WK.7 / CH.7

❖ Location Sensors, Maps, and Fragments

- **Sensors** are hardware built in to the mobile device to allow an app to capture environmental data.
- **Maps** are used to display data that can be enhanced by a visual representation of its location.
- **Fragments** are a newer approach to coding Android Activities

1) Location Sensors

- Android devices typically have 2 location sensors:

Network Sensor	GPS Sensor
- based on cell towers and/or the Wi-Fi access points your device is connected to. - provides approximate location of the device. - faster	- based on a built-in (GPS) receiver. - provide accurate position information to within a few meters, depending on conditions. - Very slow in acquiring its position information - not all devices have a GPS sensor

- Location information is accessed within app through *LocationManager* object.
- Location information exists as an Android system service,and is accessed through the method *getSystemService.*
- The LocationManager object can request updates from either or both sensors.
- To get the updates, app has to instantiate a *LocationListener* . A LocationListener implements the method *onLocationChanged*
- **If sensor reports a location change:**

- change is captured by the *LocationListener* ,
- passed a *Location object*,
- and the *onLocationChanged* method is executed.

- **Location object** contains information on the new location, including:

  1- GPS coordinates and altitude

  2- which sensor provided the location,

  3- measure of the accuracy of the coordinate

- To begin receiving location information from the sensors, the LocationManager requests the updates from a specific provider and tells it what LocationListener to use to handle the updates.

- When the LocationManager requests updates from the GPS, the GPS is activated. However, the GPS must also be enabled by the user to be activated. If it has not been enabled, the code cannot turn it on.

- It is recommended the developer test if the GPS is enabled and inform the user if it is not.

2) MAPS

- Maps are implemented using the *GoogleMap* object in the *source code file* and a *MapFragment* in the *layout file*.

- These objects are not a part of the standard Android SDK but rather the Google Play Services SDK.

- Google Play Services SDK must be installed + Using Google Maps requires an API key.

- Maps are implemented as a *MapFragment* widget in a layout.

- The Activity that implements the code for the map's behavior is a *FragmentActivity*

3) Fragments

- The FragmentActivity is a subclass of the Activity class.
- An Activity that needs to implement a map must extend the FragmentActivity class . why?
- because maps are encapsulated in a MapFragment.
- This allows a map to be a part of a layout rather than the only thing in a layout.

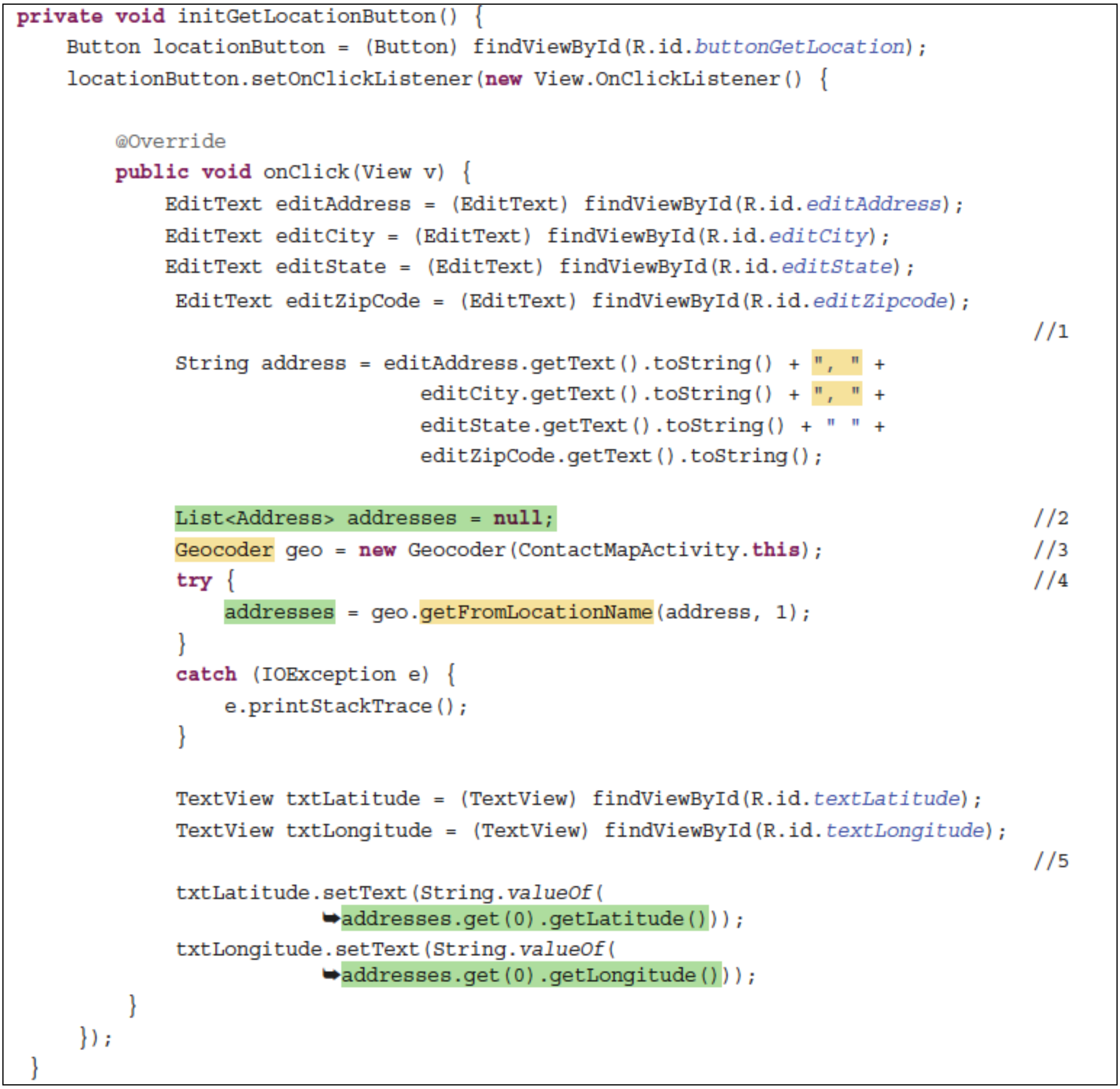
- permissions are used to alert the user during installation or upgrade
- Permissions are set in the Android manifest file.
- If a permission is required by what we are trying to do but not in the manifest, app will crash.
- Coarse permissions . give permission to use the network
- Fine permissions location give permission to use GPS listeners

❖ Finding your location (2 way)

