МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ УКРАИНЫ

ДОНЕЦКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

КАФЕДРА ПРИКЛАДНОЙ МАТЕМАТИКИ И ИНФОРМАТИКИ

**МЕТОДИЧЕСКИЕ УКАЗАНИЯ**

к выполнению лабораторных работ

по дисциплине

«ПРОГРАММИРОВАНИЕ МОБИЛЬНЫХ СИСТЕМ ИНТЕРНЕТА»

Для студентов направления подготовки

6.050103 «Программная инженерия»

всех форм обучения

Подготовили:

Костин В. И., ст. преп. каф. ПМИ

Глинская Е. С., ст. гр. ИПО-13

Утверждено на Принято на заседании

заседании учебно- кафедры прикладной

издательского совета ДонНТУ математики и информатики

протокол № \_\_\_\_\_ протокол № \_\_\_\_

от «\_\_\_\_» \_\_\_\_\_\_\_\_\_ 2017 г. от «\_\_\_\_» \_\_\_\_\_\_\_\_\_ 2017 г.

Покровск – 2017

Лабораторная работа №1

СОЗДАНИЕ ИГРЫ ДЛЯ МОБИЛЬНОГО УСТРОЙСТВА

**Цель работы**: ознакомиться с инструментами и основными элементами создания игр для мобильных устройств.

**ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ**

Существует множество сред, фреймворков и инструментов для мобильной разработки, выбор среди которых необходимо осуществлять в зависимости от целей проекта, требований к продукту и пр. Ниже представлены некоторые среды разработки, позволяющие создавать приложения на различных языках программирования и для разных платформ. В дальнейшем будем рассматривать особенности разработки приложений для Android как наиболее распространенной мобильной ОС (более 70% пользователей). Все описанные среды разработки или бесплатны, или имеют бесплатные версии продукта (кроме AppCode, предлагающей только бесплатную пробную версию на 1 месяц).

**Обзор некоторых средств разработки**

*Android Studio* – IDE для разработки приложений для Андроид от Google на основе IntelliJ IDEA. Сейчас она находится в стадии активной разработки. Студия доступна для всех основных платформ. Одна из наиболее перспективных сред на данный момент для разработки на Java.

*Eclipse* – свободная интегрированная среда разработки модульных кроссплатформенных приложений на языке Java. Долгое время оставалась основной средой разработки для Android приложений, благодаря чему существует огромное количество руководств и примеров работы в этой среде. Сейчас уступает позиции Android Studio.

*Xamarin* – фреймворк для кроссплатформенной разработки мобильных приложений (iOS, Android, Windows Phone) с использованием языка C#. Разработчикам Xamarin в качестве среды разработки предлагают использовать Visual Studio.

*Qt* – кроссплатформенный инструментарий разработки ПО на языке программирования C++. Есть также «привязки» ко многим другим языкам программирования: Python – PyQt, PySide; Ruby – QtRuby; Java – Qt Jambi; PHP – PHP-Qt и другие.

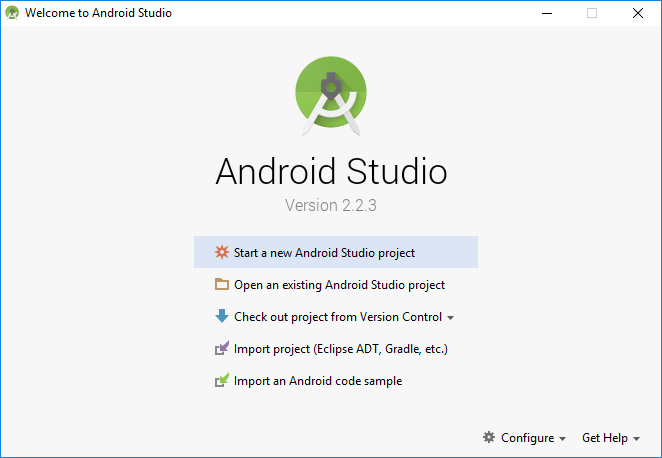
***AppCode*** – Objective-C IDE, предназначенная для разработчиков приложений для устройств Apple, таких как Mac, iPhone и iPad. AppCode содержит полный набор необходимых средств для эффективной и удобной работы с языком Objective-C и на 100% совместима с Xcode. Существует пробная бесплатная версия на 30 дней.

*Unity* – это инструмент (игровой движок) для разработки двух- и трёхмерных приложений и игр, работающий, в том числе, под операционными системами Windows Phone, Android, Apple iOS. Есть возможность создавать приложения для запуска в браузерах с помощью специального подключаемого модуля Unity (Unity Web Player), а также с помощью реализации технологии WebGL. Разработка может вестись на языках C#, JavaScript. Включает встроенную IDE Mono Develop, также легко интегрируется с Visual Studio.

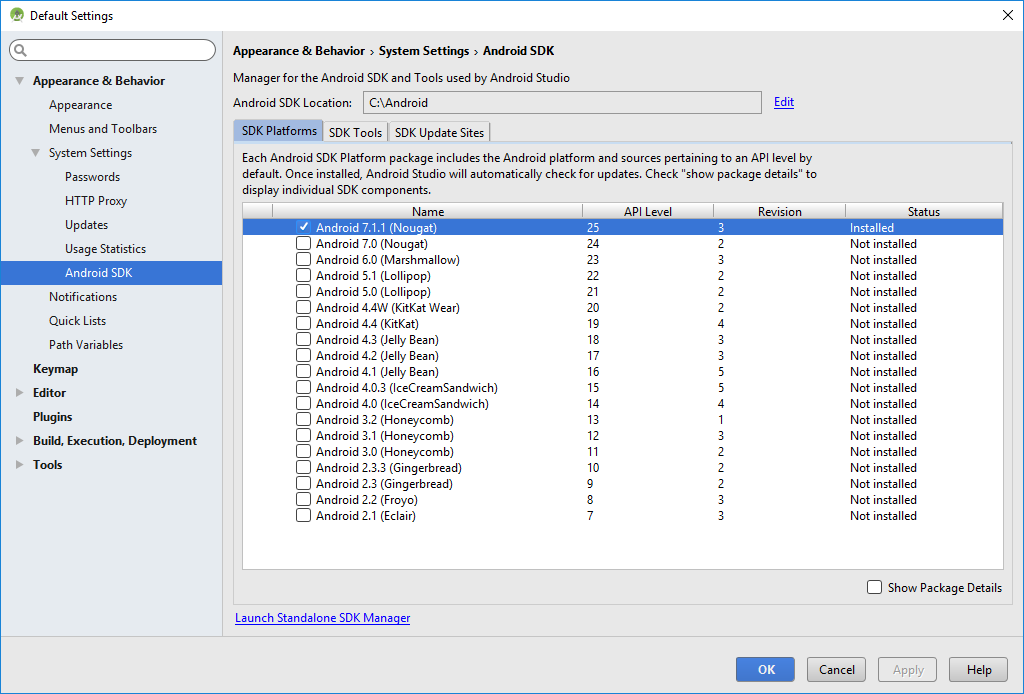
Для разработки приложений на Android на компьютере должны быть установлены JDK и Android SDK.

**Установка и настройка Android Studio**

1. Скачать среду разработки можно по ссылке [[1]](https://developer.android.com/studio/index.html). Для ОС Windows выбираем действие Download Android Studio for Windows, для другой ОС надо перейти по ссылке Download Options и выбрать нужный файл из таблицы Android Studio Packages. Соглашаемся с лицензионным соглашением и скачиваем файл размером около 1.6 ГБ.
2. Запускаем установочный файл и следуем инструкциям. От нас требуется указать два пути. Первый путь будет использован для установки Android Studio. Второй – для установки Android SDK. Пути можно изменить или оставить по умолчанию.
3. После установки Android Studio запустится и предложит импортировать настройки с прошлой версии.
4. Далее может выполниться поиск и установка недостающих компонентов и обновлений. После этого нажимаем Finish и видим стартовое окно.



1. В выпадающем списке Configure можно перейти в SDK Manager для установки дополнительных Android SDK-платформ и обновлений. По умолчанию уже должна быть установлена последняя версия SDK-платформы. В поле Android SDK Location необходимо проверить корректность пути к Android SDK.

