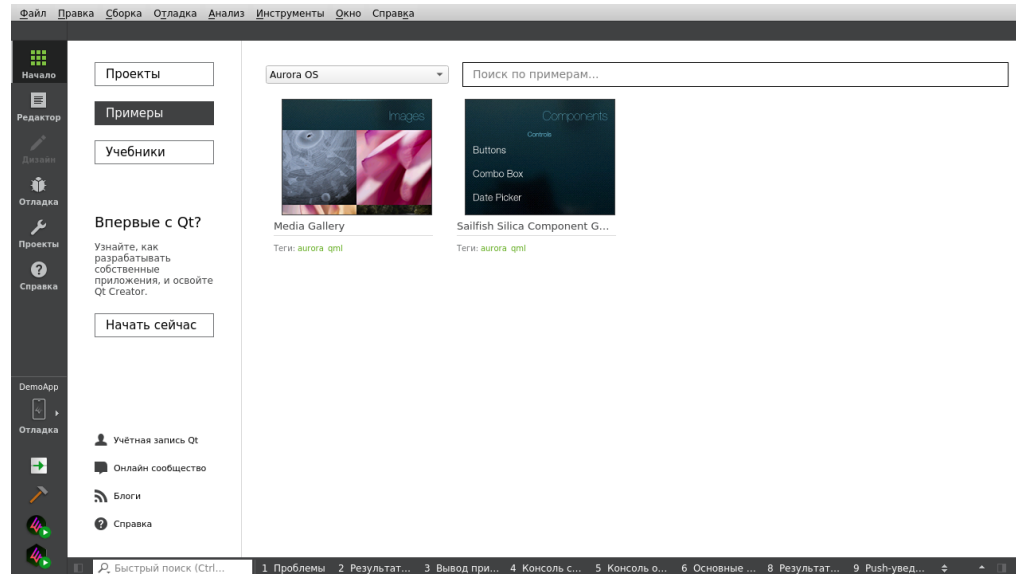
- ★ Данный способ позволяет создать приложение на базе существующего в Аврора SDK примера
- ★ Такой подход удобен для изучения на примерах способов реализации функций, связанных с особенностями разработки под ОС Аврора



А В Р О Р А

# Создание нового проекта из примера

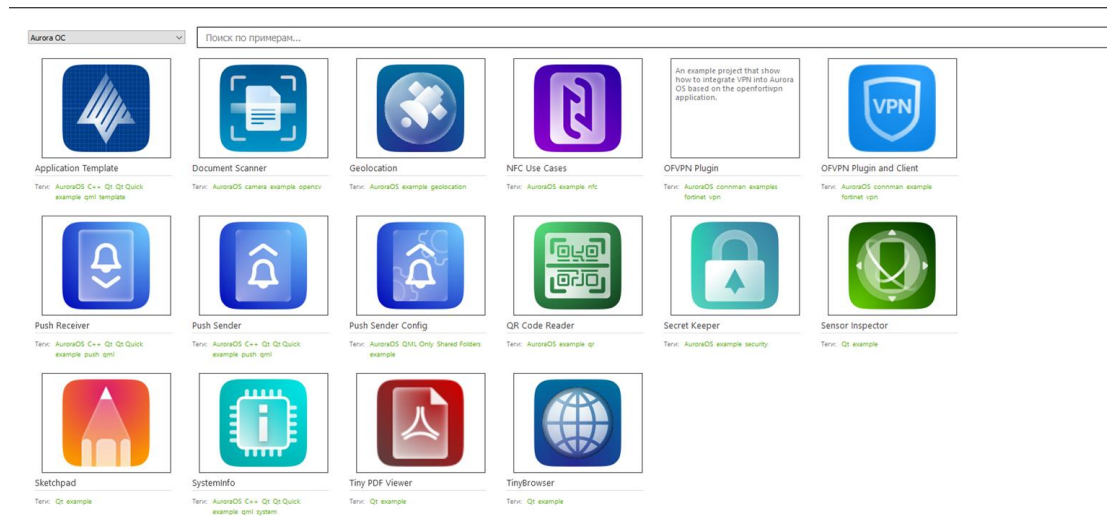
1. Запустить Аврора IDE
2. В основном окне Аврора IDE выбрать пункт «Начало» и нажать кнопку «Примеры»