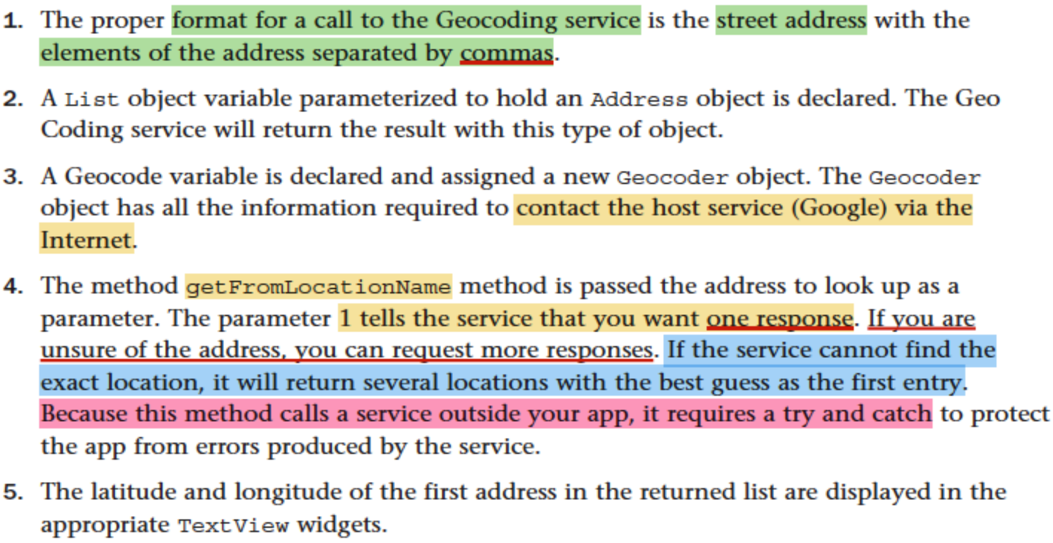
1) using a known location (address) and looks up the GPS coordinates via the Internet. (Geocoding)
2) using the device network and/or GPS sensors to locate the device in real-time.

- You can use the map's *getMyLocation()* method to get the device's current GPS coordinates.

➢ Code to Look Up Address Coordinates



```
private void initGetLocationButton() {
    Button locationButton = (Button) findViewById(R.id.buttonGetLocation);
    locationButton.setOnClickListener(new View.OnClickListener() {

        @Override
        public void onClick(View v) {
            EditText editAddress = (EditText) findViewById(R.id.editAddress);
            EditText editCity = (EditText) findViewById(R.id.editCity);
            EditText editState = (EditText) findViewById(R.id.editState);
             EditText editZipCode = (EditText) findViewById(R.id.editZipcode);
                                                                               //1
             String address = editAddress.getText().toString() + ", " +
                                   editCity.getText().toString() + ", " +
                                   editState.getText().toString() + " " +
                                   editZipCode.getText().toString();

             List<Address> addresses = null;                                   //2
             Geocoder geo = new Geocoder(ContactMapActivity.this);             //3
             try {                                                             //4
                 addresses = geo.getFromLocationName(address, 1);
             }
             catch (IOException e) {
                 e.printStackTrace();
             }

             TextView txtLatitude = (TextView) findViewById(R.id.textLatitude);
             TextView txtLongitude = (TextView) findViewById(R.id.textLongitude);
                                                                               //5
             txtLatitude.setText(String.valueOf(
                         ➥addresses.get(0).getLatitude()));
             txtLongitude.setText(String.valueOf(
                         ➥addresses.get(0).getLongitude()));
         }
     });
 }
```



1. The proper format for a call to the Geocoding service is the street address with the elements of the address separated by commas.

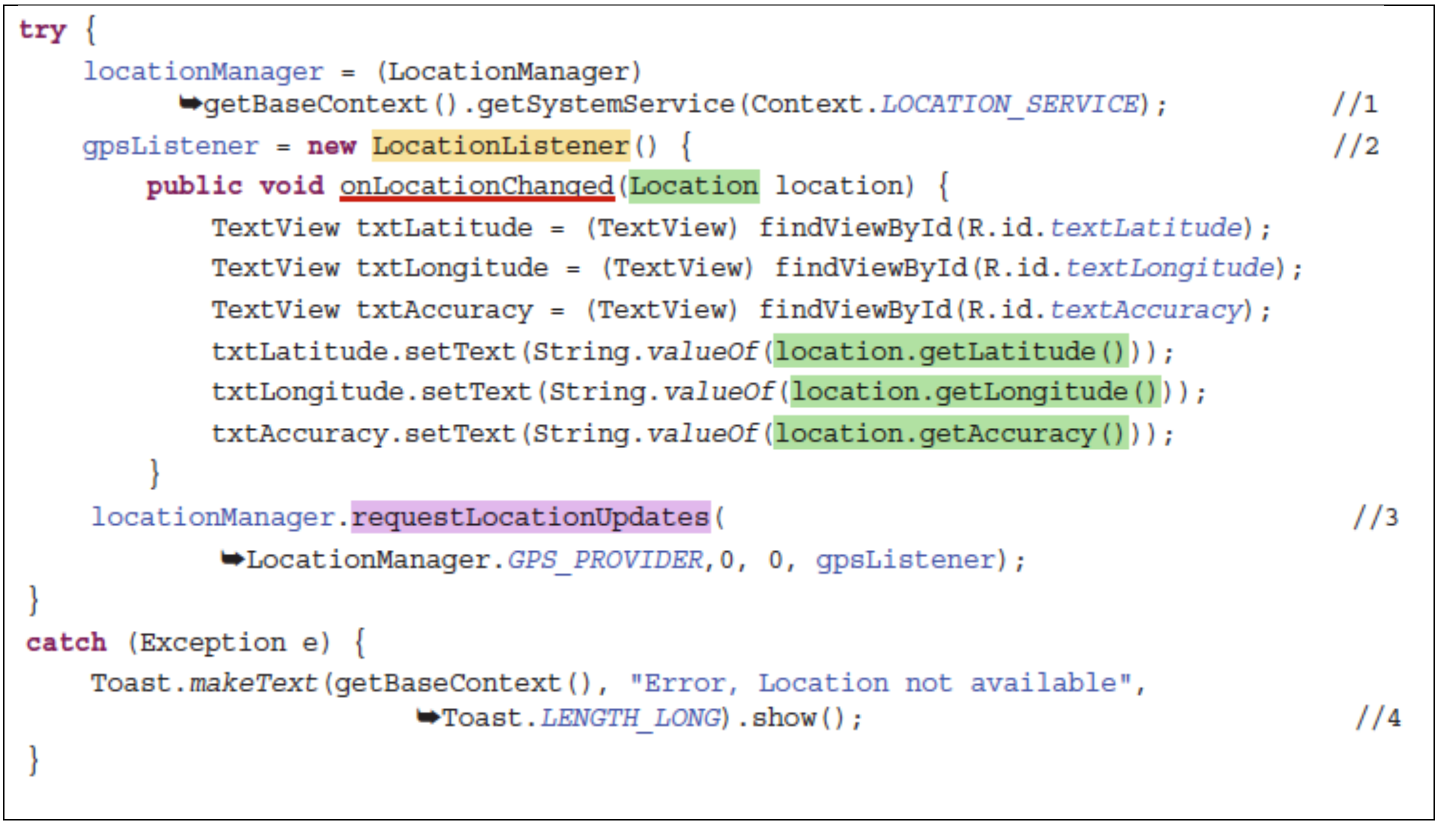
2. A `List` object variable parameterized to hold an `Address` object is declared. The Geo Coding service will return the result with this type of object.

3. A Geocode variable is declared and assigned a new `Geocoder` object. The `Geocoder` object has all the information required to contact the host service (Google) via the Internet.