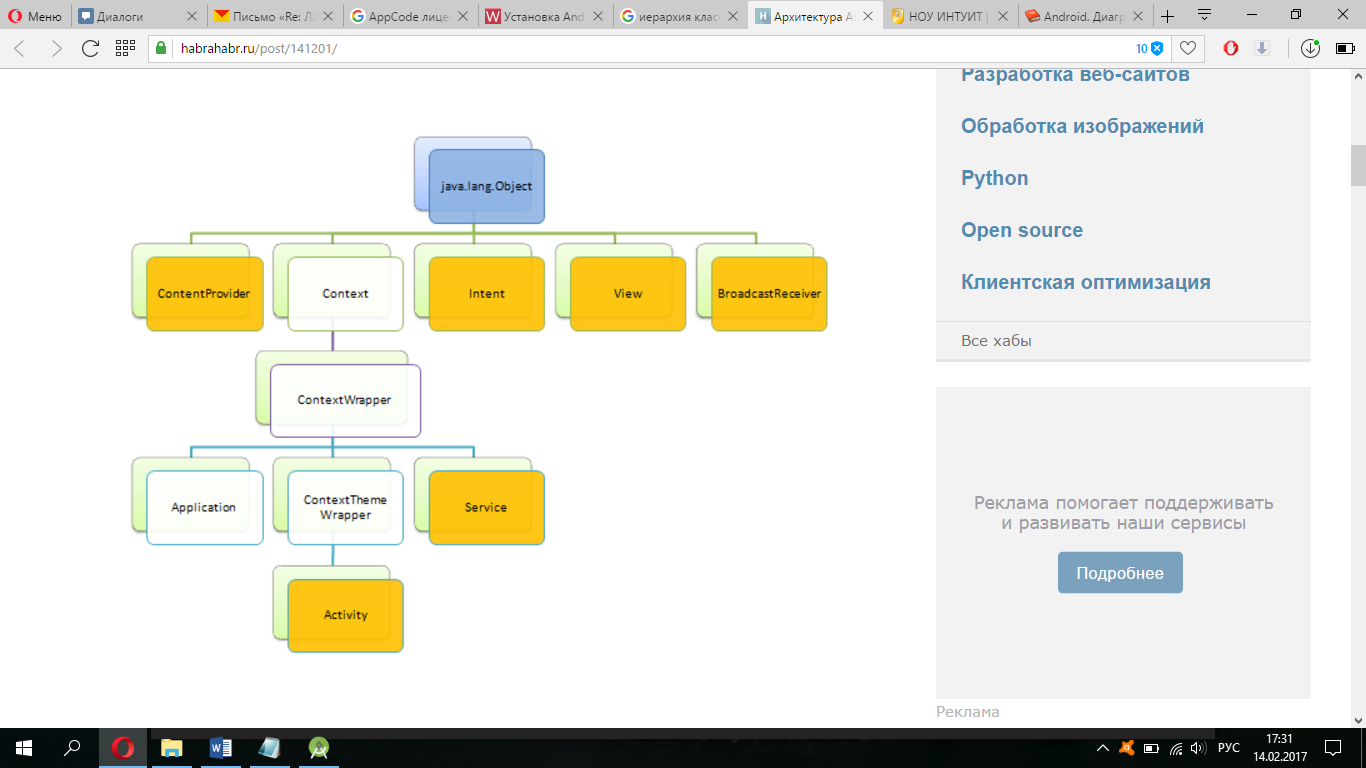


1. После установки необходимых платформ и проверки наличия обновлений возвращаемся в стартовое окно, где выбираем пункт Start a new Android Studio Project. Прежде чем начинать создание приложения, ознакомимся с базовыми понятия среды.

**Основные компоненты архитектуры приложений для Android**

* Activity (Активность, Деятельность) – представляет собой экран пользовательского интерфейса, который представлен классом Activity и разметкой в виде XML-файла (разметку можно создать и программно, но теперь так не делают). Android-приложение способно состоять из нескольких форм (Activity) и может переключаться между ними во время выполнения приложения
* Intents (Намерения) – асинхронные сообщения, которые позволяют приложению запросить функции из других служб или действий. Приложение может делать прямые запросы службе или действию (явное намерение) или запросить у Android зарегистрированные службы и приложения (неявное намерение). Для примера, приложение может запросить через Intent приложение Контакты для получения данных или запустить браузер при нажатии ссылки в тексте вашего приложения.
* Views (Представление, Вид, Элемент управления, Компонент) – пользовательский интерфейс, создаваемый виджетами классов. Это знакомые нам элементы управления: кнопки, текстовые поля, флажки, переключатели и т.п.
* Services (Службы) – выполняют фоновые задачи без предоставления пользовательского интерфейса (например, проигрывание музыки). Они могут уведомлять пользователя через систему уведомлений Android.
* Content Provider (Контент-провайдер) – предоставляет данные приложениям. С помощью контент-провайдера ваше приложение может обмениваться данными с другими приложениями.
* Broadcast Receiver (Приемник широковещательных сообщений/запросов) – принимает системные сообщения и неявные намерения, может использоваться для реагирования на изменение состояния системы. Приложение может регистрироваться как приемник определенных событий и может быть запущено, если такое событие произойдет.

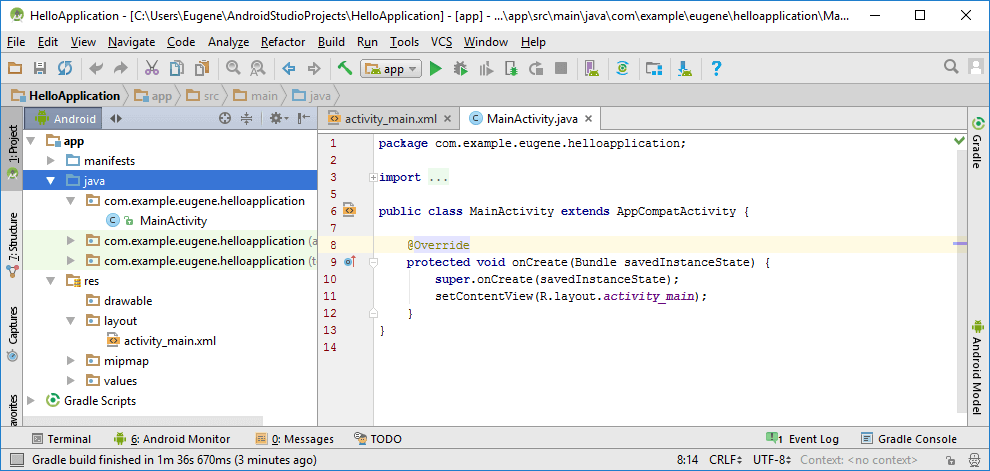
На рисунке ниже показана иерархия основных классов Android SDK. На самом деле классов намного больше, желтым цветом выделены классы, с которыми разработчик работает непосредственно, наследует от них свои классы. Остальные классы не менее важны, но они реже используются напрямую.



Иерархия классов Android SDK

**Обзор структуры проекта в Android Studio**

При создании нового приложения в среде Android Studio с настройками по умолчанию структура проекта будет представлена в следующем виде:



Структура проекта в Android Studio

Чтобы перейти к полной структуре проекта, необходимо нажать двойным щелчком на название проекта. Тогда проекта откроется полностью и будет включать следующие компоненты:

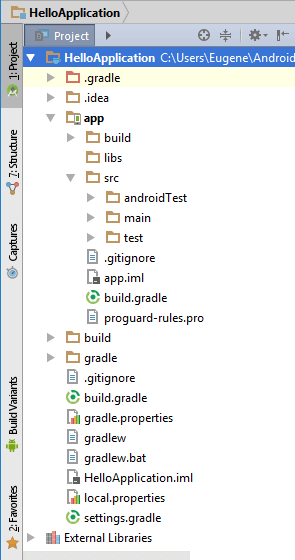
* все модули проекта описываются файлом setting.gradle;
* весь значимый код - файлы интерфейса, классы java и т.д. у нас по умолчанию находятся в папке (модуле) app;
* каталог libs предназначен для хранения библиотек, используемых приложением;
* файл build.gradle содержит информацию, которая используется при построении проекта;
* каталог src предназначен для хранения исходного кода. Он содержит ряд подкаталогов. Каталоги androidTest и test предназначены для хранения файлов тестов приложения. А собственно исходные коды располагаются в папке main;
* AndroidManifest.xml описывает фундаментальные характеристики приложения и определяет каждый из компонентов данного приложения;
* папка java содержит исходные файлы приложения. По умолчанию в ней находится файл класса MainActivity, который запускается по умолчанию при старте приложения;
* папка res содержит каталоги с ресурсами, в частности она содержит следующие каталоги:

- папка drawable предназначена для хранения изображений;

- папка layout предназначена для хранения файлов, определяющих графический интерфейс. По умолчанию здесь есть файл activity\_main.xaml, который определяют интерфейс для единственной в проекте activity – MainActivity;

- папки mipmap-xxxx содержат файлы изображений, которые преназначены для создания иконки приложения при различных разрешениях экрана. Соответственно для каждого вида разрешения здесь имеется свой каталог

- папка values хранит различные xml-файлы, содержащие коллекции ресурсов – различных данных, которые применяются в приложении.

