УТВЕРЖДЕНО

решением Учёного совета факультета математики, информационных и авиационных технологий

от «21» июня 2019 г., протокол № 5/19

Председатель _____/ М.А. Волков
«21» июня 2019 г.

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ

Дисциплина	Разработка мобильных приложений
Факультет	Факультет математики, информационных и авиационных технологий
Кафедра	Телекоммуникационные технологии и сети
Курс	4

Направление (специальность) 11.03.02 Инфокоммуникационные технологии и системы связи
код направления (специальности), полное наименование

Направленность (профиль/специализация) Интернет и гетерогенные сети
полное наименование

Форма обучения очная
очная, заочная, очно-заочная

Дата введения в учебный процесс УлГУ: «1» секнтября 2019 г.

Программа актуализированана заседании кафедры: протокол №_1_ от 1 сентября 2021 г.
Программа актуализированана заседании кафедры: протокол №_1_ от 1 сентября 2022 г.
Программа актуализированана заседании кафедры: протокол №_1_ от 1 сентября 2023 г.

Сведения о разработчиках:

ФИО	Кафедра	Должность, ученая степень, звание
Булаев Алексей Александрович	ТТС	к.т.н., доцент

СОГЛАСОВАНО	СОГЛАСОВАНО
Заведующий кафедрой телекоммуникационных технологий и сетей, реализующей дисциплину	Заведующий выпускающей кафедрой телекоммуникационных технологий и сетей
(_____/_Смагин А.А ____/ *Подпись* *ФИО* «21» июня 2019 г.	(_____/___Смагин А.А____/ *Подпись* *ФИО* «21» июня 2019 г.

1. ЦЕЛИ И ЗАДАЧИ ОСВОЕНИЯ ДИСЦИПЛИНЫ:

Цели освоения дисциплины: изучение основных проблем, возникающих при разработке приложений для мобильных устройств, а также получение представления о проблемах, стоящих перед разработчиком таких приложений.

Задачи освоения дисциплины: приобретение в рамках освоения предусмотренного курсом занятий следующих знаний, умений и навыков, характеризующих определённый уровень сформированности целевых компетенций (см. подробнее п.3):

знать:

– архитектуры мобильных ОС;
– инструментальные средства разработки, доступные у платформы Android;
– инструментальные средства разработки, доступные у платформы iOS;

уметь:

– применять средства разработки, доступные у платформ Android и iOS
– взаимодействовать с технологиями мобильных устройств

владеть:

– средствами управления доступом мобильных ОС
– навыками написания приложений для мобильных устройств

2. МЕСТО ДИСЦИПЛИНЫ В СТРУКТУРЕ ОПОП:

Дисциплина «Разработка мобильных приложений» относится к числу дисциплин блока ФТД.02, предназначенного для студентов, обучающихся по направлению: 11.03.02 Инфокоммуникационные технологии и системы связи.

Для успешного изучения дисциплины необходимы знания и умения, приобретённые в результате освоения курсов «Технология программирования», «Информационные технологии» и полностью или частично сформированные компетенции ПК-6.

Основные положения дисциплины используются в дальнейшем при изучении таких дисциплин как: «Преддипломная практика».

3. ПЕРЕЧЕНЬ ПЛАНИРУЕМЫХ РЕЗУЛЬТАТОВ ОБУЧЕНИЯ ПО ДИСЦИПЛИНЕ (МОДУЛЮ), СООТНЕСЕННЫХ С ПЛАНИРУЕМЫМИ РЕЗУЛЬТАТАМИ ОСВОЕНИЯ ОСНОВНОЙ ПРОФЕССИОНАЛЬНОЙ ОБРАЗОВАТЕЛЬНОЙ ПРОГРАММЫ

Код и наименование реализуемой компетенции	Перечень планируемых результатов обучения по дисциплине (модулю), соотнесенных с индикаторами достижения компетенций
ПК-6 Способность оценки параметров безопасности и защиты программного обеспечения и сетевых устройств администрируемой сети с помощью специальных средств управления безопасностью	знать: – инструментальные средства разработки, доступные у платформы Android; уметь: – взаимодействовать с технологиями мобильных устройств владеть: – навыками написания приложений для мобильных устройств

Код и наименование реализуемой компетенции	Перечень планируемых результатов обучения по дисциплине (модулю), соотнесенных с индикаторами достижения компетенций
ПК-22 Готовность к подготовке коммерческих предложений, документации, поиску потенциальных клиентов для продажи инфокоммуникационных систем и/или их составляющих, в том числе для торгов, проводящихся по различной форме, запросов предложений от клиентов	знать: – инструментальные средства разработки, доступные у платформы iOS уметь: – применять средства разработки, доступные у платформ Android и iOS владеть: – средствами управления доступом мобильных ОС

4. ОБЩАЯ ТРУДОЕМКОСТЬ ДИСЦИПЛИНЫ

4.1. Объем дисциплины в зачётных единицах (всего) 4

4.2. Объем дисциплины по видам учебной работы (в часах)

Вид учебной работы	Количество часов (форма обучения очная)	
	Всего по плану	В т.ч. по семестрам
		7
1	2	3
Контактная работа обучающихся с преподавателем в соответствии с УП	54	54
Аудиторные занятия:	54	54
Лекции	18	18
Семинары и практические занятия	18	18
Лабораторные работы, практикумы	18	18
Самостоятельная работа	18	18
Форма текущего контроля знаний и контроля самостоятельной работы: тестирование, контр. работа, коллоквиум, реферат и др. (не менее 2 видов)		
Курсовая работа	-	-