4. The method `getFromLocationName` method is passed the address to look up as a parameter. The parameter 1 tells the service that you want one response. If you are unsure of the address, you can request more responses. If the service cannot find the exact location, it will return several locations with the best guess as the first entry. Because this method calls a service outside your app, it requires a try and catch to protect the app from errors produced by the service.

5. The latitude and longitude of the first address in the returned list are displayed in the appropriate `TextView` widgets.

➢ Code to Get Coordinates with the GPS Sensor



```
try {
    locationManager = (LocationManager)
          ➥getBaseContext().getSystemService(Context.LOCATION_SERVICE);          //1
    gpsListener = new LocationListener() {                                          //2
        public void onLocationChanged(Location location) {
            TextView txtLatitude = (TextView) findViewById(R.id.textLatitude);
            TextView txtLongitude = (TextView) findViewById(R.id.textLongitude);
            TextView txtAccuracy = (TextView) findViewById(R.id.textAccuracy);
            txtLatitude.setText(String.valueOf(location.getLatitude()));
            txtLongitude.setText(String.valueOf(location.getLongitude()));
            txtAccuracy.setText(String.valueOf(location.getAccuracy()));
        }
    locationManager.requestLocationUpdates(                                          //3
            ➥LocationManager.GPS_PROVIDER,0, 0, gpsListener);
}
catch (Exception e) {
    Toast.makeText(getBaseContext(), "Error, Location not available",
                        ➥Toast.LENGTH_LONG).show();                                  //4
}
```

TB

1. To use both the network and GPS location services, you must instantiate two LocationManagers. **F**

2. A LocationListener must be instantiated for the location sensor you want to get location data from. **T**

3. A location change identified by the location sensor is broadcast as an Intent with Extras. **F**

4. Location changes detected by the location sensors is reported as a Location object. **T**

5. The map used in Android is a widget available to any Activity. **F**

6. You need an Application Program Interface key to display maps on an Android device. T

7. You cannot use a standard Activity if you want to display a Map in an Android device. **T**

8. Location object reports latitude and longitude but not accuracy.which is reported by the listener directly. **F**

9. To use maps,you must implement code provided by Google that has nothing to do with map functionality **T**

10. Geocoding is the process of changing an address into GPS coordinates. **T**

11. Your app must always check to see that the new reported location is better than the last reported location.F

12. Markers on a map can contain a title but no other data. **F**

13. A map type (satellite, street) is set at map creation and cannot be changed. **F**

14. A change in accuracy is reported as a new Location. **T**

15. An app needs several permissions to use a Google map. **T**

16. You cannot get a device’s GPS location from a Google map. **F**

17. You cannot add multiple markers to a Google map. **F**

18. The standard marker on a GoogleMap object can be replaced with a custom image. **T**

1. Which of the following best describes a LocationManager?

A. An object that reports location changes to an app

B. A system service that can start and stop the network sensor but not the GPS sensor

C. A system service that can request location updates from the network and GPS sensors

D. An object that is subclassed in a MapFragment Activity

2. Which of the following best describes the onLocationChanged method?

A. A required method of a LocationListener that responds to any reported Location object

B. A required method of a LocationListener that responds only to Location objects that are newer

C. A method of the LocationManager that formats a Location object for use in an app

D. A method of the Location object that responds to the detection of a new location

3. Which of the following best describes a MapFragment?

A. A subclass of Activity

B. A GoogleMap object embedded in a layout file

C. A subclass of FragmentActivity

D. A dialog used to display a map

4. Where is your Android Map's API key embedded in the app?

A. The MapFragment

B. The GoogleMap object

C. The onCameraActivate method

D. The manifest file

5. Where do you get an Android Maps API key?

A. The Developer Console

B. The API Console

C. The Debug Keystore

D. The Google Keystore

6.Which of following is the correct variable declaration to hold the results of a call to the Geocoding service?

A. Coordinates returnedCoords

B. Address[] returnedAddresses

C. List<Address> returnedAddresses

D. Array[Address] returnedAddresses

7. Why should the call to the Geocoder service be protected with a try and catch statement?

A. Because the results of the call to the service are outside the control of the developer.

B. Because Geocoding is inherently error prone. There might not be a GPS location.

C. Because Geocoding is not in the Android SDK.

D. It should not be used with a try and catch.

8. Which of the following is the best description of a Location object?

A. An object that detects a change in location reported by the network or GPS sensor.

B. An object that is created when a location change is detected by the network or GPS sensor.

C. An object that responds to onLocationChanged.

D. Any object that contains location or Geocoding data.

9. What is a Toast?

A. An object created when the GPS or network sensors fail to detect any location

B. An object created when the sensors produce a location detection outside the bounds of the requestUpdates parameters

C. An object that reports errors through onLocationChanged

D. An object that displays a short message on the user screen for limited amount of time

10. Which of the following objects can turn off the GPS sensor?

A. LocationManager

B. LocationListener

C. LocationSensor

D. GPSManager

11. Which sensor produces the most accurate location determinations?

A. Network sensor

B. Wi-Fi sensor

C. Position sensor

D. GPS sensor

12. After your app sets MyLocationEnabled to true, what do you need to do to detect the device's location?

A. Nothing. Enabling this displays and reports GPS coordinates to the map.

B. Add a OnMyLocationChangeListener to the FragmentActivity.

C. Add the onMyLocationChanged method to the FragmentActivity.

D. Set a new LatLng point to track the device's location.

13. Which of the following is NOT a type of map that can be set for a GoogleMap object?

A. NORMAL

B. SATELLITE

C. TERRAIN

D. STREET

14. Which of the following is the method that is used to zoom the map to a specific location?

A. animateCamera

B. zoomTo

C. displayMap

D. zoomLevel

15. Which of the following is the correct method to add a marker to a GoogleMap object?

A. displayMarker

B. animateMarker

C. addMarker

D. addPinWithImage

16. Which of the following objects is used to collect a set of locations for display on a GoogleMap object?

A. LatLngSet

B. LatLngBounds.Builder

requestLocationUpdates	The method used to start listening for location changes from a location sensor.
LocationManager	A system service object that can request location updates from either location sensor.
GoogleMap object	An object in an SDK separate from the Android SDK.
FragmentActivity	An activity used to display multiple Activities on a single layout.
Location	An object that holds the data reported from a LocationListener.
Geocoding	A procedure to turn a street address into GPS coordinates.
onMyLocationChange	The method used to get location updates from a map.
isBetterLocation	A method not provided by the map, or any other object in the Android SDK.
LocationListener	An object that determines what an app will do when its location changes.
Snippet	Text that is displayed when the user touches a marker on a map.

WK.9 / CH.8

❖ Sensors

- The Android platform supports 12 sensors.
- sensors that measure the devices' ambient environment, temperature, humidity, atmospheric pressure, magnetic field, light level, sensors detect moving or rotating the device.
- **Sensor** class represents all types of sensors.
- Sensors are system service instantiated by OS not by an app.
- Sensors are accessed through the *SensorManager* class.
- SensorManager is a system service instantiated by OS and not by an app.
- Access is through Rreference created by calling the *getSystemService* method within an app.