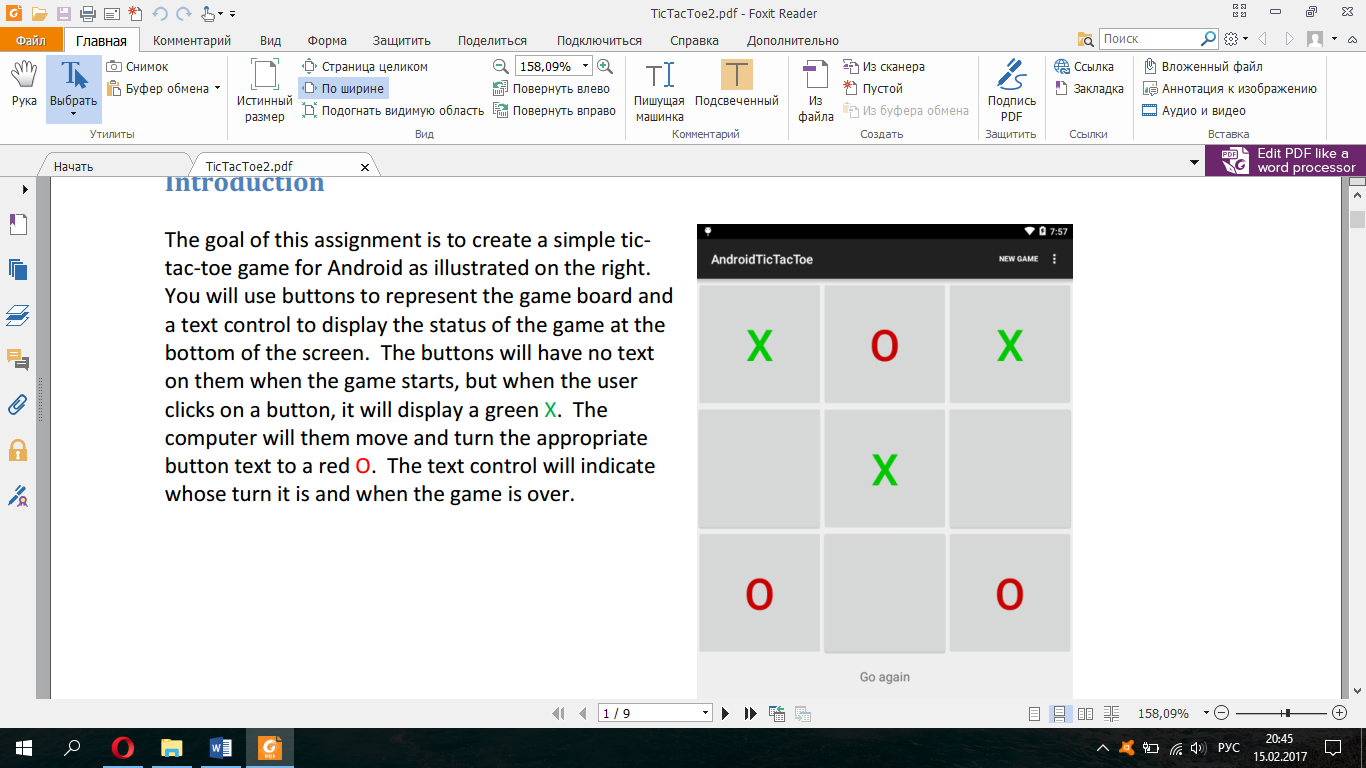


Полная структура проекта в Android Studio

**Разработка игры “Крестики-нолики” 3х3**

Опишем общие моменты разрабатываемого примера. На рисунке ниже показано, как будет выглядеть игровое поле в процессе игры. Будем использовать кнопки для представления клеток поля. Первоначально кнопки не будут содержать текста, при щелчке на них игроком (или в результате хода компьютера) на кнопках будет появляться крестик или нолик. Сверху можно будет выводить количество побед текущего игрока и его имя (для реализации таблицы рекордов), а также кнопку для начала новой игры. Внизу в текстовом поле будет отображаться текущий статус игры (чей ход или кто победил).

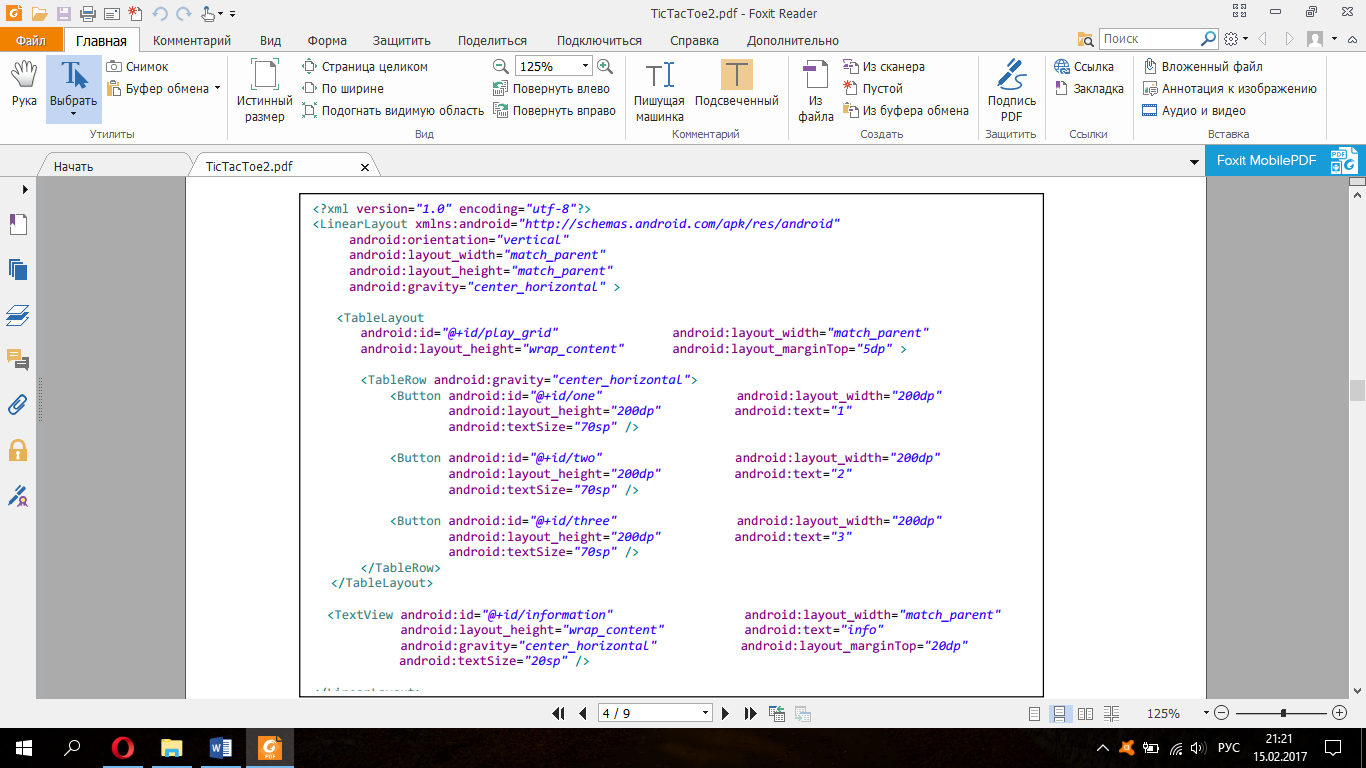
Игра должна включать в себя простой ИИ. Компьютер будет или выстраивать необходимую для победы последовательность из трех нулей в любом доступном месте поля, или блокировать действия игрока, когда они близки к победе.