## Создание нового проекта из примера

3. В открывшейся галерее доступных примеров приложений выбрать интересующий пример и кликнуть правой кнопкой мыши по нему

При наведении курсором мыши на миниатюру примера будет показано его описание



## Создание нового проекта из примера

4. В появившемся окне «Copy Project to writable Location» указать директорию и нажать кнопку «ОК»

В данную директорию будет скопирована папка с файлами примера, которую можно будет модифицировать

The project you are about to open is located in the write-protected location:

/home/user/AuroraOS/examples/mediagallery

Please select a writable location to copy and open the files.

/home/user/Workspace

Обзор...



ОК



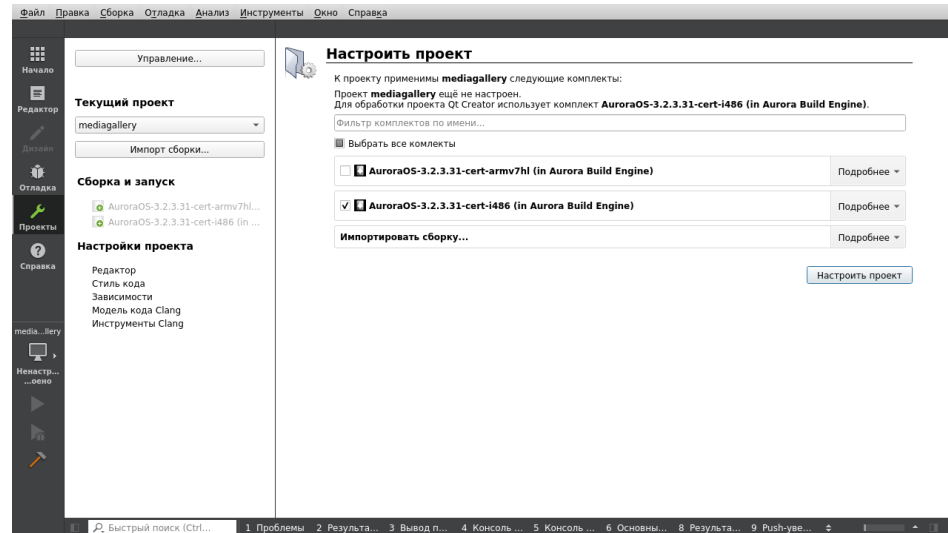
Отмена



## Создание нового проекта из примера

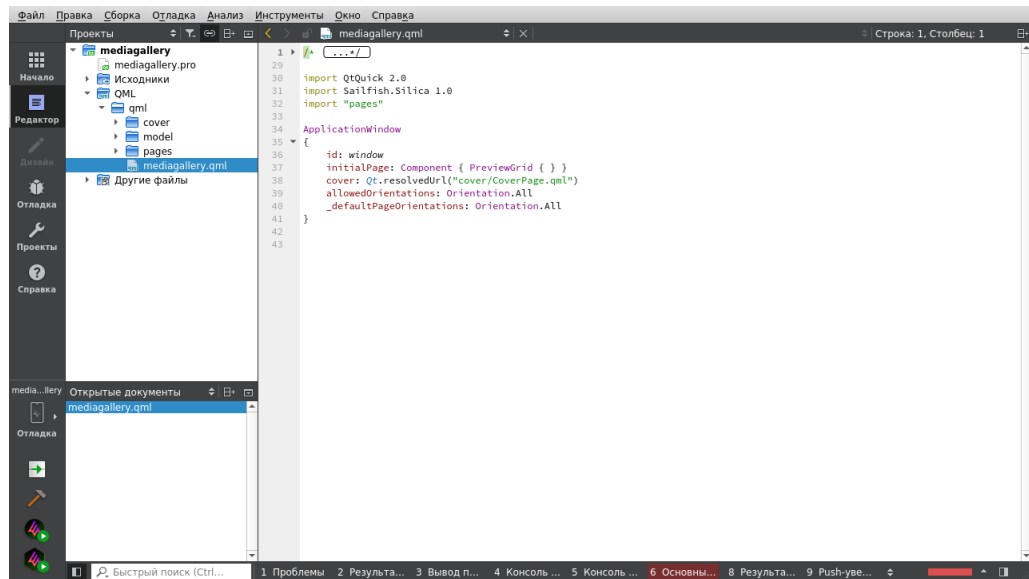
5. В следующем окне выбрать необходимые комплекты для сборки и нажать кнопку «Настроить проект»