- ***SensorEvent*** and ***SensorEventListener*** are also required to work with sensor.
- *SensorEvent* is an object created by a Sensor when it has something to report.
- SensorEvent object has information about the event:
  1. timestamp for when the object was created
  2. the sensor that produced the event,
  3. data that represents the sensor's measurements at the time of the event
- *SensorEventListener* is an interface implemented by any app that wants to use sensor information encapsulated in a SensorEvent

❖ Managers

- Manager monitor the status of the hardware ,Example (battery , storage , power manager) .

❖ Other hardware

- Android devices also have other hardware features, such as a phone and a camera.
- These devices have an app associated with them to provide access to their functionality.
- Hardware items are accessed by calling their Application Program Interface (API).
- The functionality of the hardware can also be accessed by integrating the features within the app by calling the associated app's API.

Listing 8.2 **Monitoring the Battery**

```
BroadcastReceiver batteryReceiver = new BroadcastReceiver() {                    //1
   @Override
   public void onReceive(Context context, Intent intent) {
       double batteryLevel= intent.getIntExtra(BatteryManager.EXTRA_LEVEL,0);     //2
       double levelScale= intent.getIntExtra(BatteryManager.EXTRA_SCALE,0);       //3
       int batteryPercent = (int) Math.floor(batteryLevel/levelScale*100);        //4
       TextView textBatteryState=(TextView)findViewById(R.id.textBatteryLevel);
       textBatteryState.setText(batteryPercent+"%");
   }
};

IntentFilter filter = new IntentFilter(Intent.ACTION_BATTERY_CHANGED);            //5
registerReceiver(batteryReceiver, filter);                                        //6
```

1. A `BroadcastReceiver` variable is declared and instantiated with a new `BroadcastReceiver`. This object receives `Intents` and has the code used to respond to the `Intent`. An `Intent` is broadcast from other apps or objects executing on the device.

2. The `Intent` concerning battery status sent by the OS contains information about the battery as `Extras`. This line gets the extra associated with the battery's current charge level. Although the value is retrieved as an integer, it is assigned to a double variable so that it can be used as a double later.

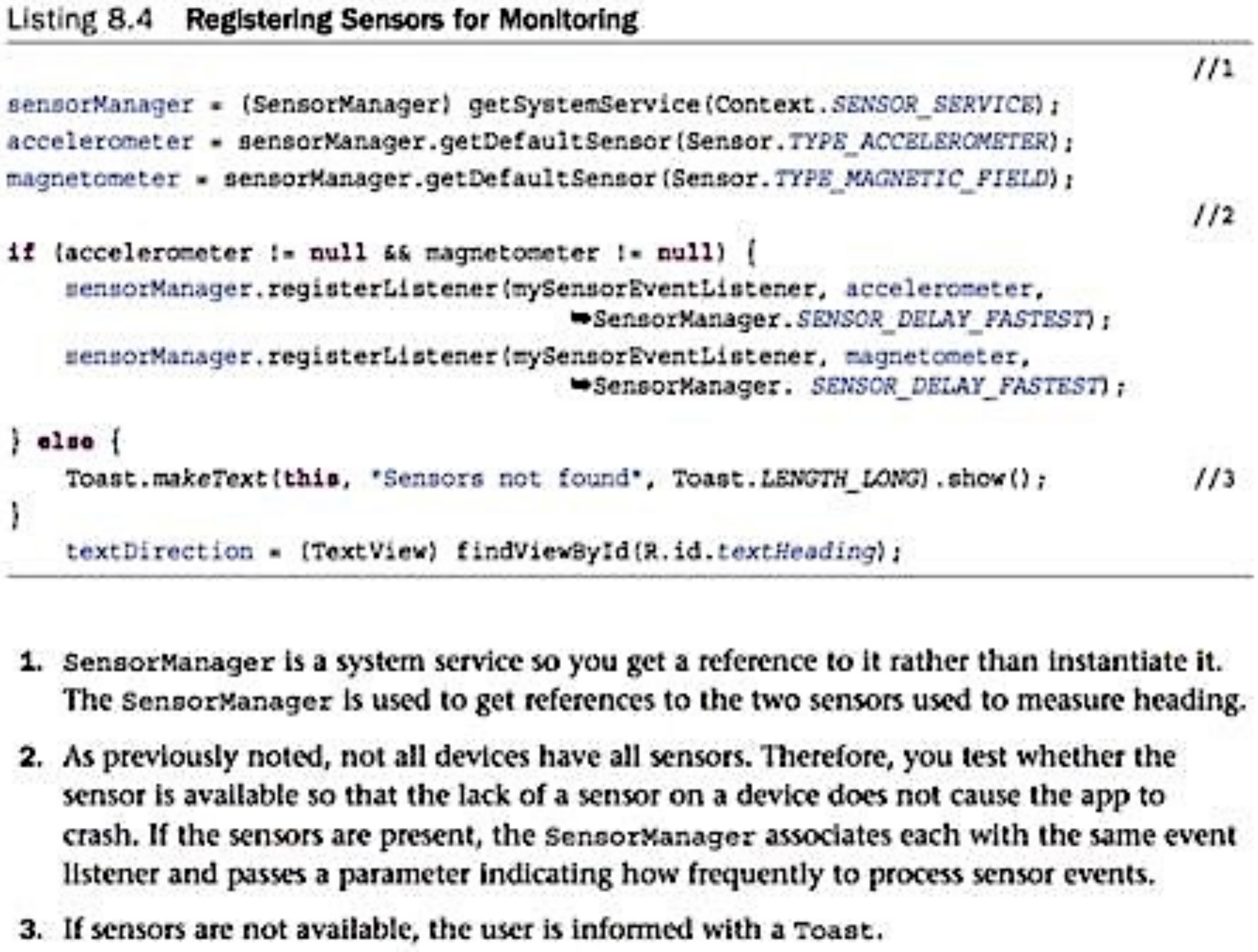
3. The extra associated with the scale used for measuring the charge is retrieved and assigned to a double variable. Capturing the scale is important because different devices may use different scales for measuring charge.

4. The percentage of battery charge left is calculated by dividing the level by the scale. If these two variables were not defined as doubles, this calculation would produce incorrect results because a divide operation needs to produce double value. The result of the calculation is a number between 0 and 1, which is multiplied by 100 to get a percentage. The floor function is applied to take on the integer value of the result.

5. A new `IntentFilter` variable is declared and assigned a new `IntentFilter`. An `IntentFilter` listens for `Intents` that have been broadcast by the system and only lets through the ones the developer is looking for. In this case, the filter looks for Battery Status changed intent. This is required because a `BroadcastReceiver` can respond to any intent. However, you want it to respond only to `Intents` sent by the battery.

6. The `BroadcastReceiver` is registered, which means that the app is told to listen for battery status intents and handle them with the `BroadcastReceiver` defined in the activity.