1. Создадим новый проект в Android Studio. При создании необходимо задать ему имя (например, AndroidTicTacToy), путь, минимальную поддерживаемую версию Андроид. Android Studio рекомендует выбирать «API 15 Android 4.0.3».
2. Выберем шаблон проекта из предложенных – Basic Activity. Назовем активность TicTacToy.
3. Сформируем игровую доску.

Перейдем в директорию res/layout и выберем файл с разметкой активности your\_activity\_name.xml. Файл можно открывать в режиме Design (графический, так будет выглядеть экран в эмуляторе) или Text (XML-разметка).

Чтоб разместить кнопки игрового поля в нужном виде используем компоненты LinearLayout (распологает элементы интерфейса горизонтально или вертикально) и TableLayout (распологает элементы интерфейса по строкам/столбцам). Откроем окно с разметкой xml и опишем там необходимые элементы. На рисунке ниже приведен пример для первых трех кнопок, а также информационного текста внизу экрана (для отображения текущего статуса игры). Оставшиеся кнопки (4 – 9) описываются аналогично в собственных компонентах TableRow.



Атрибуты кнопок:

- id – уникальный идентификатор, будет использоваться в исходном коде для обращения к кнопке;

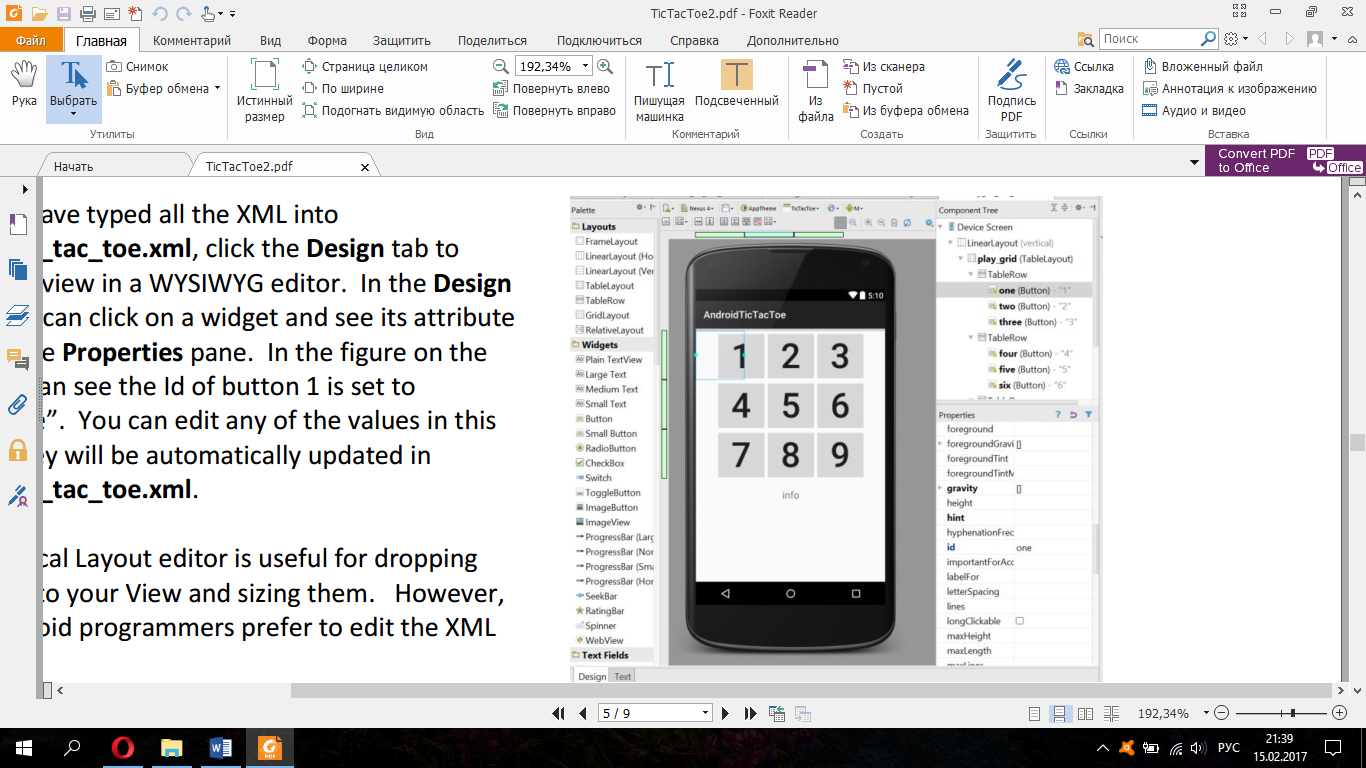
- layout\_width – ширина компонентов в dp (независимых от плотности пикселей).

- layout\_height – высота компонентов;

- text – отображаемый на кнопке текст;

- textSize – размер текста.

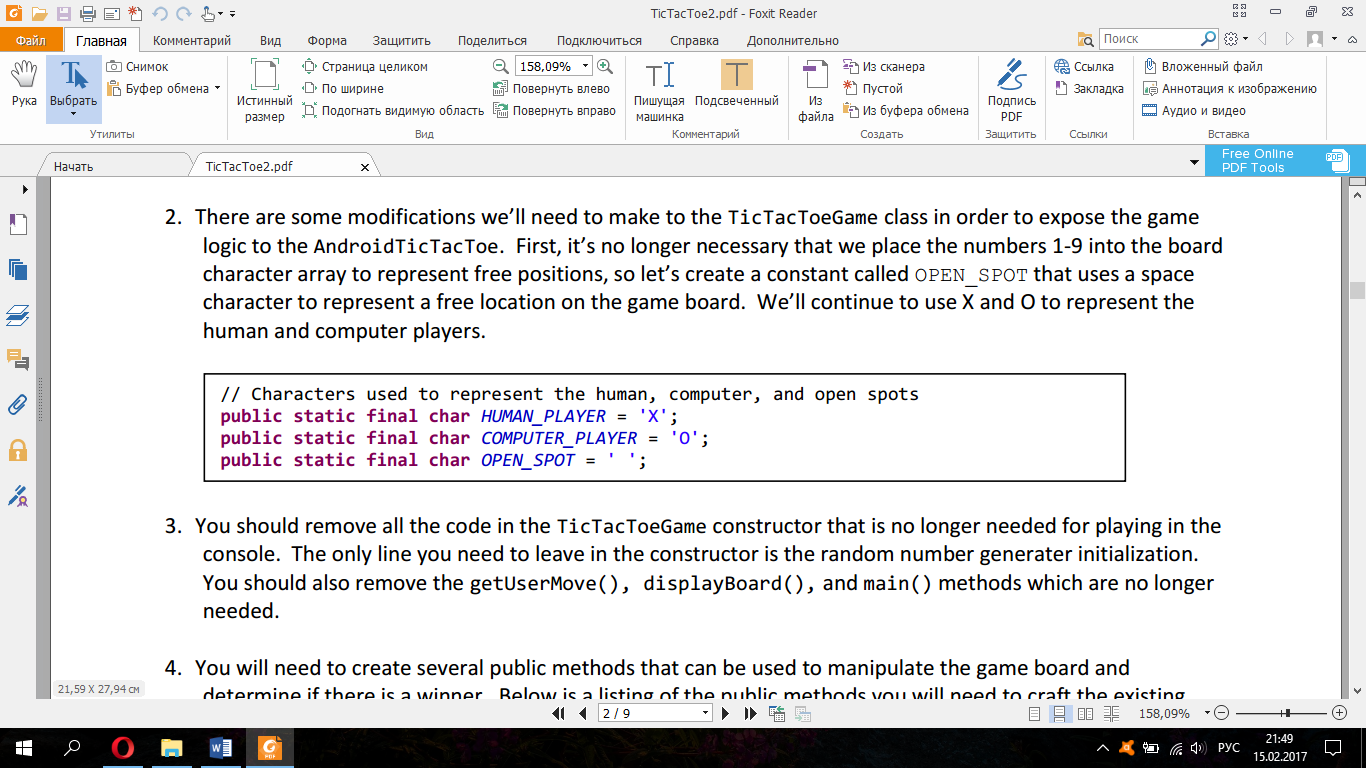
Перейдите на вкладку Design, чтобы увидеть, как будет выглядеть экран телефона для заданной разметки (при желании некоторые из них можно изменить). На рисунке показан полученный результат.