Комплект arm7hl используется для мобильных устройств, i486 – для эмулятора



# Создание нового проекта из примера

6. В открывшемся редакторе исходного кода можно приступить к работе над проектом



## Открытие существующего проекта

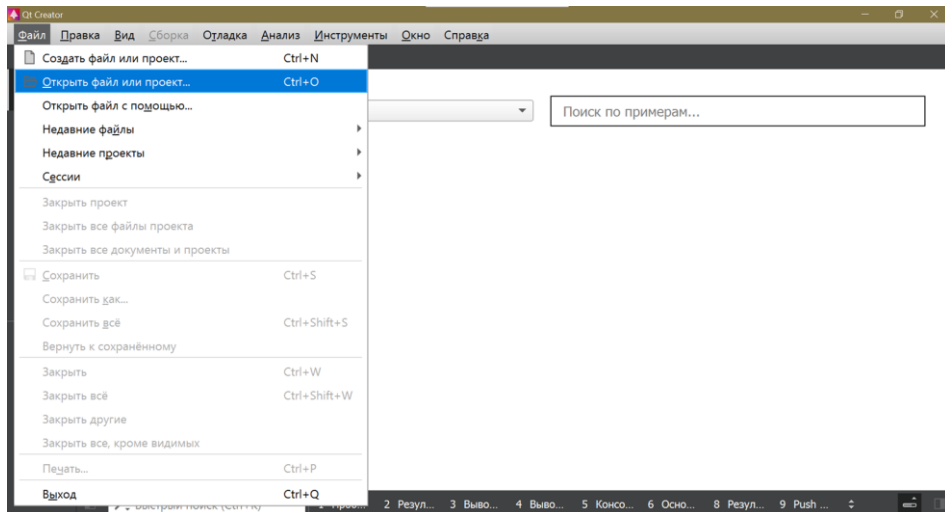
Для проектов приложений, использующих систему сборки qmake, структура проектов для ОС Аврора определяется файлом \*.pro

Для открытия существующего проекта необходимо указывать файл с данным расширением