2. Using Sensors to Create a Compass



Listing 8.4 **Registering Sensors for Monitoring**

```
                                                                                    //1
sensorManager = (SensorManager) getSystemService(Context.SENSOR_SERVICE);
accelerometer = sensorManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER);
magnetometer = sensorManager.getDefaultSensor(Sensor.TYPE_MAGNETIC_FIELD);
                                                                                    //2
if (accelerometer != null && magnetometer != null) {
    sensorManager.registerListener(mySensorEventListener, accelerometer,
                                              ➥SensorManager.SENSOR_DELAY_FASTEST);
    sensorManager.registerListener(mySensorEventListener, magnetometer,
                                              ➥SensorManager. SENSOR_DELAY_FASTEST);
} else {
    Toast.makeText(this, "Sensors not found", Toast.LENGTH_LONG).show();            //3
}
    textDirection = (TextView) findViewById(R.id.textHeading);
```

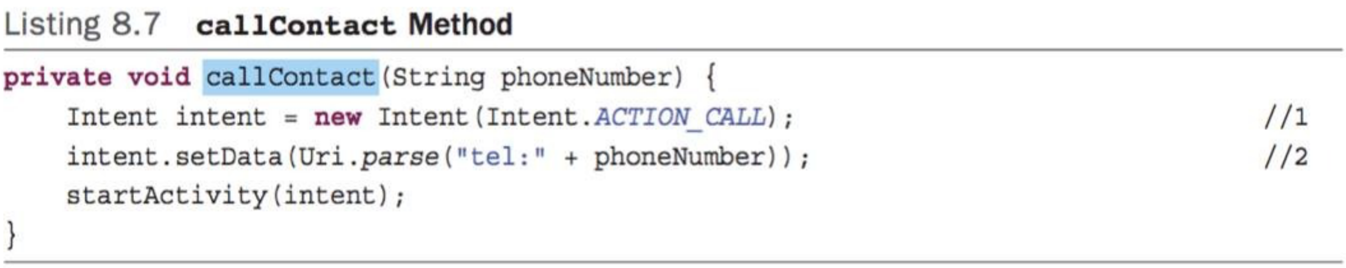
1. `SensorManager` is a system service so you get a reference to it rather than instantiate it. The `SensorManager` is used to get references to the two sensors used to measure heading.
2. As previously noted, not all devices have all sensors. Therefore, you test whether the sensor is available so that the lack of a sensor on a device does not cause the app to crash. If the sensors are present, the `SensorManager` associates each with the same event listener and passes a parameter indicating how frequently to process sensor events.
3. If sensors are not available, the user is informed with a `Toast`.

Listing 8.5 **SensorEventListener Code**

```
private SensorEventListener mySensorEventListener = new SensorEventListener() {

    public void onAccuracyChanged(Sensor sensor, int accuracy) {  }              //1

    float[] accelerometerValues;                                                 //2
    float[] magneticValues;

    public void onSensorChanged(SensorEvent event) {                             //3
    if (event.sensor.getType() == Sensor.TYPE_ACCELEROMETER)
        accelerometerValues = event.values;
    if (event.sensor.getType() == Sensor.TYPE_MAGNETIC_FIELD)
        magneticValues = event.values;
    if (accelerometerValues!= null && magneticValues!= null) {                   //4
        float R[] = new float[9];
        float I[] = new float[9];
        boolean success = SensorManager.getRotationMatrix(R, I,
                                       ➥accelerometerValues, magneticValues);
        if (success) {                                                           //5
            float orientation[] = new float[3];
            SensorManager.getOrientation(R, orientation);
            float azimut = (float) Math.toDegrees(orientation[0]);                //6
        }
```

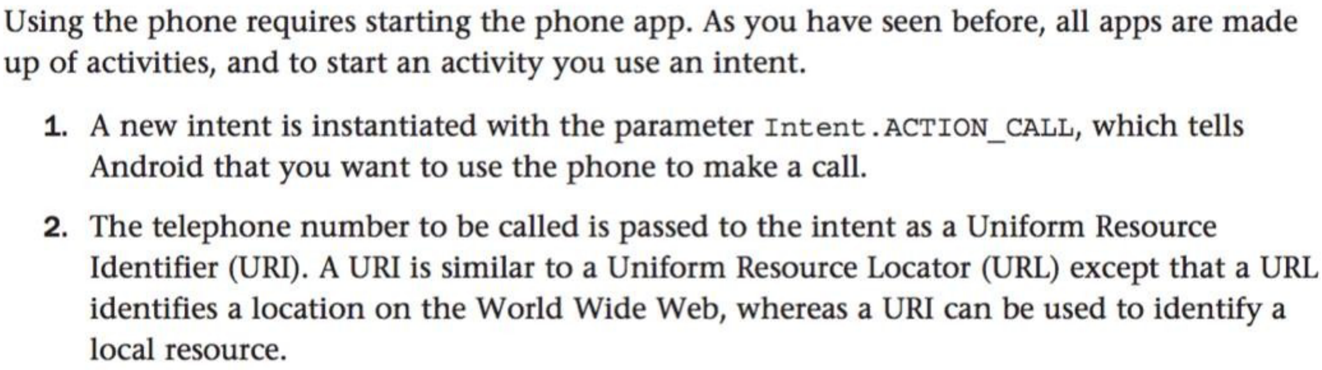
1. A `SensorEventListener` requires the implementation of two events, `onAccuracyChanged` and `onSensorChanged`. To calculate a heading, you don't need accuracy, so its method block is empty.
2. Sensor readings are returned as a `float` array. Two variables to hold the response from each sensor are declared.
3. The `onSensorEvent` first determines which sensor triggered the event and then captures the values it provided.
4. If there are values available for both sensors, the `SensorManager` is asked for two rotational matrices used for orientation calculation. Discussion of the rotational matrices is beyond the scope of this book.
5. If the matrices are successfully calculated, the `SensorManager` is asked to calculate the orientation of the device. Orientation is measured in three dimensions.
6. The first orientation measure is the value used to calculate the heading. It is reported in radians, so these are changed to degrees.

3. Using the Phone



Listing 8.7 **callContact Method**

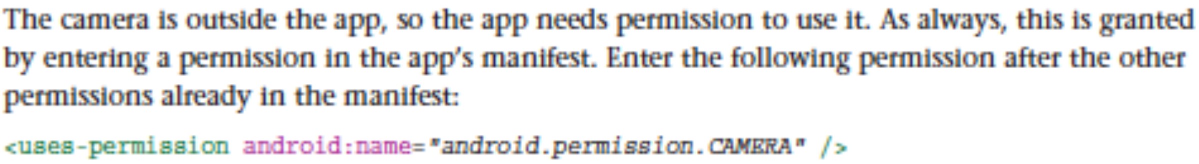
```
private void callContact(String phoneNumber) {
    Intent intent = new Intent(Intent.ACTION_CALL);                            //1
    intent.setData(Uri.parse("tel:" + phoneNumber));                           //2
    startActivity(intent);
}
```



Using the phone requires starting the phone app. As you have seen before, all apps are made up of activities, and to start an activity you use an intent.