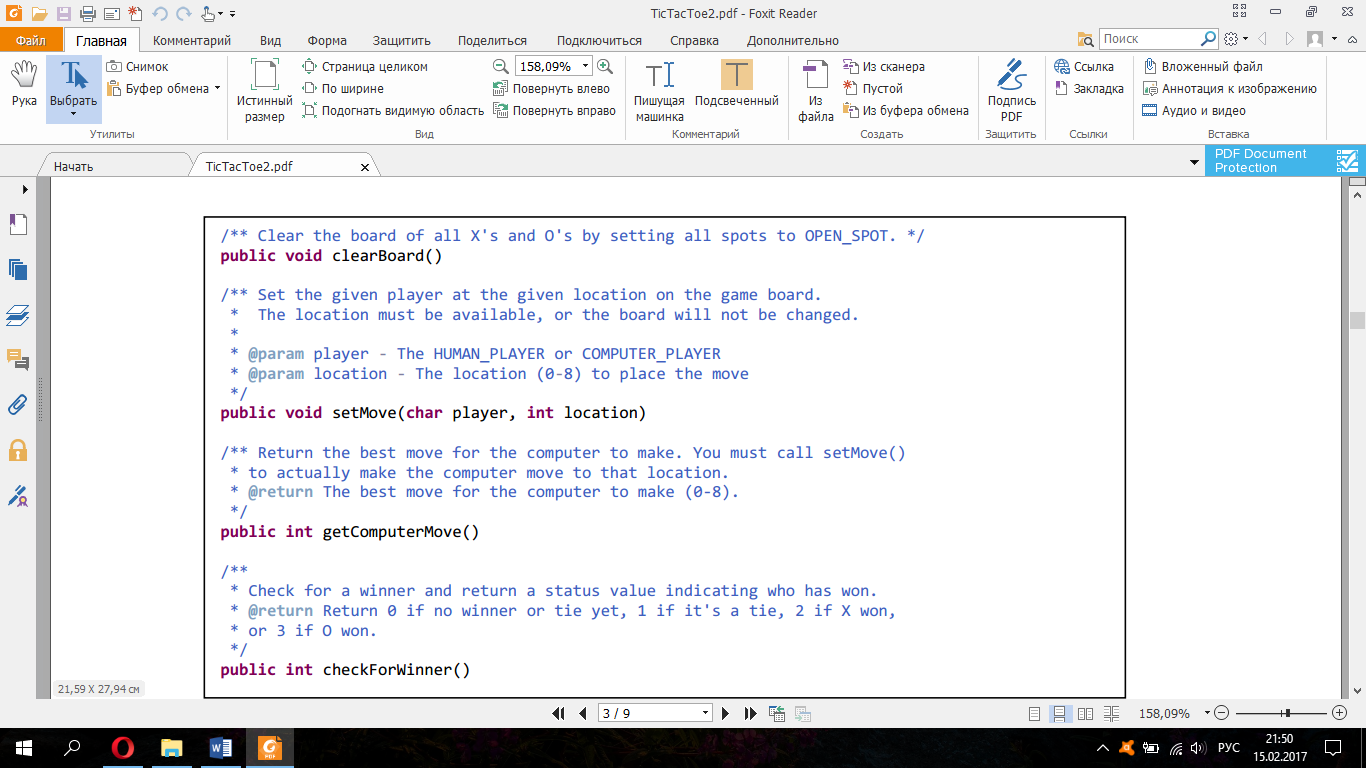
1. Реализуем взаимодействие кода с игровой доской

Создадим файл TicTacToyGame.java, нажав File -> New -> Java Class. Здесь будем описывать логику игры.

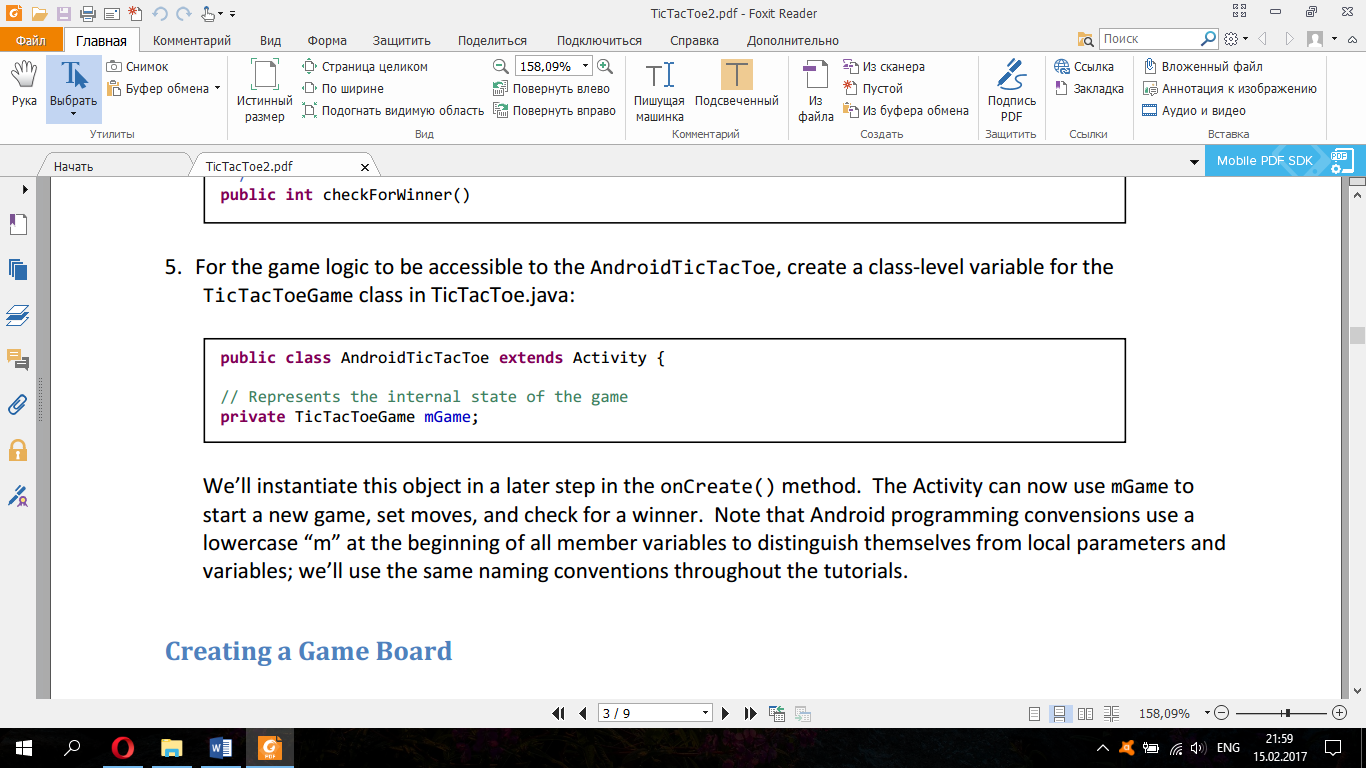
- опишем переменные для хранения текста кнопок (значений ходов игроков):