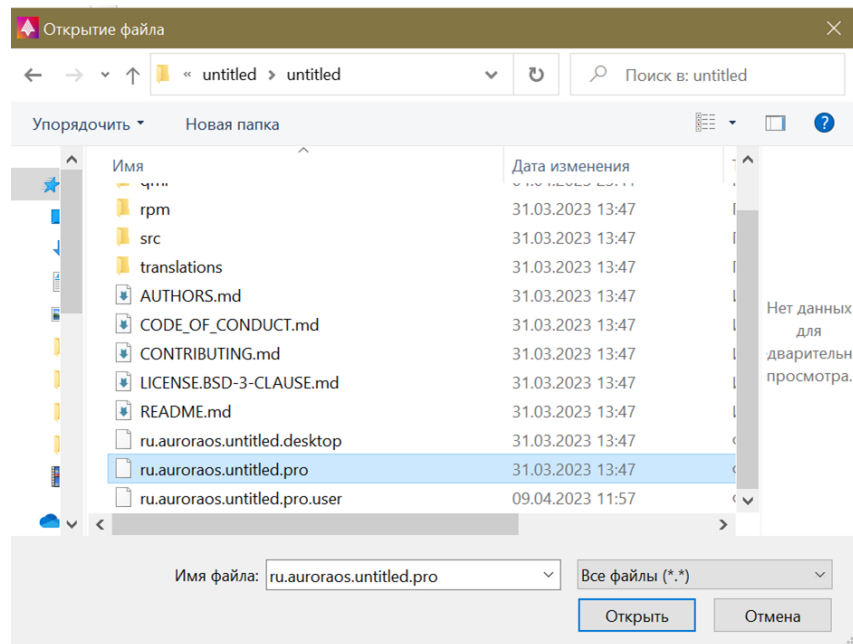
# Открытие существующего проекта

1. Запустить Аврора IDE
2. В основном окне Аврора IDE выбрать пункт меню «Файл» → «Открыть файл или проект...»



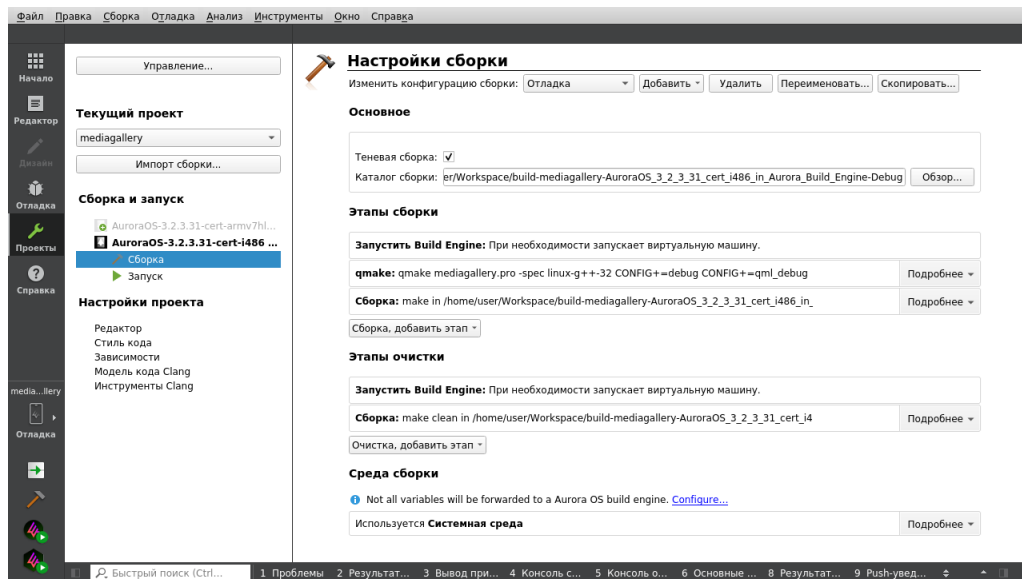
## Открытие существующего проекта

3. В появившемся окне выбрать файл с расширением .pro и нажать кнопку «Открыть»



