Провели тестирование с помощью белого и черного ящиков.

Тестирование белым ящиком было проведено средствами MSVisualStudio 2015 в ходе предыдущих этапов.

В результате проделанной работы по разработке android-приложение «Расписание» был разработан функционирующий комплекс приложений реализующий обмен данными между клиентом и сервером, с последующим администрированием. В приложении пользователю доступен интуитивно понятный интерфейс, позволяющий пользователю посредством выбора факультета, курса и группы получить свое расписание занятий.

Дальнейшая разработка программного средства может вестись в направлении разработки дополнительных функций для клиентского приложения и их поддержки сервером.

ЛИТЕРАТУРА

1. Хашими С. Разработка приложений для Android / С.Хашими , С.Коматинени, Д.Маклин СПб.: Питер, 2011. — 736 с.

2. Голощапов А.Л. Google Android программирование для мобильных устройств. 2-е изд., перераб. и доп. СПб.: БХВ-Петербург, 2012. — 448 с.: ил. — (Профессиональное программирование).

3. Кроссплатформенное мобильное приложение. Режим доступа: http://wiki.soloten.com/ (дата обращения 21.01.2015).

4. Подробно о Xamarin. Режим доступа: http://habrahabr.ru/post/188130/ (дата обращения 21.01.2015).

УДК 004.91

Студ. Д. А. Шуманский
Науч. рук. доц. Н. А. Жиляк
(кафедра информационных технологий, БГТУ)

РЕАЛИЗАЦИЯ ПРИЛОЖЕНИЯ В ANDROID STUDIO

AndroidStudio представляет собой инструментарий для создания *Android* приложений. С помощью этой платформы разработка программ становится проще.

В качестве языка программирования для *Android* используется *Java*. Для создания пользовательского интерфейса используется *XML*.

При создании проекта необходимо выбрать тип устройства, под которое будет разрабатываться приложение. Кроме выбора типа устройства, необходимо выбрать минимальную версию системы, под которую будет работать приложение. Так же *Студия* предоставляет

пользователю выбор готовых шаблонов внешнего вида экрана, что позволяет сэкономить время на написание стандартного кода для типичных ситуаций.

Студия формирует проект и создаёт необходимую структуру из различных файлов и папок.

В левой части среды разработки во вкладке *Android* находятся две основные папки: *app*и *GradleScripts*. Первая папка *app*содержит все необходимые файлы приложения – код, ресурсы картинок и т.п. Вторая папка служит для различных настроек, управления проектом и др.

Папка app содержит три папки: *manifest*, *java*, *res*.

Папка *manifest* содержит единственный файл манифеста *AndroidManifest.xml*. В этом файле должны быть объявлены все активности, службы, приёмники и контент-провайдеры приложения. Также он должен содержать требуемые приложению разрешения такие как доступ к сети.

Папка *java*содержит две подпапки – рабочую и для тестов. Рабочая включает файлы классов проекта.

Папка *res*содержит файлы ресурсов, разбитых на отдельные подпапки. *Drawable* – в этой папке хранятся графические ресурсы – картинки и xml-файлы, описывающие цвет и фигуры. *Layout* – в данной папке содержатся xml-файлы, описывающие внешний вид окон приложения. *Mipmap* – здесь хранят значки приложения под разные разрешения экрана. *Values*–тут располагаются какие-либо строковые ресурсы, ресурсы цветов, тем, стилей и измерений, которые используются в проекте.

По центру находится внешний вид экрана устройства, который формируется в xml-файле. Файл можно просматривать в двух режимах: текстовом и визуальном. Для переключения данных режимов служат две вкладки в нижней части окна редактора: *Design*и *Text*.

Правее размещено дерево компонентов, а под ним находятся свойства данных компонентов.

Левее от графического представления окна приложения находится перечень компонентов, сгруппированных по группам *Layout*, *Widgets*, *TextFields* и т. д., которые можно добавить путем перетаскивания или путем написания в самом xml-документе.