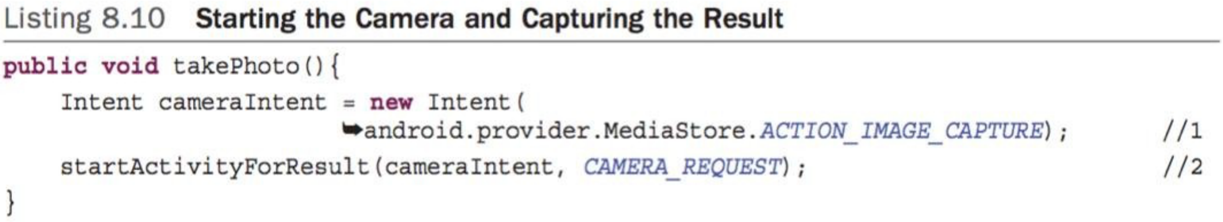
1. A new intent is instantiated with the parameter `Intent.ACTION_CALL`, which tells Android that you want to use the phone to make a call.
2. The telephone number to be called is passed to the intent as a Uniform Resource Identifier (URI). A URI is similar to a Uniform Resource Locator (URL) except that a URL identifies a location on the World Wide Web, whereas a URI can be used to identify a local resource.

4. Using the Camera



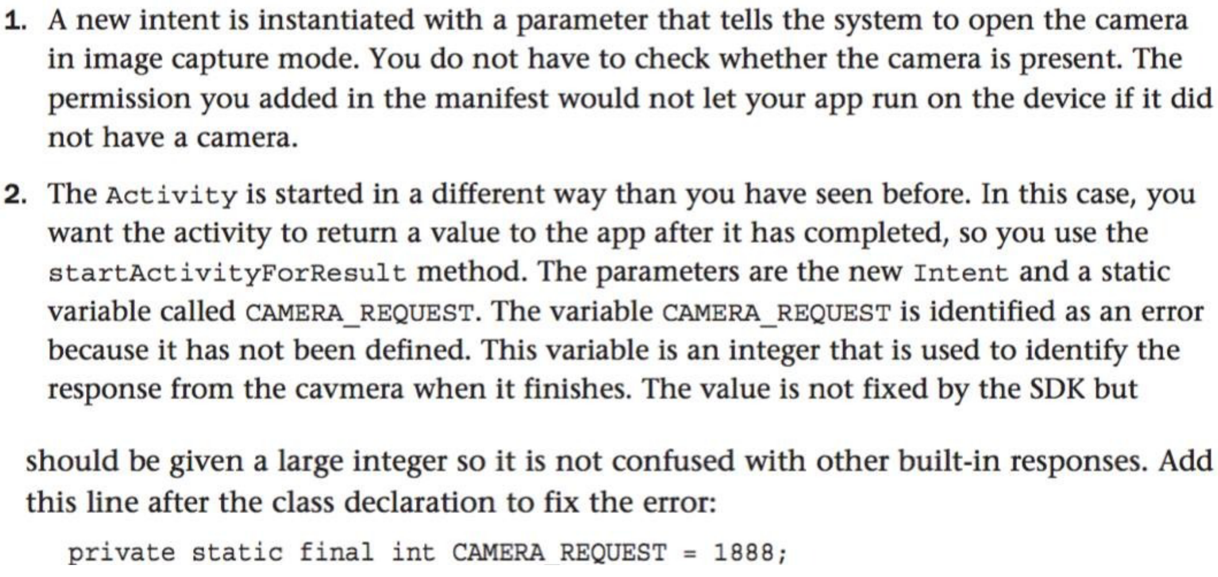
The camera is outside the app, so the app needs permission to use it. As always, this is granted by entering a permission in the app's manifest. Enter the following permission after the other permissions already in the manifest:

```
<uses-permission android:name="android.permission.CAMERA" />
```



Listing 8.10 **Starting the Camera and Capturing the Result**

```
public void takePhoto(){
    Intent cameraIntent = new Intent(
                       ➥android.provider.MediaStore.ACTION_IMAGE_CAPTURE);     //1
    startActivityForResult(cameraIntent, CAMERA_REQUEST);                      //2
}
```



1. A new intent is instantiated with a parameter that tells the system to open the camera in image capture mode. You do not have to check whether the camera is present. The permission you added in the manifest would not let your app run on the device if it did not have a camera.
2. The `Activity` is started in a different way than you have seen before. In this case, you want the activity to return a value to the app after it has completed, so you use the `startActivityForResult` method. The parameters are the new `Intent` and a static variable called `CAMERA_REQUEST`. The variable `CAMERA_REQUEST` is identified as an error because it has not been defined. This variable is an integer that is used to identify the response from the cavmera when it finishes. The value is not fixed by the SDK but should be given a large integer so it is not confused with other built-in responses. Add this line after the class declaration to fix the error:

```
private static final int CAMERA_REQUEST = 1888;
```

TB

1. All hardware features on an Android device are accessed through a Manager. : False
2. The Android platform supports about 12 different sensors. : True
3. A SensorEvent is an object created by the SensorEventListener. : False
4. The SensorManager object is never instantiated. It is a system service. : True
5. Access to the phone and camera is made through calls to their API. : True
6. Accessing information about the device's battery requires use of a SensorManager. : False
7. The SensorManager broadcasts every few seconds with an update on the sensor's status. False
8. An IntentFilter is used with a BroadcastReceiver to limit the Intents that it responds to. : True
9. BroadcastReceivers must be registered with the appropriate Manager to receive broadcast Intents. True
10. To use a sensor, you must instantiate a SensorEventListener. : True
11. An app requires a permission in its manifest to use the Camera but not the Phone. : False
12. To use the phone, you must first create an Intent. : True

1. What is a sensor?
A. Any hardware resource on an Android device
B. A device that reports the status of hardware components, such as the battery, on an Android device
C. A hardware device that captures information about the Android device's environment
D. A hardware device that is used by the Android device to interact with its environment or other devices

2. Which of the following best describes a SensorManager?
A. A system service used by an app to interact with a sensor
B. An object that can send Intents to update sensor readings
C. An object that can receive Intents with status updates from sensors
D. An object used to manipulate sensor capabilities

3. Which of the following best describes a BatteryManager, Power Manager, and StorageManager?
A. Objects that produce the appropriate event when status changes (for example, BatteryEvent).
B. System services that monitor the status of hardware on the Android device
C. Objects that listen for hardware status changes
D. Objects that are instantiated by an app to monitor the specific device

4. Which of the following objects limits what a BroadcastReceiver can respond to?
A. EventFilter
B. ReceivedAction
C. ActionResponder
D. IntentFilter

5. Which of the following objects is used to listen for Intents sent from inside or outside an app?
A. IntentFilter
B. ActivityFilter
C. BroadcastReceiver
D. IntentReceiver

6. Which of the following is the best description of the Extras object?
A. An object used to identify all the sensors on an Android device
B. An object that is used to send data with an Intent
C. An object used to expand the events that a BroadcastReceiver can respond to
D. An object used to capture Sensor status changes

7. Which of the following is the correct code to get a reference to an Android device's proximity sensor?

A. sensorManager.getSystemService(TYPE_PROXIMITY)
B. getSystemService(Context.PROXIMITY_SENSOR)
C. getSystemService(Sensor.TYPE_PROXIMITY)
D. sensorManager.getDefaultSensor(Sensor.TYPE_PROXIMITY)

8. Which of the following best describes a SensorEventListener?

A. A private class instantiated in an app to respond to SensorEvents
B. A system object used to respond to SensorEvents
C. A sensorManager method used to respond to SensorEvents
D. An object that responds to Intents sent by a Sensor

9. Which of the following is the correct code to call a phone number from within an app?

A. startActivity(Intent.ACTION_CALL) ;
B. startActivity(new Intent(Uri.parse(phoneNumber))) ;
C. startActivity(new Intent(Intent.ACTION_CALL).setData(Uri.parse("tel :"+phoneNumber))) ;
D. startIntent(new Intent(Uri.parse("tel :"+phoneNumber))) ;

10. What is the difference between a URL and a URI?

A. A URL is used only with a web browser, whereas a URI can be used only locally.
B. A URL identifies a location on the Web, whereas a URI can be used to identify a local resource.
C. A URL identifies a resource anywhere, whereas a URI can only be used to access a resource external to an app.
D. No difference. They can be used interchangeably.