Вид учебной работы	Количество часов (форма обучения очная)	
	Всего по плану	В т.ч. по семестрам
		7
1	2	3
Виды промежуточной аттестации (экзамен, зачет)	зачёт	зачёт
Всего часов по дисциплине	72	72

4.3. Содержание дисциплины (модуля.) Распределение часов по темам и видам учебной работы:

Форма обучения очная

Название разделов и тем	Всего	Виды учебных занятий					Форма текущего контроля знаний
		Аудиторные занятия			Занятия в интерактивной форме	Самостоятельная работа	
		Лекции	Практические занятия, семинары	Лабораторные работы, практикумы			
1	2	3	4	5	6	7	
Мобильные операционные системы	13	2	2	2	2	2	-
Основы разработки мобильных приложений	17	3	3	3	2	3	-
Разработка мобильных приложений под Android	17	3	3	3	2	3	-
Разработка мобильных приложений под iOS	17	3	3	3	1	3	-
Разработка графических мобильных приложений	17	3	3	3	2	3	-
Разработка кроссплатформенных мобильных приложений	14	2	2	2	2	2	-
Перспективные технологии в мобильных приложениях	13	2	2	2	2	2	-
Итого	**108**	**18**	**18**	**18**	**13**	**18**	**-**

***В интерактивной форме проводятся все лабораторные работы. Тема и содержание занятия приведены в пункте «ЛАБОРАТОРНЫЕ РАБОТЫ (ЛАБОРАТОРНЫЙ ПРАКТИКУМ). Столбец «Занятия в интерактивной форме» в подсчёте итогов не участвует,*

т.к. дублирует столбец «Лабораторная работа».

5. СОДЕРЖАНИЕ ДИСЦИЛИНЫ (МОДУЛЯ)

Тема 1. Мобильные операционные системы

История развития мобильных операционных систем. Операционные системы Android, iOS, Blackberry OS, Symbian OS, Microsoft Phone.

Тема 2. Основы разработки мобильных приложений

Среды разработки под операционные системы Android и iOS. Android Studio, Eclipse, Apache Cordova, Kotlin, Xamarin, XCode.

Тема 3. Разработка мобильных приложений под Android

Архитектура OS Android. Структура разрабатываемых приложений под Android

Тема 4. Разработка мобильных приложений под iOS

Архитектура OS iOS. Структура разрабатываемых приложений под iOS.

Тема 5. Разработка графических мобильных приложений

Среды разработки графических приложений. Библиотека OpenGL. Среда Unity 3D. Игровой движок Unreal Engine.

Тема 6. Разработка кроссплатформенных мобильных приложений

Технология Apache Cordova. Перенос мобильных приложений из одной ОС в другую.

Тема 7. Перспективные технологии в мобильных приложениях

Технология WiFi Direct для Android. Технология Bluetooth LE. Библиотека MultipeerConnectivity для iOS. Smart TV.

6. ТЕМЫ ПРАКТИЧЕСКИХ И СЕМИНАРСКИХ ЗАНЯТИЙ

Тема 1. Мобильные операционные системы (форма проведения – семинар)

1. История развития мобильных операционных систем.
2. Операционные системы Android, iOS, Blackberry OS, Symbian OS, Microsoft Phone

Тема 2. Основы разработки мобильных приложений (форма проведения – семинар)

1. Среды разработки под операционные системы Android и iOS.
2. Особенности IDE Android Studio, Eclipse, Apache Cordova, Kotlin, Xamarin, XCode

Тема 3. Разработка мобильных приложений под Android (форма проведения – семинар)

1. Архитектура OS Android.
2. Структура разрабатываемых приложений под Android

Тема 4. Разработка мобильных приложений под iOS (форма проведения – семинар)

1. Архитектура OS iOS.
2. Структура разрабатываемых приложений под iOS

Тема 5. Разработка графических мобильных приложений (форма проведения – семинар)

1. Среды разработки графических приложений.
2. Библиотека OpenGL.

3. Среда Unity 3D.
4. Игровой движок Unreal Engine

Тема 6. Разработка кроссплатформенных мобильных приложений (форма проведения – семинар)

1. Технология Apache Cordova.
2. Перенос мобильных приложений из одной ОС в другую

Тема 7. Перспективные технологии в мобильных приложениях (форма проведения – семинар)

1. Технология WiFi Direct для Android.
2. Технология Bluetooth LE.
3. Библиотека MultipeerConnectivity для iOS.
4. Smart TV

7. ЛАБОРАТОРНЫЕ РАБОТЫ, ПРАКТИКУМЫ

Лабораторная работа №1. Компоновка и позиционирование элементов.
Цель: Научиться работать с элементами управления и создавать страницы.
Задачи:

- Создать новый проект.
- Изучить разметку.
- Изучить расположение элементов на экране.
- Изучить новые элементы управления.
- Наполнить приложение элементами управления с использованием
- компоновки.

В данной лабораторной работе показывается, как можно располагать элементы на экране приложения с помощью позиционирования. В лабораторной работе идет описание элемента «Grid» и его свойств. Описывается, как добавить текст с применением стилей. Также описывается создание нового проекта и добавление такого элемента как «Pivot», за счет которого страница приложения получает вид страницы с закладками. Также идет описание добавления новых элементов управления.
Результатом выполнения данной работы является приложение с титулом, закладками, задачами и элементами управления «CheckBox», которые отвечают за выполнение или невыполнение какой-либо задачи. Задачи в приложении можно менять

Лабораторная работа №2. Подключение локальной или удаленной базы данных к мобильному приложению.
Цель: Научиться подключать базу данных SQLite к мобильному приложению в OS Android.