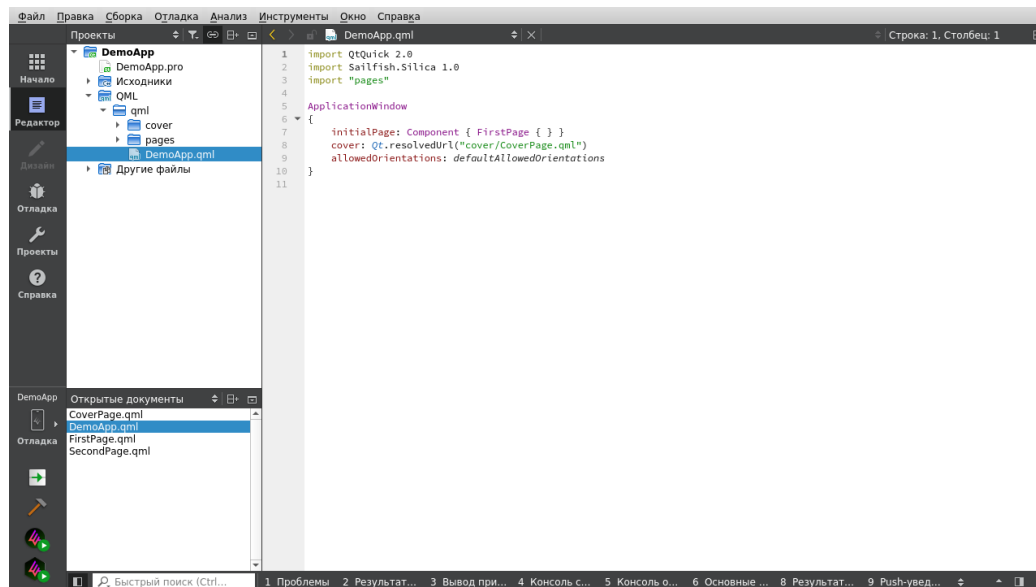
# Открытие существующего проекта

4. Если файл с расширением **\*.user** отсутствует, или он некорректен, в окне Аврора IDE перейти в режим **«Проекты»** и выбрать необходимые комплекты для сборки



# Открытие существующего проекта

5. В окне Аврора IDE в режиме **«Редактор»** можно приступить к работе над проектом



## Сборка проекта

- ★ Предполагается, что в Аврора IDE существует открытый проект
- ★ Используется среда сборки, поэтому независима от ОС
- ★ В среде сборки настроено несколько общих папок для обмена файлами с домашней ОС





А В Р О Р А




## Сборка проекта

1. Запустить среду сборки. Для управления виртуальной машиной в ручном режиме необходимо выполнить следующее:

★ для запуска на панели слева нажать кнопку  **Запуск «Aurora Build Engine»**,  
и дождаться, пока она не примет вид  **Остановка «Aurora Build Engine»**

★ для остановки нажать кнопку  **Остановка «Aurora Build Engine»**

## Сборка проекта

2. На панели слева нажать кнопку  **Опции** сборки и выбрать комплекты и способы сборки. Для эмулятора необходимо выбрать AuroraOS-i486, для мобильных устройств — AuroraOS-armv7hl



А В Р О Р А



## Сборка проекта

- ★ **«Выпуск»** — завершающая сборка пакета для передачи заказчику или пользователю программы
- ★ **«Отладка»** — в сборку пакетов будет добавлена информация для отладки приложения (пошаговое исполнение, наблюдение значений переменных и т. п.)
- ★ **«Профилирование»** — в сборку пакетов будет добавлена информация для профилирования и оптимизации быстродействия работы приложения (вычисление временных затрат на работу отдельных подпрограмм)

# Сборка проекта

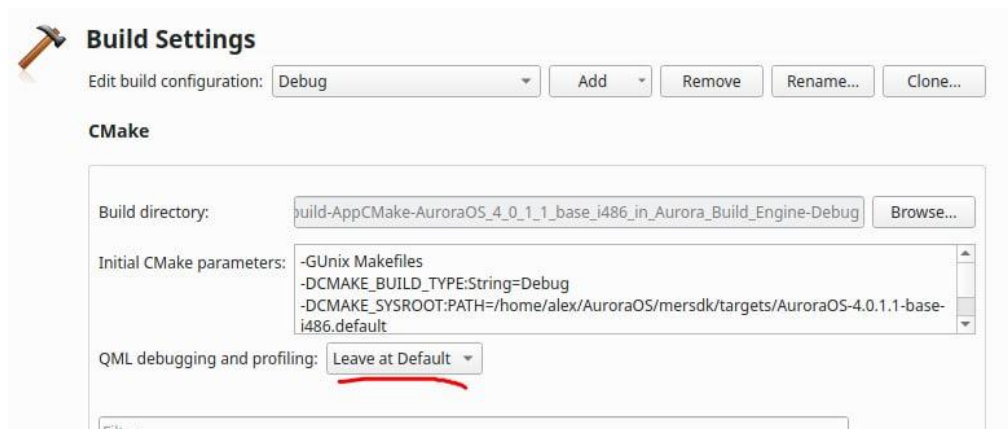


3. После завершения настроек нажать кнопку



**Сборка** проекта для запуска сборки проекта

Для того, чтобы отладка и профилирование проекта были доступны, необходимо у настройки Проекты → Сборка → Отладка и профилирование QML установить значение Enable