Структура xml-файла достаточно проста – стандартное дерево xml-элементов, где каждый узел является именем класса *View*, например, *TextView*. Корневой элемент может быть представлен *RelariveLayout*, *FrameLayout*и другие доступные нам компоненты, представленные в группе *Layout.* Добавив в выбранный корневой элемент дру-

гие компоненты, описанные выше, можно задать им некоторые свойства такие как цвет текста, фон, высота, ширина и тд. Хотелось бы обратить внимание, что размеры компонентов целесообразно задавать в так называемых *dp*. *Dp* - это такая абстрактная единица, которая на всех устройствах отображается одинаково, в отличие от того, если бы мы использовали к примеру значения в пикселях. Пример такой записи будет иметь вид: *android:layout_width = "42dp"*.Для задания фона компонента используется запись вида *android:background = "#ffffc0cb"*.Здесь мы установили нужное нам значение цвета. Но это неправильный подход. Правильным будет использование цветовых реурсов, которые будут хранится в файле *color.xml*. Тогда запись у нас будет иметь вид: *android:background = "@color/background_color"*. Похожей записью будет представлена загрузка картинки в качестве фона: *android:background = "@drawable/image"*.

Для многих компонентов целесообразно задавать *id*, чтобы в дальнейшем иметь возможность обратится к ним в файле *java*. Допустим у нас имеется *Button*. Хотелось бы напомнить, что строковый параметр необходимо хранить в файле ресурсов *string.xml*, находящийся в папке *values*. Присвоение *id*элементу будет иметь следующий вид:*android:id = "@+id/button"*.

Разместив на экране элемент *button*и задав ему некоторые свойства, которые включают в себя *id*необходимо создать обработчик нажатия. Для этого и необходимы классы *java*. Предварительно создав новое *activity*, содержащий класс javaи xml-файл, необходимо, чтобы по нажатию на кнопку открылось данное окно. Для этого нам необходимо в главном классе объявить нашу кнопку следующим образом: *Buttonbutton*; в методе *onCreate*найти и инициализировать наши элементы экрана с помощью метода *findViewById*, в метода *SetContentView*: *button = (Button) findViewById(R.id.button);* объект определен и готов к работе. Далее для каждой кнопки необходимо настроить нажатие. Для этой реализации существуют три способа. Традиционным является присвоение слушателя объекту с помощью метода *setOnClickListener*, которому на вход идет наше текущее activity, которое обозначается словом *this:button. setOnClickListener(this)*. Далее в метода *onClick* методом *getId* получаем идентификатор элемента, который был нажат, а после перейдем к новому классу. Если имеется несколько кнопок, то целесообразно использовать конструкцию *switch…case*.

AndroidStudio поддерживает использование библиотек. Интересная библиотека *jcoup*, используется для так называемого парсинга и разбора *html*-кода. То есть получения данных с сайта в приложение. Так как в таком случае приложение должно подключатся к интернету,

то в файле манифеста необходимо заранее прописать пару строк, открывающих права доступа. Чтобы получить данные необходима также предварительно создать *layout*с помещенным в него *textView*, сюда мы будем получать данные с сайта. В файле *java* необходимо создать метод, который будет запрашивать данные, также добавить метод, в котором будет создан документ и используется конструкция *try…catch*, которая перехватывает ошибки. В блок *try* в документ поместить ссылку на сайт и название класса откуда берутся данные. После почистить наш *textView*, чтобы впоследствии заполнить. А далее в цикле *foreach*, аналог циклу *for*, поместить полученные данные в необходимый элемент.

Результаты проделанной работы можно посмотреть, запустив встроенный эмулятор android устройства, нажав на кнопку *run*, или воспользоваться имеющимся устройством, подключив его при помощи *usb*-кабеля.

УДК 324.679

Студ. Ю. С. Богдан
Науч. рук. доц. Н.А. Жиляк
(кафедра информационных систем и технологий, БГТУ)

ПРОГРАММНЫЙ КОМПЛЕКС ДЛЯ ТЕСТИРОВАНИЯ УЧАЩИХСЯ

Под тестами понимаются стандартизированные задания, предназначенные для проверки знаний, умений, профессиональной подготовки во всех областях.

Для любого образовательного процесса важен вопрос проверки знаний. Тесты получают все более широкое распространение в современном процессе образования. Точность и объективность результатов тестирования являются основой их применения.

На сегодняшний день тесты активно используются в сфере образования как универсальное средство проверки знаний и умений, профессиональной подготовки во всех областях. Тесты принято разделять на тесты по итогам обучения и тесты способностей.

Тесты по итогам обучения или тесты знаний оценивают степень усвоения школьной программы, информированности или сформированности навыков выпускников. Эти тесты в значительной степени ориентированы на проверку фактических знаний и навыков по отдельному предмету или группе предметов. Они призваны оценить и количество, и качество усвоенных знаний в соответствии с той программой, по которой изучался данный предмет. В связи с этим соста-