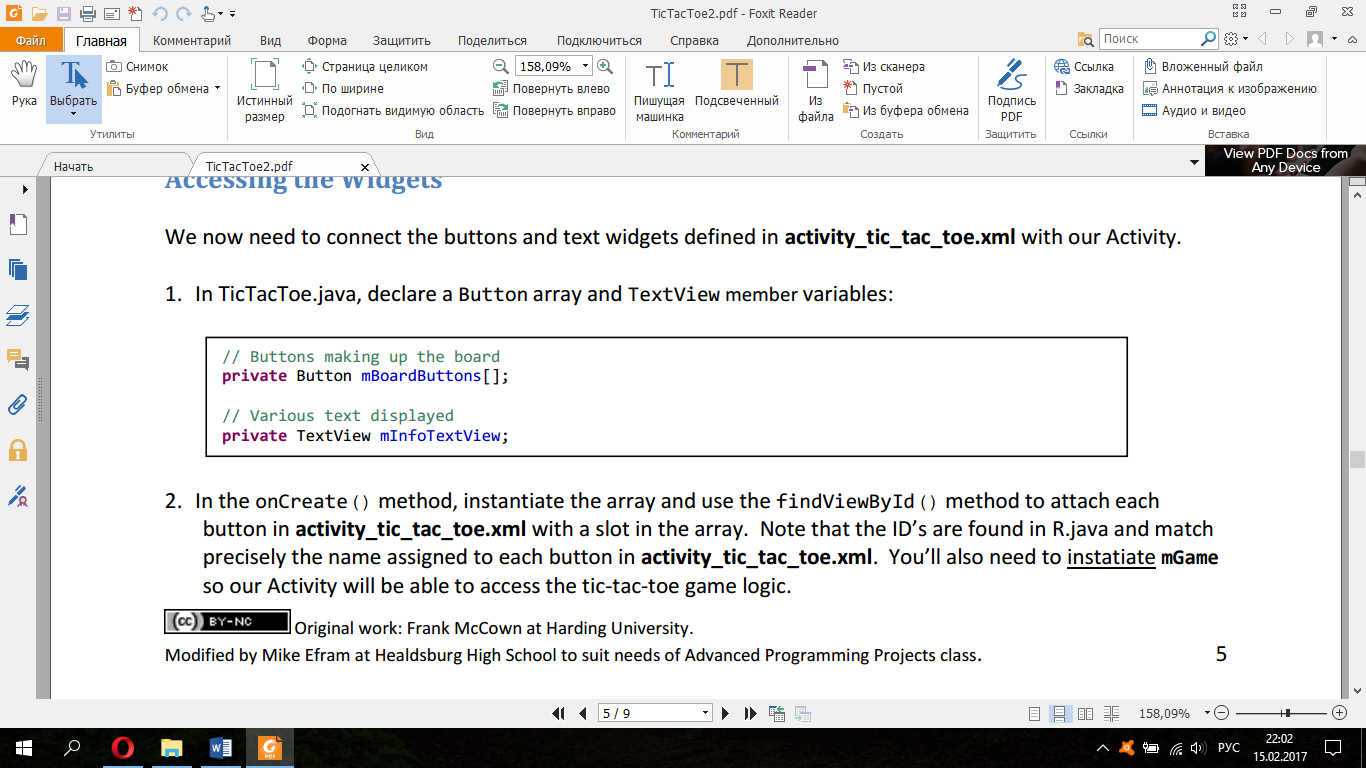
- опишем структуру основных методов класса:



Чтобы реализованная логика была доступна в AndroidTicTacToy, создадим объект только что созданного класса TicTacToyGame в основной активности TicTacToy.java (создана при создании всего проекта):



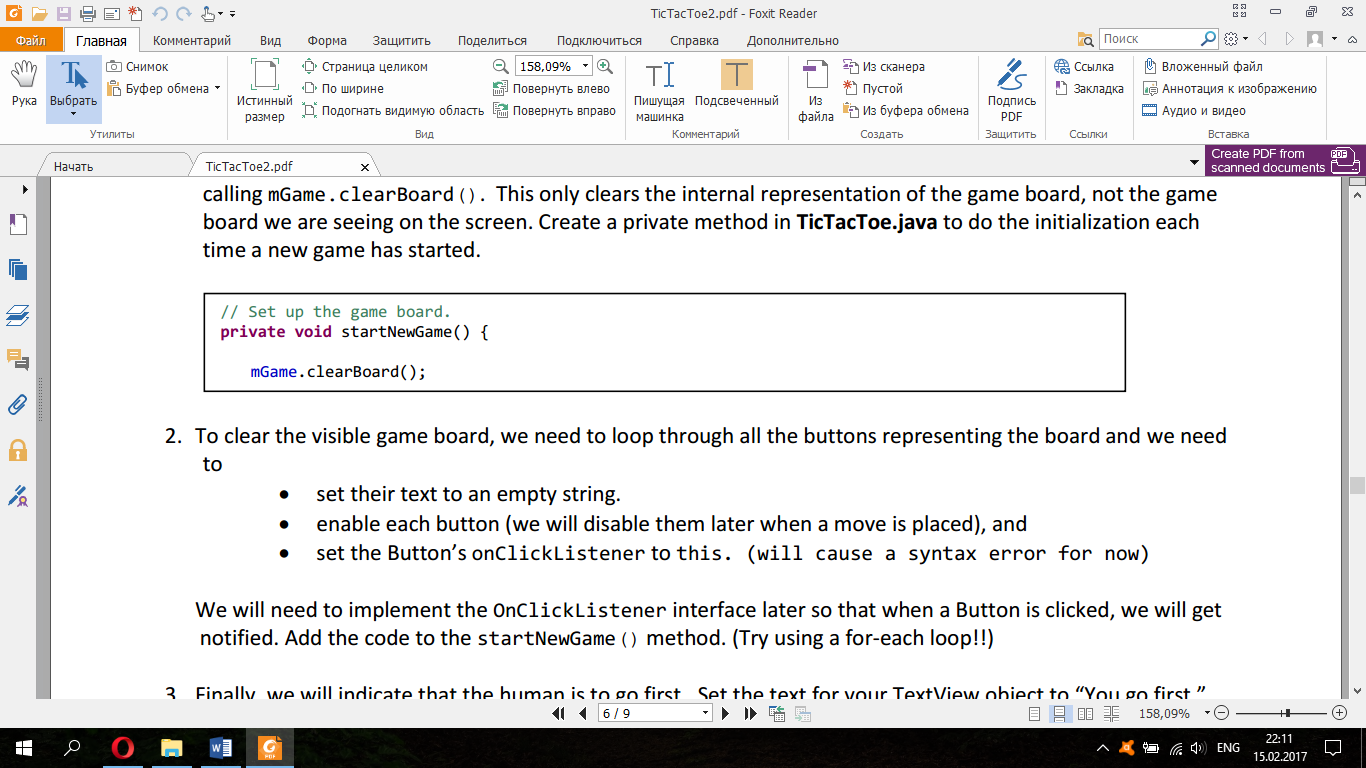
Активность сможет использовать этот объект для вызова описанных выше методов и управления игровым процессом. Опишем в этом же классе переменные для взаимодействия с кнопками и элементом TextView:



В методе классе onCreate() (создается автоматически) свяжем элементы массива с соответствующими кнопками с помощью метода findViewById(). Тут же проинициализируем переменную mGame.

Реализуем метод startNewGame() в классе TicTacToy.java:

* Очистим игровое поле от предыдущих игр описанным ранее методом clearBoard() созданного класса TicTacToyGame, вызвав его для объекта mGame.



- Метод clearBoard() должен перебрать все кнопки из массива и для каждой:

- установить значение текста как пустую строку;