## Запуск приложения

Приложение может быть запущено как на внешнем устройстве, работающем под управлением ОС Аврора, так и в эмуляторе, который устанавливается при инсталляции Аврора SDK



А В Р О Р А

# Запуск приложения на эмуляторе

1. На панели слева нажать кнопку



**Опции** запуска и выбрать комплект AuroraOS-i486.



А В Р О Р А

## Запуск приложения на эмуляторе

2. Запустить эмулятор нажатием кнопки



**Запуск** «Aurora Emulator» и дождаться,

пока она не примет вид



**Остановка** «Aurora Emulator». Откроется новое окно VirtualBox, и загрузится эмулятор.

Если нажать кнопку



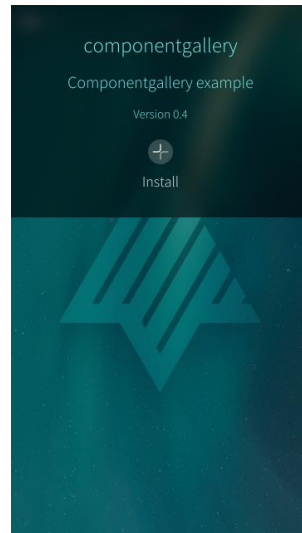
**Остановка** «Aurora Emulator», эмулятор остановится, и окно VirtualBox закроется

## Запуск приложения на эмуляторе

3. Для запуска приложения нажать кнопку «Запустить»

для отладки — кнопку Отладка.

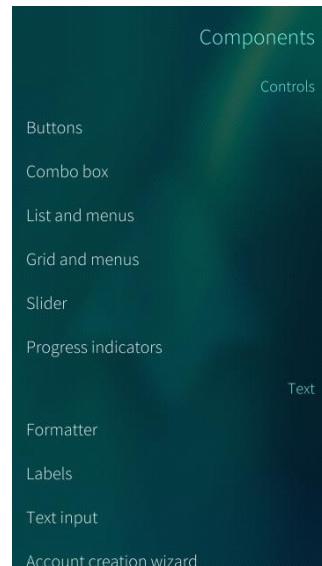
Чтобы кнопки стали активны, необходимо выбрать «Deploy As RPM Package» или «Deploy by Copying Binaries» в панели выбора комплектов и способов сборки





## Запуск приложения на эмуляторе

4. Для установки приложения коснуться значка «Install» на экране эмулятора. Приложение установится, и откроется его стартовая страница



# Запуск приложения на устройстве

Запуск приложения на устройстве происходит аналогичным образом, но дополнительно требуется:

1. В основном окне Аврора IDE выбрать пункт меню «Инструменты» → «Параметры»
2. Перейти на вкладку «Устройства» и подключить устройство
3. Настроить подписание установочного пакета





---

## 3. Основы qmake/stake

---



# qmake

- ★ Инструмент qmake распространяется вместе с набором библиотек Qt начиная с версии 3.0 (2001 г.)
- ★ qmake не является системой построения в строгом понимании этого слова, он всего лишь генерирует файлы описания проектов для других систем
- ★ На вход программе qmake подаётся описание проекта на высокоуровневом языке
- ★ Интегрированная среда Qt Creator использует файлы «\*.pro»
- ★ Язык qmake поддерживает специальные переменные, определяющие тип проекта



# qmake

## Достоинства:

- ★ Простой высокоуровневый язык описания проектов
- ★ Переносимость
- ★ Возможность работы над проектами в средах Visual Studio, XCode, Qt Creator
- ★ Поддержка в генерируемых проектах библиотек Qt
- ★ Простая в использовании поддержка построения вне каталога проекта
- ★ Расширяемость



# qmake

## Недостатки:

- ★ Ориентированность в первую очередь на набор библиотек Qt
- ★ Не предусматривает возможности запуска серии тестов для определения особенностей среды построения
- ★ Не предусмотрены средства для генерирования заголовочных и прочих файлов