В Android имеется встроенная поддержка одной из распространенных систем управления базами данных - SQLite. Для этого в пакете android.database.sqlite определен набор классов, которые позволяют работать с базами данных SQLite. И каждое приложение может создать свою базу данных.
Чтобы использовать SQLite в Android, надо создать базу данных с помощью выражение на языке SQL. После этого база данных будет храниться в каталоге прило-

жения по пути:

DATA/data/[Название_приложения]/databases/[Название_файла_базы_данных]

ОС Android по умолчанию уже содержит ряд встроенных бад SQLite, которые используются стандартными программами - для списка контактов, для хранения фотографий с камеры, музыкальных альбомов и т.д.

Основную функциональность по работе с базами данных предоставляет пакет android.database. Функциональность непосредственно для работы с SQLite находится в пакете android.database.sqlite.

База данных в SQLite представлена классом android.database.sqlite.SQLiteDatabase. Он позволяет выполнять запросы к бд, выполнять с ней различные манипуляции.

Класс android.database.sqlite.SQLiteCursor предоставляет запрос и позволяет возвращать набор строк, которые соответствуют этому запросу.

Класс android.database.sqlite.SQLiteQueryBuilder позволяет создавать SQL-запросы.

Сами sql-выражения представлены классом android.database.sqlite.SQLiteStatement, которые позволяют с помощью плейсхолдеров вставлять в выражения динамические данные.

Класс android.database.sqlite.SQLiteOpenHelper позволяет создать базу данных со всеми таблицами, если их еще не существует.

В SQLite применяется следующая система типов данных:

- INTEGER: представляет целое число, аналог типу int в java
- REAL: представляет число с плавающей точкой, аналог float и double в java
- TEXT: представляет набор символов, аналог String и char в java
- BLOB: представляет массив бинарных данных, например, изображение, аналог типу int в java

Сохраняемые данные должны представлять соответствующие типы в java.

Создание и открытие базы данных

Для создания или открытия новой базы данных из кода Activity в Android мы можем вызвать метод openOrCreateDatabase(). Этот метод может принимать три параметра:

- название для базы данных
- числовое значение, которое определяет режим работы (как правило, в виде константы MODE_PRIVATE)
- необязательный параметр в виде объекта SQLiteDatabase.CursorFactory, который представляет фабрику создания курсора для работы с бд

Для выполнения запроса к базе данных можно использовать метод execSQL класса SQLiteDatabase. В этот метод передается SQL-выражение.

Если необходимо не просто выполнить выражение, но и получить из бд какие-либо данные, то используется метод rawQuery(). Этот метод в качестве параметра принимает SQL-выражение, а также набор значений для выражения sql.

Метод db.rawQuery() возвращает объект Cursor, с помощью которого мы можем извлечь полученные данные.

Возможна ситуация, когда в базе данных не будет объектов, и для этого методом query.moveToFirst() пытаемся переместиться к первому объекту, полученному из бд. Если этот метод возвратит значение false, значит запрос не получил никаких данных из бд.

Лабораторная работа №1. Разработка графического мобильного приложения.
Цель: Научиться разрабатывать графическое мобильное приложение с несколькими формами с возможностью перехода по ним и отображением информации из базы данных.

Приложения для Android пишутся на языке программирования Java. Инструменты Android SDK (Software Development Kit – комплект разработки программного обеспечения) компилируют написанный вами код — и все требуемые файлы данных и ресурсов — в файл APK – *программный пакет Android*, который представляет собой файл архива с расширением .apk. В файле APK находится все, что требуется для работы Android-приложения, и он позволяет установить приложение на любом устройстве под управлением системы Android.

Каждое приложение Android, установленное на устройстве, работает в собственной "песочнице" (изолированной программной среде):

- операционная система Android представляет собой многопользовательскую систему Linux, в которой каждое приложение является отдельным пользователем;
- по умолчанию система назначает каждому приложению уникальный идентификатор пользователя Linux (этот идентификатор используется только системой и неизвестен приложению); система устанавливает полномочия для всех файлов в приложении, с тем чтобы доступ к ним был разрешен только пользователю с идентификатором, назначенным этому приложению;
- у каждого процесса имеется собственная виртуальная машина (ВМ), так что код приложения выполняется изолированно от других приложений;
- по умолчанию каждое приложение выполняется в собственном процессе Linux. Android запускает процесс, когда требуется выполнить какой-либо компонент приложения, а затем завершает процесс, когда он больше не нужен либо когда системе требуется освободить память для других приложений.

Таким образом система Android реализует *принцип предоставления минимальных прав*. То есть каждое приложение по умолчанию имеет доступ только к тем компонентам, которые ему необходимы для работы, и ни к каким другим. Благодаря этому формируется исключительно безопасная среда, в которой приложение не имеет доступа к недозволенным областям системы.

Однако у приложения есть варианты предоставления своих данных другим приложениям и доступа к системным службам:

- двум приложениям можно назначить один идентификатор пользователя Linux. В этом случае каждый из них сможет обращаться к файлам другого приложения. Для экономии ресурсов системы также можно сделать так, чтобы приложения с одинаковым идентификатором пользователя выполнялись в одном процессе Linux и использовали одну ВМ (приложения также должны быть подписаны одним сертификатом);
- приложение может запросить разрешение на доступ к данным устройства, например к контактам пользователя, SMS-сообщениям, подключаемой карте

памяти (SD-карте), камере, Bluetooth и др. Все разрешения должны предоставляться приложению при его установке.