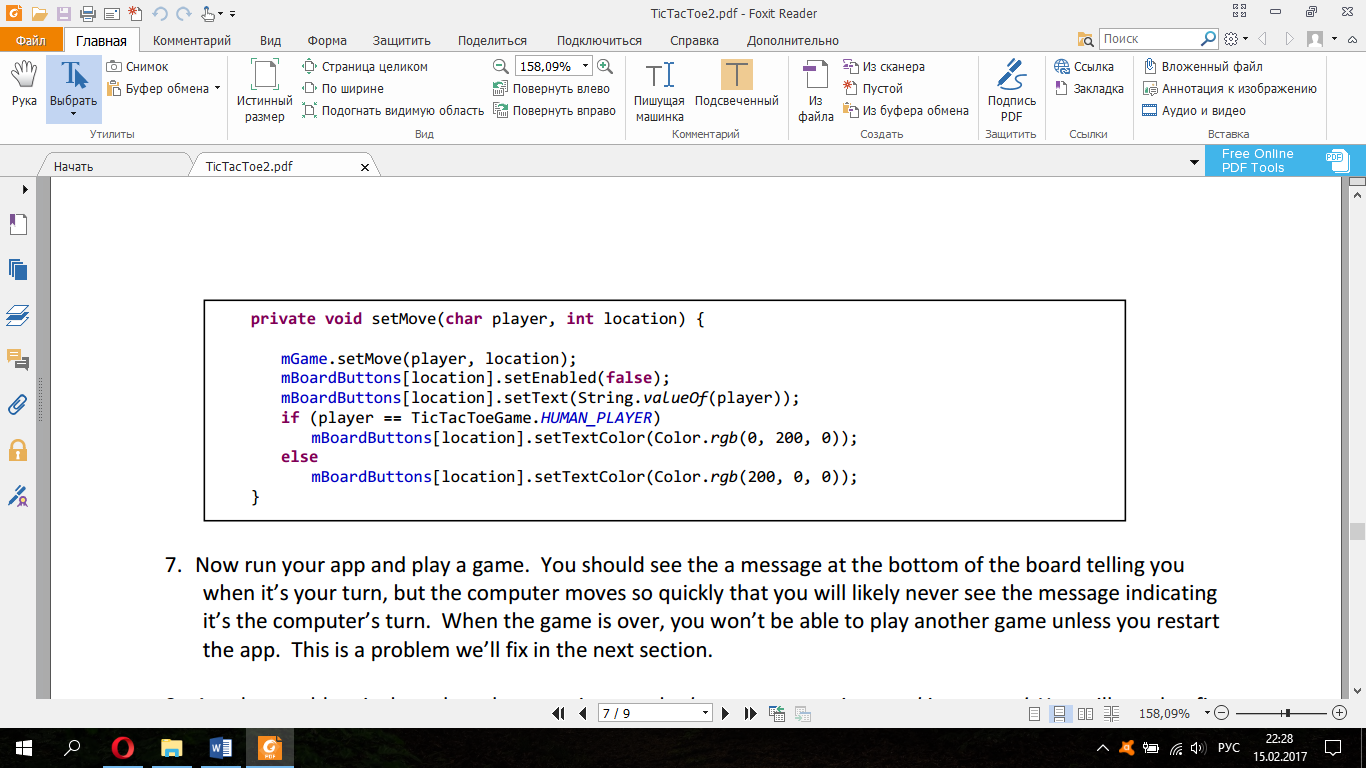
- сделать доступной (enable), так как после нажатия (хода) по этой кнопке она будет делаться недоступной для других ходов в текущей игре;

- установить для кнопки onClickListener для распознавания клика и, как следствие, обработки события onClick().

- Установим значение текста элемента TextView в «Ваш ход», чтобы игрок мог начинать игру.

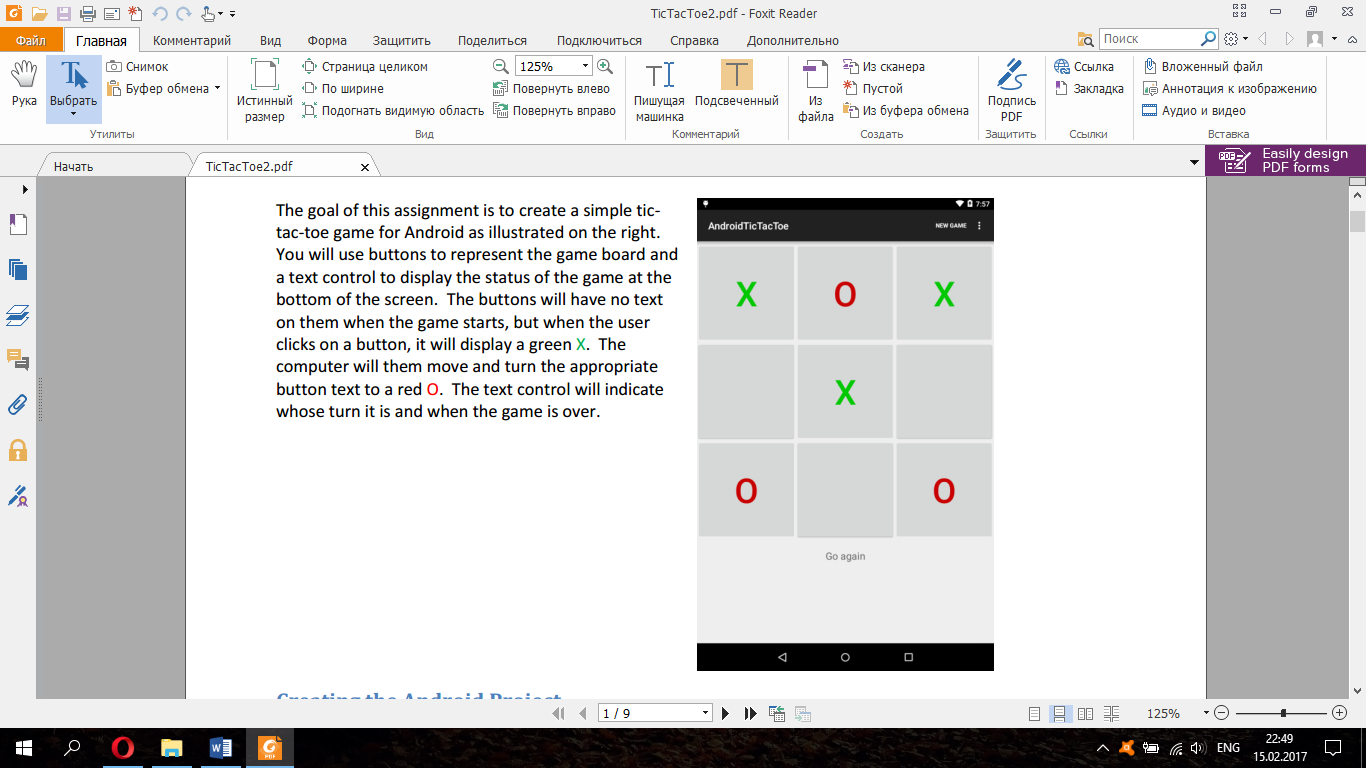
Метод startNewGame() можно вызывать при запуске приложения, добавив его вызов в реализацию метода onCreate().

Перейдем к реализации обработки события нажатия на кнопку onClick(). Если кнопка (location – номер кнопки от 0 до 8) доступна (enable), будем вызывать метод setMove(), представленный на рисунке ниже. Он отображает на экране крестик или нолик (разными цветами) в зависимости от игрока player, делает кнопку более недоступной для хода.