# qmake

- ★ **TARGET** – Указывает имя целевого файла
- ★ **CONFIG** – Определяет конфигурацию проекта и параметры компилятора
- ★ **PKGCONFIG** – используется для тонкой настройки поведения инструмента pkg-config
- ★ **SOURCES** – Указывает имена всех исходных файлов в проекте

```
TARGET = ru.auroraos.project1

CONFIG += \
    auroraapp

PKGCONFIG += \

SOURCES += \
    src/main.cpp \

HEADERS += \

DISTFILES += \
    rpm/ru.auroraos.project1.spec \
    AUTHORS.md \
    CODE_OF_CONDUCT.md \
    CONTRIBUTING.md \
    LICENSE.BSD-3-CLAUSE.md \
    README.md \

AURORAAPP_ICONS = 86x86 108x108 128x128 172x172

CONFIG += auroraapp_i18n

TRANSLATIONS += \
    translations/ru.auroraos.project1.ts \
    translations/ru.auroraos.project1-ru.ts \
```

ru.auroraos.project1.pro

# qmake

★ **HEADERS** — Определяет заголовочные файлы для проекта

★ **DISTFILES** — Определяет список файлов, которые должны быть включены в dist

★ **TRANSLATIONS** — Указывает список файлов перевода (.ts)

```
TARGET = ru.auroraos.project1

CONFIG += \
    auroraapp

PKGCONFIG += \

SOURCES += \
    src/main.cpp \

HEADERS += \

DISTFILES += \
    rpm/ru.auroraos.project1.spec \
    AUTHORS.md \
    CODE_OF_CONDUCT.md \
    CONTRIBUTING.md \
    LICENSE.BSD-3-CLAUSE.md \
    README.md \

AURORAAPP_ICONS = 86x86 108x108 128x128 172x172

CONFIG += auroraapp_i18n

TRANSLATIONS += \
    translations/ru.auroraos.project1.ts \
    translations/ru.auroraos.project1-ru.ts \
```

ru.auroraos.project1.pro



# CMake

- ★ CMake является свободным инструментом с открытым исходным кодом, основным разработчиком которого выступает компания Kitware
- ★ Название системы расшифровывается как «cross-platform make»



# CMake

Принцип работы инструмента CMake аналогичен принципу работы qmake:  
из директории исходных кодов считывается файл CMakeLists.txt  
с описанием проекта, на выходе инструмент генерирует файлы проекта  
для одной из множества конечных систем построения



# CMake

## Требования для сборки:

- ★ Наличие утилиты make
- ★ Наличие интерпретатора сценариев на языке bash  
либо скомпилированного инструмента CMake одной из предыдущих версий
- ★ Наличие компилятора C++



# CMake

- ★ Система CMake управляется при помощи универсального процедурного языка
- ★ При помощи CMake легче описывать проекты, которые используют другие библиотеки и инструменты
- ★ При помощи CMake легче решать нестандартные задачи, которые возникают при организации процесса построения



# CMake

## Достоинства:

- ★ Простой интерфейс для подключения библиотек, являющихся результатами построения одних целей к другим
- ★ Команды и сценарии для поиска в системе наборов библиотек
- ★ Средства генерирования исходных файлов и сценариев в процессе



## Итоги

1. Разобрали файлы, входящие в проект
2. Рассмотрели создание, сборку и запуск проекта
3. Познакомились с qmake/cmake
4. Разобрали достоинства qmake/cmake



# Рекомендации

Дубров Д.В. – Система построения  
проектов CMake





## Источники

1. [Структура Qt-проекта](#)
2. [Запуск и отладка проекта | Портал разработчиков ОС Аврора](#)
3. [Qt 5.6](#)
4. [Дубров, Д.В. Система построения проектов CMake: учебник / Д. В. Дубров; Южный федеральный университет. — Ростов-на-Дону: Издательство Южного федерального университета, 2015. — 419 с.](#)



Спасибо за внимание!



А В Р О Р А