11. What is the function of the requestCode when you execute the method startActivityForResult?

A. It is used by the onActivityResult method to determine if the result it received is one it should respond to.
B. It is used by the IntentFilter to identify which service to start.
C. It is used by the Android OS to determine which service to start.
D. It is used by the Android OS to determine if the app has permission to start a service.

12. Which of the following data is needed to start an Activity for a result?

A. The service to start and a timestamp for the request
B. An Intent that identifies the service to start and the BroadcastReceiver that will respond to the result
C. The name of the service to start and a result Intent
D. An Intent that identifies the service to start and a unique code to identify the result

13. Which of the following is the SQLite data type used to store a picture?

A. bytearray
B. blob

C. bytestream
D. bitmap

14. A variable that holds an image provided by the camera should be declared as what type?

A. Image
B. ImageButton
C. Bitmap
D. ByteArray

15. Before an image can be stored in the SQLite database, it must convert to a _____?

A. ByteArray
B. Bitmap
C. Blob
D. ImageMap

SensorEvent	D. An object produced when the status of the accelerometer changes.
getSystemService	I. A method used to get a reference to a SensorManager.
BroadcastReceiver	A. An object that can receive Intents from activities.
SensorEventListener	G. An object that is registered with a Manager to respond to status changes of a system service.
URI	F. An object used to identify a local resource.
onActivityResult	B. A method executed when an external resource completes its requested work.
onSensorChanged	H. A method executed when a sensor reading changes.
setData	J. A method used to tell an Intent what it wants to do.
ByteArrayInputStream	C. An object used to read a blob from storage.
Bitmap	E. An object used to hold a picture.

WK.10 / CH.9

❖ IOS & Xcode

- **iOS SDK** is the software development kit that allows application programs to utilize classes and frameworks provided by the SDK.
- iOS is multitasking and runs on different devices (iPhones, iPod , iPads, and Apple TVs).
- Apple provides an integrated development environment (IDE) called Xcode.
- Xcode is the IDE used by iOS &OS X developers.
- Xcode provides an interface to the compiler, editor, debugger, and code profiling tools.
- The language used to develop software for iOS is Objective-C.

❖ Content of IOS project file :

1. **AppDelegate.h and AppDelegate.m**
   They manage issues related to the entire app. Primarily used to manage the life cycle of the app.

2. **Main.storyboard**
   Used to design the interaction between multiple screens in your app + designing the layout of the individual screens.

3. **ViewController.h** and **ViewController.m**
   They contains the code that controls the user interactions with the app.

4. **Images.xcassets**
   Contains all the images, including icons, needed for your app.

5. **Supporting Files**
   a. appname-Info.plist : contains a few app-specific settings.
   b. main.m: responsible for launching the app.

6. **Frameworks**
   various libraries that you can include in your project to add functionality to your app.

7. **Products**
   This is your compiled app file.

❖ Action & Outlet :

Action & outlet are ways (connection) by which view controller will interact with the view.

Action is connection between user element & the code with a method that is called when a certain event happens.
We use it when we want to detect a trigger (i.e when button is pressed)

Outlet is connection between UI element & the code allowing for referencing the element in code.
We use it when we want to change some property of control (i.e text color,text size)

TB

1. Eclipse is the IDE most commonly used to develop iOS apps. : False
2. A Universal iOS app is one that can run on iPads, iPhones, and iPod touches. : True
3. Wiring up an outlet will add a method to the underlying code. : False
4. Wiring up an action will add a method to the underlying code. : True
5. The user interface for an iOS app is designed primarily in code. : False
6. iOS apps should have the same icon in different sizes. : True
7. The keyboard that shows up when the user types in text can be configured to fit the type of data the user is expected to enter. : True
8. The launch image is recommended to contain information about the developers of the app. : False
9. The app delegate contains graphics for use in the app. : False
10. A Text Field is used to allow the user to enter text using the onscreen keyboard. : True

1. A Universal iOS app can run on which of the following types of devices?
A. iPad
B. iPhone
C. iPod touch
D. All of the above

2. What is the Storyboard used for?
A. Writing the program logic for the app
B. Designing the user interface for the app
C. Uploading the app to the App Store
D. Describing the functionality of the app

3. Which type of file contains the program logic for a portion of an app?
A. Main.Storyboard
B. Images.xcassets
C. ViewController.m
D. AppDelegate.m

4. How do you limit which iOS devices can download your app?
A. By setting the target device setting in the app's Plist file.
B. By changing the Deployment Target setting on the Project Summary screen.
C. By setting the app's minimum Scheme number.
D. Not possible. All iOS devices can download apps from the App Store.

5. What is the effect of changing the Scheme within Xcode?
A. Changes the minimum iOS version that the app will target
B. Determines which set of icons to use in the app
C. Changes the look and feel of the app
D. Changes the device used by the simulator

6. Which of the following UI objects is used when you want a user to be able to enter data via the keyboard?
A. Label
B. Text Field
C. EditText
D. Text Box

7. Which of the following UI objects is used when you want to display static text on a screen?
A. Label
B. Text Field
C. LabelText
D. Outlet

8. What is the primary purpose of an outlet in an iOS app?
A. This is a place where apps are sold at a discount.
B. Provides a name for a user interface element that can be used in code.
C. It's a way to provide feedback to the user about what the app is doing.
D. The method that is called when the user interacts with a UI control.

9. What is the primary purpose of an action in an iOS app?
A. Provides a name for a user interface element that can be used in code
B. A general term for an Obective-C method
C. The method that is called when the user interacts with a UI control
D. A name for any user interaction with a UI control

10. What does it mean that a device is equipped with a retina screen?
A. Nothing. It's just a marketing term.
B. The screen has an anti-glare coating.
C. The programmer has to use points instead of pixels when determining how to lay out the user interface.
D. The screen resolution is doubled in both directions.

11. Which of the following is NOT a graphic asset that is used with an iOS app?
A. Launch image
B. App Icon
C. Profile picture
D. App Store icon

12. When wiring up an outlet, what happens in the underlying code?
A. A property is added to the code.
B. A method is added to the code.
C. It reformats the indentation in the code file.
D. Nothing. It is only related to the user interface.

13. Which of the following is not a type of keyboard available in iOS apps?
A. Number Pad
B. Email Address
C. Phone Number
D. Voice Entry

14. A launch image in the resolution 640x1136 should be supplied for which type of device?
A. iPad Air
B. iPad Mini
C. iPhone 5
D. iPod touch 4th generation

15. What is Apple's recommendation about the content of the launch image?
A. A splash screen with a graphic that represents the app.
B. A blank version of the app's first screen.
C. Information about the developers of the app and the app version number.
D. Nothing. A launch image is not recommended by Apple.