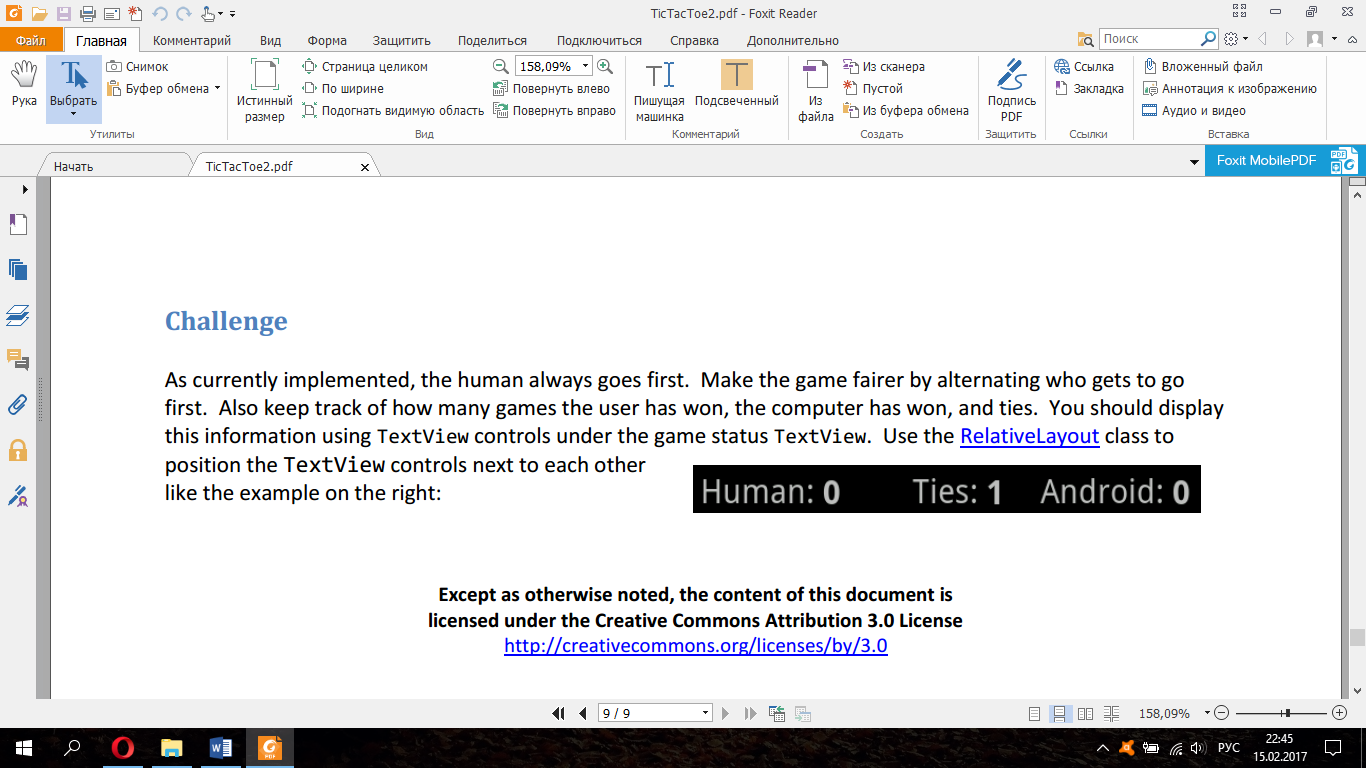


После хода игрока необходимо проверить, не победил ли он, методом mGame.checkForWinner(), после чего вызвать метод mGame.getComputerMove(). Ходы игрока и компьютера (вместе с проверками о победе) повторяются циклически до победы.

Чтобы запустить игру в Android Studio можно создать виртуальное устройство, задав ему необходимые параметры. Можно также использовать сторонний эмулятор или андроид в режиме отладки.



Для отображения числа побед и имени игрока вернемся в .xml файл и добавим в него компонент RelativeLayout с необходимыми элементами TextView. В файле исходного кода .java добавим переменные для хранения требуемой информации.



Для сохранения игровой статистики описанные данные необходимо записывать в файл перед выходом из приложения. Рассмотрим способы хранения информации в Android.

**Работа с БД**

Доступом к хранилищу данных управляет Контент-провайдер. Для реализации провайдера в Android приложении должен быть создан набор классов в соответствии с манифестом приложения. Один из этих классов должен быть наследником класса ContentProvider, который обеспечивает интерфейс между контент-провайдером и другими приложениями.

Если приложение требует использования контент-провайдера, необходимо выполнить несколько этапов для создания этого компонента: 1. Проектирование способа хранения данных. Данные, с которыми работают контент-провайдеры, могут быть организованы двумя способами:

• Данные представлены файлом, например, фотографии, аудио или видео. В этом случае необходимо хранить данные в собственной области памяти приложения. В ответ на запрос от другого приложения, провайдер может возвращать ссылку на файл.

• Данные представлены некоторой структурой, например, таблица, массив. В этом случае необходимо хранить данные в табличной форме. Строка таблицы представляет собой некоторую сущность, например, сотрудник или товар. А столбец – некоторое свойство этой сущности, например, имя сотрудника или цена товара. В системе Android общий способ хранения подобных данных – база данных SQLite, но можно использовать любой способ постоянного хранения.2. Создание класса-наследника от класса ContentProvider напрямую или через любого его потомка. При этом в реализации класса необходимо переопределить обязательные методы:

query() – метод, извлекающий данные из провайдера, в качестве аргументов получает таблицу, строки и столбцы, а также порядок сортировки результата, возвращает объект типа Cursor.

insert() – метод, добавляющий новую строку, в качестве аргументов получает таблицу, и значения элементов строки, возвращает URI добавленной строки.

update() – метод, обновляющий существующие строки, в качестве аргументов получает таблицу, строки для обновления и новые значения элементов строк, возвращает количество обновленных строк.

delete() – метод, удаляющий строки, в качестве аргументов принимает таблицу и строки для удаления, возвра щает количество удаленных строк.

getType() – метод, возвращающий String в формате MIME, который описывает тип данных, соответствующий URI.

onCreate() – метод, вызываемый системой, сразу после создания провайдера, включает инициализацию провайдера. Стоит отметить, что провайдер не создается до тех пор, пока объект ContentResolver не попытается получить к нему доступ.

3. Определение строки авторизации провайдера, URI для его строк и имен столбцов.

**Использование эмуляторов**

Протестировать андроид-приложение на компьютере можно с помощью эмуляторов. Большинство эмуляторов позволяют протестировать «поведение» программы на устройствах с разными характеристиками, а также реализуют их основной функционал (управление, имитация работы с СМС, входящими звонками, камерой, GPS, батареей, загрузка и экспорт файлов и др.), хотя и не могут заменить тестирование на реальных устройствах. Наиболее известные эмуляторы:

- BlueStacks 2

- Andy

- NOX App Player

- Leapdroid

- Droid4x

- Genymotion

**Вопросы для самопроверки**

1. Назовите известные вам среды разработки для мобильных приложений. Каковы их особенности?
2. С какими классами Android SDK чаще всего взаимодействует разработчик? Для чего они предназначены?
3. Какие основные компоненты проекта в Android Studio? Каково их назначение?
4. Какой класс в Android SDK отвечает за доступ к хранилищу данных?
5. Каким образом можно хранить и использовать данные в мобильном приложении? Какую БД для этого, как правило, используют?
6. Зачем может понадобиться использование эмуляторов при разработке мобильного приложения? Какие эмуляторы андроид-устройств вам известны?

**Дополнительные материалы**

1. Официальный сайт для загрузки среды Android Studio [Электронный ресурс] – <https://developer.android.com/studio/index.html> - дата обращения 09.02.2017
2. Разработка приложений для мобильных операционных систем Android, курс лекций [Электронный ресурс] <http://www.brsu.by/sites/default/files/priclmath/android.pdf> - дата обращения 09.02.2017.
3. Курс по работе с Android [Электронный ресурс] – <http://metanit.com/java/android/1.1.php> - дата обращения 12.02.2017.
4. Первая программа в Android Studio [Электронный ресурс] <http://ds-release.ru/pervaya-programma-v-android-studio/> - дата обращения 09.02.2017.
5. Игра для тренировки памяти [Электронный ресурс] <http://ru-code-android.livejournal.com/3332.html> - дата обращения 09.02.2017.
6. Работа с базой данных SQLite [Электронный ресурс] <http://ds-release.ru/kak-rabotat-s-bazoj-dannyx-sqlite-v-android-studio/> - дата обращения 09.02.2017.
7. Основные принципы разработки игр [Электронный ресурс] / Основ- ные принципы разработки игр. - Режим доступа: http://habrahabr.ru/ post/188372/. - дата обращения 09.02.2017.
8. Суровый геймдев на примере трех игр для Android [Электронный ресурс] / Суровый геймдев на примере трех игр для Android. - Режим доступа: <http://habrahabr.ru/post/195828/> . - Дата доступа: 09.02.2017.

**Задание**

Создать игру для мобильного устройства (КПК, Smartphone и т.п.), используя любой язык программирования для мобильных устройств (C#, C++, J2ME, Python и т.п.). В игре обязательно должен выполняться подсчёт очков. Рекордное количество очков (10 рекордов) должно сохраняться в файле и по выбору пользователя выводиться в виде таблицы рекордов в отсортированном по количеству набранных очков виде. Если пользователь в результате игры набрал рекордное количество очков, то ему необходимо предоставить возможность ввести своё имя, а после ввода имени показать таблицу рекордов, в которой выделяется только что добавленная запись.

**Варианты заданий**

1. Игра «Doodle Jump»
2. Игра «Убей крота»
3. Игра «Виселица»
4. Игра «Морской бой»
5. Игра «Тетрис»
6. Игра «Змейка»
7. Игра «Сапер»
8. Игра «Zuma»
9. Игра «Миллионер»
10. Игра «Fruit Ninja»
11. Игра «Flappy Bird»
12. Игра «Booble Pope»
13. Игра «Unblock Me»
14. Игра «Поиск слов»

Или любая другая игра на выбор студента