Это основные сведения о том, каким образом приложение Android существует в системе. В остальной части этого документа раскрываются следующие темы:

- базовые компоненты, которые определяют приложение;
- файл манифеста, в котором объявляются компоненты и функции устройства, необходимые для приложения;
- ресурсы, которые существуют отдельно от кода приложения и позволяют приложению адаптировать свою работу к устройствам с различными конфигурациями.

Компоненты приложения

Компоненты приложения являются кирпичиками, из которых состоит приложение для Android. Каждый компонент представляет собой отдельную точку, через которую система может войти в приложение. Не все компоненты являются точками входа для пользователя, а некоторые из них зависят друг от друга. При этом каждый компонент является самостоятельной структурной единицей и играет определенную роль — каждый из них представляет собой уникальный элемент структуры, который определяет работу приложения в целом.

Компоненты приложения можно отнести к одному из четырех типов. Компоненты каждого типа предназначены для определенной цели, они имеют собственный жизненный цикл, который определяет способ создания и прекращения существования компонента.

Операции

Операция (Activity) *представляет* собой один экран с пользовательским интерфейсом. Например, в приложении для работы с электронной почтой одна операция может служить для отображения списка новых сообщений, другая – для составления сообщения и третья операция – для чтения сообщений. Несмотря на то что операции совместно формируют связное взаимодействие пользователя с приложением по работе с электронной почтой, каждая из них не зависит от других операций. Любые из этих операций могут быть запущены другим приложением (если это позволяет приложение по работе с электронной почтой). Например, приложение для камеры может запустить операцию в приложении по работе с электронной почтой, которая составляет новое сообщение, чтобы пользователь мог отослать фотографию.

Операция относится к подклассу класса Activity. Подробные сведения об этом можно найти в руководстве для разработчиков в статье Операции .

Службы

Служба (Service) *представляет* собой компонент, который работает в фоновом режиме и выполняет длительные операции, связанные с работой удаленных процессов. Служба не имеет пользовательского интерфейса. Например, она может воспроизводить музыку в фоновом режиме, пока пользователь работает в другом приложении, или же она может получать данные по сети, не блокируя взаимодействие пользователя с операцией. Служба может быть запущена другим компонентом, который затем будут взаимодействовать с ней, – например операцией.

Служба относится к подклассу класса Service. Подробные сведения об этом можно найти в руководстве для разработчиков в статье Службы .

Поставщики контента

Поставщик *контента (Content provider)* управляет общим набором данных приложения. Данные можно хранить в файловой системе, базе данных SQLite, в Интернете или любом другом постоянном месте хранения, к которому у вашего приложения имеется доступ. Посредством поставщика контента другие приложения могут запрашивать или даже изменять данные (если поставщик контента позволяет делать это). Например, в системе Android есть поставщик контента, который управляет информацией контактов пользователя. Любое приложение, получившее соответствующие разрешения, может запросить часть этого поставщика контента (например ContactsContract.Data), для чтения и записи сведений об определенном человеке.

Поставщики контента также используются для чтения и записи данных, доступ к которым внешним компонентам приложение не предоставляет. Например, в образце приложения Note Pad с помощью поставщика контента выполняется сохранение заметок.

Поставщик контента относится к подклассу класса ContentProvider. Он должен реализовывать стандартный набор API-интерфейсов, с помощью которых другие приложения будут выполнять транзакции. Подробные сведения можно найти в руководстве для разработчиков в статье Поставщики контента .

Приемники широковещательных сообщений

Приемник широковещательных сообщений (Broadcast receiver) *представляет* собой компонент, который реагирует на объявления распространяемые по всей системе. Многие из этих объявлений рассылает система — например объявление о том, что экран выключился, аккумулятор разряжен или был сделан фотоснимок. Объявления также могут рассылаться приложениями, — например, чтобы сообщить другим приложениям о том, что какие-то данные были загружены на устройство и теперь готовы для использования. Несмотря на то что приемники широковещательных сообщений не имеют пользовательского интерфейса, они могутсоздавать уведомления в строке состояния, чтобы предупредить пользователя о событии "рассылка объявления". Однако чаще всего они являются просто "шлюзом" для других компонентов и предназначены для выполнения минимального объема работы. Например , они могут инициировать выполнение службой определенных действий при возникновении события.

Приемник широковещательных сообщений относится к подклассу класса BroadcastReceiver , а каждое такое сообщение предоставляется как объект Intent. Подробные сведения изложены в руководстве, посвященном классу BroadcastReceiver.

Уникальной особенностью системы Android является то, что любое приложение может запустить компонент другого приложения. Например, если вы хотите дать пользователю возможность фотографировать, используя камеру устройства, то, поскольку наверняка имеется другое приложение, которое может выполнить это действие, вместо того чтобы разработать операцию фотографирования в своем приложении, вы можете вызвать такое приложение. Вам не нужно внедрять код из приложения для камеры или даже устанавливать на него ссылку. Вместо этого вы можете просто запустить операцию фотографирования из приложения для камеры. По завершении этой операции фотография будет возвращена в ваше приложение, и ее можно будет использовать. Для пользователя это будет выглядеть как одно приложение.

Когда система запускает компонент, она запускает процесс для этого приложения

(если он еще не был запущен) и создает экземпляры классов, которые требуются этому компоненту. Например, если ваше приложение запустит операцию фотографирования в приложении для камеры, эта операция будет выполняться в процессе, который относится к этому стороннему приложению, а не в процессе вашего приложения. Поэтому, в отличие от приложений для большинства других систем, в приложениях для Android отсутствует единая точка входа (например, в них нет функции main()).

Поскольку система выполняет каждое приложение в отдельном процессе с такими правами доступа к файлам, которые ограничивают доступ в другие приложения, ваше приложение не может напрямую вызвать компонент из другого приложения. Это может сделать сама система Android. Поэтому, чтобы вызвать компонент в другом приложении, необходимо сообщить системе о своем намерении (*Intent*) запустить определенный компонент. После этого система активирует для вас этот компонент.

- Активация компонентов

  Компоненты трех из четырех возможных типов — операции, службы и приемники широковещательных сообщений — активируются асинхронным сообщением, которое называется *Intent* (намерение). Объекты Intent связывают друг с другом отдельные компоненты во время выполнения, будь то это компоненты вашего или стороннего приложения (эти объекты Intent можно представить себе в виде мессенджеров, которые посылают другим компонентам запрос на выполнение действий).

Объект Intent создается с помощью объекта Intent, который описывает запрос на активацию либо конкретного компонента, либо компонента конкретного *типа* — соответственно, намерение Intent может быть явным или неявным.

Для операций и служб Объект Intent определяет действие, которое требуется выполнить (например, просмотреть (view) или отправить (send) что-то), а также может указывать URI (Uniform Resource Identifier – унифицированный идентификатор ресурса) данных, с которыми это действие нужно выполнить (помимо прочих сведений, которые нужно знать запускаемому компоненту). Например, объект Intent может передавать запрос на выполнение операции "показать изображение" или "открыть веб-страницу". В некоторых ситуациях операцию можно запустить, чтобы получить результат. В этом случае операция возвращает результат также в виде объекта Intent (например, можно отправить сообщение Intent, чтобы дать пользователю возможность выбрать контакт и вернуть его вам — в ответном сообщении Intent будет содержаться URI, указывающий на выбранный контакт).

Для приемников широковещательных сообщений Intent просто определяет передаваемое объявление (например, широковещательное сообщение о низком уровне заряда аккумулятора содержит только строку "аккумулятор разряжен").

Компоненты четвертого типа – поставщики контента – сообщениями Intent не активируются. Они активируются по запросу от ContentResolver. Процедура определения контента (content resolver) обрабатывает все прямые транзакции с поставщиком контента, с тем чтобы этого не пришлось делать компоненту, который выполняет транзакции с поставщиком. Вместо этого он вызывает методы для объекта ContentResolver. Это формирует слой, абстрагирующий (в целях безопасности) поставщика контента от компонента, запрашивающего информацию.

Для активации компонентов каждого типа имеются отдельные методы:

- Можно запустить операцию (или определить для нее какое-то новое действие), передав объект Intent методу startActivity() или startActivityForResult() (если

требуется, чтобы операция вернула результат).

- Можно запустить службу (либо выдать работающей службе новые инструкции), передав объект Intent методу startService(). Либо можно установить привязку к службе, передав объектIntent методу bindService().
- Можно инициировать рассылку сообщений, передав объект Intent таким методам, как sendBroadcast(), sendOrderedBroadcast() и sendStickyBroadcast().
- Можно выполнить запрос к поставщику контента, вызвав метод query() для объекта ContentResolver.

  Подробные сведения об использовании объектов Intent приведены в документе Объекты Intent и фильтры объектов Intent. Более подробная информация об активации определенных компонентов также приведена в следующих документах: Операции, Службы, BroadcastReceiver и Поставщики контента.
- Файл манифеста

  Для запуска компонента приложения системе Android необходимо знать, что компонент существует. Для этого она читает файл AndroidManifest.xml приложения (файл манифеста). В этом файле, который должен находиться в корневой папке приложения, должны быть объявлены все компоненты приложения.

  Помимо объявления компонентов приложения, манифест служит и для других целей, среди которых:
- указание всех полномочий пользователя, которые требуются приложению, например разрешения на доступ в Интернет или на чтение контактов пользователя;
- объявление минимальногоуровня API, требуемого приложению, с учетом того, какие API-интерфейсы оно использует;
- объявление аппаратных и программных функций, которые нужны приложению или используются им, например камеры, службы Bluetooth или сенсорного экрана;
- указание библиотек API, с которыми необходимо связать приложение (отличные от API-интерфейсов платформы Android), например библиотеки Google Maps ;
- и многое другое.
- Объявление компонентов

  Основная задача манифеста – это информировать систему о компонентах приложения. Например, файл манифеста может объявлять операцию следующим образом:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest ... >
   <application android:icon="@drawable/app_icon.png" ... >
      <activity android:name="com.example.project.ExampleActivity"
               android:label="@string/example_label" ... >
      </activity>
      ...
   </application>
</manifest>
```

  Атрибут android:icon в элементе <application> указывает на ресурсы для значка, который обозначает приложение.

  Атрибут android:name в элементе <activity> указывает полное имя класса подкласса Activity, а атрибут android:label указывает строку, которую необходимо использовать в качестве метки операции, отображаемой для пользователя.

Все компоненты приложения необходимо объявлять следующим образом:

- элементы <activity> для операций;

- элементы <service> для служб;
- элементы <receiver> для приемников широковещательных сообщений;
- элементы <provider> для поставщиков контента
  Системе не видны операции, службы и поставщики контента, которые имеются в исходном коде, но не объявлены в манифесте, поэтому они не могут быть запущены. А вот приемники широковещательных сообщений можно либо объявить в манифесте, либо создать динамически в коде (как объекты BroadcastReceiver) и зарегистрировать в системе путем вызова registerReceiver().
- Объявление возможностей компонентов
  Как уже говорилось в разделе Активация компонентов, с помощью объекта Intent можно запускать операции, службы и приемники широковещательных сообщений. Для этого в объекте Intent следует явно указать имя целевого компонента (с помощью имени класса компонента). Однако в полной мере возможности объектов Intent раскрываются при использовании концепции *неявных Intent*. В неявном сообщении Intent просто описывается тип действия, которое требуется выполнить (а также, хотя это и не обязательно, данные, в отношении которых вы бы хотели выполнить это действие). Системе же предоставляется возможности найти на устройстве компонент, который может выполнить это действие, и запустить его. При наличии нескольких компонентов, которые могут выполнить действие, описанное в сообщении Intent, пользователь выбирает, какой из них будет использоваться.
  Система определяет компоненты, которые могут ответить на сообщение Intent, путем сравнения полученного сообщения Intent с *фильтрами объектов Intent,* указанными в файле манифеста других приложений, имеющихся на устройстве.

При объявлении операции в манифесте своего приложения по желанию можно указать фильтры объектов Intent, которые указывают возможности операции, с тем чтобы она могла реагировать на сообщения Intent от других приложений. Чтобы объявить фильтр Intent для своего компонента, необходимо добавить элемент <intent-filter> в качестве дочернего для элемента объявления компонента.

Например, если вы создали приложение для работы с электронной почтой с операцией составления нового сообщения, вы можете объявить фильтр для ответа на сообщения Intent типа "send" (для отправки нового сообщения электронной почты) следующим образом:

```
<manifest ... >
  ...
  <application ... >
    <activity android:name="com.example.project.ComposeEmailActivity">
      <intent-filter>
        <action android:name="android.intent.action.SEND" />
        <data android:type="*/*" />
        <category android:name="android.intent.category.DEFAULT" />
      </intent-filter>
    </activity>
  </application>
</manifest>
```

Затем, если другое приложение создаст объект Intent с действием ACTION_SEND и передаст его в startActivity(), система сможет запустить вашу операцию, дав пользователю

возможность написать и отправить сообщение электронной почты.

- Объявление требований приложения

  Существует огромное количество устройств, работающих под управлением Android, и не все они имеют одинаковые функциональные возможности. Чтобы ваше приложение не могло быть установлено на устройствах, в которых отсутствуют функции, необходимые приложению, важно четко определить профиль для типов устройств, поддерживаемых вашим приложением, указав требования к аппаратному и программному обеспечению в файле манифеста. Эти объявления по большей части носят информационный характер, система их не читает. Однако их читают внешние службы, например Google Play, с целью обеспечения фильтрации для пользователей, которые ищут приложения для своих устройств.

  Например, если вашему приложению требуется камера и оно использует API-интерфейсы из Android 2.1 (уровень API 7), эти параметры следует объявить в файле манифеста в качестве требований следующим образом:

```
<manifest ... >
    <uses-feature android:name="android.hardware.camera.any"
                  android:required="true" />
    <uses-sdk android:minSdkVersion="7" android:targetSdkVersion="19" />
    ...
</manifest>
```

  Теперь ваше приложение нельзя будет установить из Google Play на устройствах, в которых *нет* камеры, а также на устройствах, работающих под управлением Android версии *ниже* 2.1.

  Однако можно также объявить, что приложение использует камеру, но для его работы она не является *непременно необходимой*. В этом случае в приложении атрибуту required необходимо задать значение "false", а во время работы оно должно проверять, имеется ли на устройстве камера, и при необходимости отключать свои функции, которые используют камеру.

  Более подробные сведения об управлении совместимостью своего приложения с различными устройствами приведены в документе Совместимость устройств .

- Ресурсы приложения

  Приложение Android состоит не только из кода — ему необходимы такие существующие отдельно от исходного кода ресурсы, как изображения, аудиофайлы и все, что связано с визуальным представлением приложения. Например, необходимо определять анимацию, меню, стили, цвета и макет пользовательских интерфейсов операций в файлах XML. Используя ресурсы приложения, можно без труда изменять его различные характеристики, не меняя код, а, кроме того, — путем предоставления наборов альтернативных ресурсов — можно оптимизировать свое приложение для работы с различными конфигурациями устройств (например, для различных языков или размеров экрана).

  Для каждого ресурса, включаемого в проект Android, инструменты SDK задают уникальный целочисленный идентификатор, который может использоваться, чтобы сослаться на ресурс из кода приложения или из других ресурсов, определенных в XML. Например, если в вашем приложении имеется файл изображения с именем logo.png (сохраненный в папке res/drawable/), инструменты SDK сформируют идентификатор ресурса под именем R.drawable.logo, с помощью которого на изображение можно будет ссылаться и вставлять его в

пользовательский интерфейс.
Один из наиболее важных аспектов предоставления ресурсов отдельно от исходного кода заключается в возможности использовать альтернативные ресурсы для различных конфигураций устройств. Например, определив строки пользовательского интерфейса в XML, вы сможете перевести их на другие языки и сохранить эти переводы в отдельных файлах. Затем по *квалификатору* языка , добавленному к имени каталога ресурса (скажем res/values-fr/ для строк на французском языке), и выбранному пользователем языку система Android применит к вашему пользовательскому интерфейсу строки на соответствующем языке.
Android поддерживает разные *квалификаторы* для соответствующих ресурсов. Квалификатор представляет собой короткую строку, которая включается в имена каталогов ресурсов с целью определения конфигурации устройства, для которой эти ресурсы следует использовать. В качестве другого примера можно сказать, что для своих операций следует создавать разные макеты, которые будут соответствовать размеру и ориентации экрана устройства. Например, когда экран устройства имеет книжную ориентацию (расположен вертикально), кнопки в макете можно также размещать по вертикали, а когда экран развернут горизонтально (альбомная ориентация), кнопки следует размещать по горизонтали. Чтобы при изменении ориентации экрана изменялся макет, можно определить два разных макета и применить соответствующий квалификатор к имени каталога каждого макета. После этого система будет автоматически применять соответствующий макет в зависимости от ориентации устройства.

8. ТЕМАТИКА КУРСОВЫХ, КОНТРОЛЬНЫХ РАБОТ, РЕФЕРАТОВ

Данный вид работы не предусмотрен УП

9. ПЕРЕЧЕНЬ ВОПРОСОВ К ЭКЗАМЕНУ (ЗАЧЕТУ)

1. История развития мобильных операционных систем.
2. Операционные системы Android, iOS, Blackberry OS, Symbian OS, Microsoft Phone.
3. Среды разработки под операционные системы Android и iOS.
4. Android Studio, Eclipse, Apache Cordova, Kotlin, Xamarin, XCode.
5. Архитектура OS Android.
6. Структура разрабатываемых приложений под Android
7. Архитектура OS iOS.
8. Структура разрабатываемых приложений под iOS.
9. Среды разработки графических приложений.
10. Библиотека OpenGL.
11. Среда Unity 3D.
12. Игровой движок Unreal Engine.
13. Технология Apache Cordova.
14. Перенос мобильных приложений из одной ОС в другую.
15. Технология WiFi Direct для Android.
16. Технология Bluetooth LTE.
17. Библиотека MultipeerConnectivity для iOS.

18. Smart TV.

10. САМОСТОЯТЕЛЬНАЯ РАБОТА ОБУЧАЮЩИХСЯ

Форма обучения очная

Название разделов и тем	**Вид самостоятельной работы** ***(проработка учебного материала, решение задач, реферат, доклад, контрольная работа, подготовка к сдаче зачета, экзамена и др.)***	**Объем в часах**	**Форма контроля (*проверка решения задач, реферата и др.)***
Мобильные операционные системы	чтение основной и дополнительной литературы, самостоятельное изучение материала по литературным источникам;	4	опрос
Основы разработки мобильных приложений	чтение основной и дополнительной литературы, самостоятельное изучение материала по литературным источникам;	4	опрос
Разработка мобильных приложений под Android	чтение основной и дополнительной литературы, самостоятельное изучение материала по литературным источникам;	4	опрос
Разработка мобильных приложений под iOS	чтение основной и дополнительной литературы, самостоятельное изучение материала по литературным источникам;	4	опрос
Разработка графических мобильных приложений	самостоятельное выполнение практических заданий репродуктивного типа (ответы на вопросы, тренировочные упражнения, задачи, тесты);	4	Проверка решения задач
Разработка кросс-сплатформенных мобильных приложений	самостоятельное выполнение практических заданий репродуктивного типа (ответы на вопросы, тренировочные упражнения, задачи, тесты);	6	опрос
Перспективные технологии в мобильных приложениях	самостоятельное выполнение практических заданий репродуктивного типа (ответы на вопросы, тренировочные упражнения, задачи, тесты);	4	Проверка решения задач

10. УЧЕБНО-МЕТОДИЧЕСКОЕ И ИНФОРМАЦИОННОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ

а) Список рекомендуемой литературы

основная

1) Соколова, В. В. Разработка мобильных приложений : учебное пособие / В. В. Соколова. — Томск : Томский политехнический университет, 2014. — 176 с. — ISBN 978-5-4387-0369-3. — Текст : электронный // Электронно-библиотечная система

IPR BOOKS : [сайт]. — URL: http://www.iprbookshop.ru/34706.html (дата обращения: 05.11.2019). — Режим доступа: для авторизир. Пользователей

2) Введение в разработку приложений для ОС Android / Ю. В. Березовская, О. А. Юфрякова, В. Г. Вологдина [и др.]. — 2-е изд. — М. : Интернет-Университет Информационных Технологий (ИНТУИТ), 2016. — 433 с. — ISBN 2227-8397. — Текст : электронный // Электронно-библиотечная система IPR BOOKS : [сайт]. — URL: http://www.iprbookshop.ru/73669.html (дата обращения: 05.11.2019). — Режим доступа: для авторизир. пользователей

дополнительная

3) Вычислительная техника и информационные технологии. Разработка мобильных приложений : учеб. пособие для прикладного бакалавриата и для магистров по направл. "Информатика и вычислительная техника" / Соколова Вероника Валерьевна; Нац. исслед. Томск. политехн. ун-т. - Москва : Юрайт, 2018. - 175 с. - (Университеты России). - Библиогр.: с.174-175. - ISBN 978-5-9916-6525-4 (в пер.) : 150.00.

учебно-методическая

1) Семакова, А. Введение в разработку приложений для смартфонов на ОС Android / А. Семакова. — 2-е изд. — М. : Интернет-Университет Информационных Технологий (ИНТУИТ), 2016. — 102 с. — ISBN 2227-8397. — Текст : электронный // Электронно-библиотечная система IPR BOOKS : [сайт]. — URL: http://www.iprbookshop.ru/73670.html (дата обращения: 05.11.2019). — Режим доступа: для авторизир. Пользователей

Согласовано:

_____/_____/_____/_____Дол
Должность сотрудника научной библиотеки ФИО подпись дата

б) Программное обеспечение

1. Android Studio
2. Kotlin
3. Текстовый редактор

в) Профессиональные базы данных, информационно-справочные системы:

1. Электронно-библиотечные системы:

1.1. **IPRbooks** [Электронный ресурс]: электронно-библиотечная система / группа компаний Ай Пи Эр Медиа . - Электрон. дан. - Саратов, [2019]. - Режим доступа: http://www.iprbookshop.ru.

1.2. **ЮРАЙТ** [Электронный ресурс]: электронно-библиотечная система / ООО Электронное издательство ЮРАЙТ. - Электрон. дан. – Москва, [2019]. - Режим доступа: https://www.biblio-online.ru.

1.3. **Консультант студента** [Электронный ресурс]: электронно-библиотечная си-

стема / ООО Политехресурс. - Электрон. дан. – Москва, [2019]. - Режим доступа: http://www.studentlibrary.ru/pages/catalogue.html.

1.4. **Лань** [Электронный ресурс]: электронно-библиотечная система / ООО ЭБС Лань. - Электрон. дан. – С.-Петербург, [2019]. - Режим доступа: https://e.lanbook.com.

1.5. Znanium.com [Электронный ресурс]: электронно-библиотечная система / ООО Знаниум. - Электрон. дан. – Москва, [2019]. - Режим доступа: http://znanium.com.

2. **КонсультантПлюс** [Электронный ресурс]: справочная правовая система. /Компания «Консультант Плюс» - Электрон. дан. - Москва : КонсультантПлюс, [2019].

3. **База данных периодических изданий** [Электронный ресурс] : электронные журналы / ООО ИВИС. - Электрон. дан. - Москва, [2019]. - Режим доступа: https://dlib.eastview.com/browse/udb/12.

4. **Национальная электронная библиотека** [Электронный ресурс]: электронная библиотека. - Электрон. дан. – Москва, [2019]. - Режим доступа: https://нэб.рф.

5. **Электронная библиотека диссертаций РГБ** [Электронный ресурс]: электронная библиотека / ФГБУ РГБ. - Электрон. дан. – Москва, [2019]. - Режим доступа: https://dvs.rsl.ru.

6. Федеральные информационно-образовательные порталы:

6.1. Информационная система Единое окно доступа к образовательным ресурсам. Режим доступа: http://window.edu.ru

6.2. Федеральный портал Российское образование. Режим доступа: http://www.edu.ru

7. Образовательные ресурсы УлГУ:

7.1. Электронная библиотека УлГУ. Режим доступа : http://lib.ulsu.ru/MegaPro/Web

7.2. Образовательный портал УлГУ. Режим доступа : http://edu.ulsu.ru

Согласовано:

_____/_____/_____/_____

Должность сотрудника УИТиТФИО подпись дата

МАТЕРИАЛЬНО-ТЕХНИЧЕСКОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ:

Аудитории для проведения лекций, семинарских занятий, для выполнения лабораторных работ и практикумов, для проведения текущего контроля и промежуточной аттестации.

Аудитории укомплектованы специализированной мебелью, учебной доской. Аудитории для проведения лекций оборудованы мультимедийным оборудованием для предоставления информации большой аудитории. Помещения для самостоятельной работы оснащены компьютерной техникой с возможностью подключения к сети «Интернет» и обеспечением доступа к электронной инфромационно-образовательной среде, электронно-библиотечной системе. Перечень оборудования, используемого в учебном процессе, указывается в соответствии со сведениями о материально-техническом обеспечении и оснащенности образовательного процесса, размещенными на официальном сайте УлГУ в разделе «Сведения об образовательной организации».

11. СПЕЦИАЛЬНЫЕ УСЛОВИЯ ДЛЯ ОБУЧАЮЩИХСЯ С ОГРАНИЧЕННЫМИ ВОЗМОЖНОСТЯМИ ЗДОРОВЬЯ

В случае необходимости, обучающимся из числа лиц с ограниченными возможностями здоровья (по заявлению обучающегося) могут предлагаться одни из следующих вариантов восприятия информации с учетом их индивидуальных психофизических особен-

ностей:

– для лиц с нарушениями зрения: в печатной форме увеличенным шрифтом; в форме электронного документа; в форме аудиофайла (перевод учебных материалов в аудиоформат); в печатной форме на языке Брайля; индивидуальные консультации с привлечением тифлосурдопереводчика; индивидуальные задания и консультации;

– для лиц с нарушениями слуха: в печатной форме; в форме электронного документа; видеоматериалы с субтитрами; индивидуальные консультации с привлечением сурдопереводчика; индивидуальные задания и консультации;

– для лиц с нарушениями опорно-двигательного аппарата: в печатной форме; в форме электронного документа; в форме аудиофайла; индивидуальные задания и консультации.

Разработчик	_____	доцент кафедры ТТС	Булаев А.А.
	подпись	должность	ФИО