Outlet	A connection between a user interface element and the code allowing for referencing the element in code.
Action	A connection between a user element and the code with a method that is called when a certain event happens.
Storyboard	A tool used to design the interaction between multiple screens in an app.
Retina screen	A screen technology where the resolution has been doubled in both directions.
Label	A user interface element used to display static text.
Text View	A user interface element allowing for input from the keyboard.
Scheme	An identification of a particular device to be used for executing an app.
Universal app	An app that is targeted at both iPhone and iPad.
Asset Catalog	A container for graphical images used in the app.
Launch image	An image that is displayed when the app is launching.

WK.11 / CH.10

❖ Views and Controllers

- UI classes in iOS are contained in the UIKit framework.
- UIKit framework classes arranged in an inheritance hierarchy with the top class being UIView
- **UIView** class

  describes a basic rectangle with width, height, background color, and contain subviews that has a parent view.
- **UIKit > UIView** > UIWindow
  - which has been restricted to set its origin to the top left of the screen.
  - Each iOS app has one UIWindow object created when the app launches.
  - All other screens are subviews of UIWindow .
- **UIKit > UIView** > UILabel

  UIScrollView

  UINavigationBar

  UITableCell

  UIControl > UIButton

  UITextField

  UISlider

❖ View Control

View Controller managed three files:

.storyboard specifies the layout of user interface elements.

.h has information about outlets and actions needed to control the UI.

.m contains the implementation of the UI actions.

❖ Tab Bar Controller

shows up at the bottom of iPhone apps to allow user to choose Between different screens in app.

❖ Navigation Controller

used to allow the user to drill down through multiple screens while keeping track of the path so user can go back the same way.

❖ Segues

Is the connections between the view controllers which define how screens will transition back and forth.

TB

1. The Tab Bar Controller shows up at the bottom of iPhone apps and allows the user to choose between different screens in the app. : True
2. All user interface elements must be wired to an outlet. : False
3. The Navigation Controller is used to allow the user to drill down through multiple screens while keeping track of the path so the user can later go back the same way. : True
4. All user interface screens in an iOS app are described in the storyboard. : False
5. The Attributes Inspector is used to specify connections between code and user interface elements. : False
6. The Connections Inspector is used to specify connections between code and user interface elements. True
7. Images used for the Tab Bar icons must be exactly 20x20 pixels. : False
8. A Scroll View can allow the user to zoom in on an image. : True
9. A Date Picker can be used to choose a time. : True
10. A Scroll View can be used to dismiss the keyboard from the screen. : True

1. Which of the following allows the user to choose from different screens in an app?

A. Navigation Controller
B. Tab Bar Controller
C. Screen Controller
D. View Controller

2. Which of the following allows the user to move through multiple screens and back again the same way?

A. Navigation Controller
B. Tab Bar Controller
C. Path Controller
D. Table Controller

3. After adding a new View Controller to the Storyboard, how do you add it into the Tab Bar?

A. Specify the connection in the Connection Inspector.
B. Control-drag from the Tab Bar to the new View Controller.
C. Drag a segue from the toolbox to connect the two.
D. Control-drag from the new View Controller to the Tab Bar.

4. How do you change the title of a View Controller?

A. Add this code to viewDidLoad: `viewController.title = "<Title of Controller>";`
B. Change the Title property in the Connections Inspector.
C. Double-click the Title in Interface Builder and type a new one.
D. Change the Title property in the Attributes Inspector.

5. Why is it recommended to NOT use the system icons for Tab Bar icons?

A. It's not allowed by Apple.
B. It allows for higher resolution than the icons Apple provides.
C. It allows for a greater variety than the icons Apple provides.
D. Trick question: You can't add icons to the Tab Bar.

6. What's the difference between 1x and 2x graphics?

A. 2x are for iPad and 1x are for iPod touch and iPhone.
B. 2x are used for retina screens and are supposed to be twice the resolution of 1x.
C. 2x are more expensive versions of the graphics.
D. 2x allows the device to run faster than the 1x files.

7. How do you add multiple controls to a Scroll View?

A. Select all the controls and control-drag onto the Scroll View.

B. Use the Connections Inspector to specify which controls to include in the Scroll View.

C. Use the Assistant Editor and control-drag into the ScrollView.h file.

D. Select all the controls and then choose Editor > Embed In > Scroll View.

8. What is the content size used to specify for a Scroll View?

A. The size of the widest control included in the Scroll View

B. The size of the rectangle that is scrolled

C. The size of the rectangle that encompasses all the controls in the Scroll View

D. The size of the screen

9. Which of these is NOT a valid way to use a Scroll View to dismiss the keyboard?

A. Dismiss interactively

B. Dismiss on drag

C. Dismiss on tap

10. How do you add a View Controller to a Navigation Controller using Storyboard (in an app with no pre-existing Navigation Controllers)?

A. Control-drag from the View Controller to the Navigation Controller.

B. Use the Connections Inspector to specify that the View Control should be embedded in the Navigation Controller.

C. Select the View Controller and choose Editor > Embed In > Navigation Controller.

D. Drag a Navigation Controller from the toolbox and drop it on top of the View Controller.

11. How do you add a View Controller to a preexisting Navigation Controller using Storyboard?

A. Control-drag from a control on one controller to the second controller and choose Push on the pop-up menu.

B. Control-drag from a control on one controller to the .h file for the other controller and add an Action to Push the controller.

C. Select the View Controller and choose Editor > Embed In > Navigation Controller.

D. Use the Connections Inspector to specify that the View Control should be embedded in the Navigation Controller.

12. Which of the following is the correct method for disabling a Text Field called txtField?

A. [txtField.disable()]

B. [txtField setEnabled:NO]

C. [txtField setDisabled:YES]

D. [txtField.enabled(NO)]

13. How is the maximum date value set for a Date Picker?

A. Use the Size Inspector to set the Maximum Date Constraint.

B. It can't be set.

C. Control-drag from the Date Picker to the Date Controller and choose Maximum Date in the pop-up menu.

D. Use the Attributes Inspector to set the Maximum Date Constraint.

14. Which of these classes can be used as the basis for allowing the user to drag content horizontally and vertically on the screen?

A. UIDragView

B. UIView

C. UIScrollView

D. UISlider

15. What is the Dock used for in Xcode?

A. Choosing a specific user interface element to work with.

B. It holds all the user interface elements that can be dragged into the app.

C. It has the list of all the files in the project.

D. It's the place where you control execution of the app.

Attributes Inspector	H. Lets the developer set specific properties for a user interface element.
Segmented Control	G. Allows the user to indicate a choice among several predetermined options.
Connection Inspector	C. In this area of Xcode, the developer can manage how user interface elements are wired to code.
Navigation Controller	A. Allows the user to move through multiple screens and return the way they came.
Scroll View	I. Allows several controls to be moved horizontally or vertically around the screen by the user.

UIKit	J. The framework containing all user interface classes.
Tab Bar Controller	Allows the user to choose from multiple screens.
Date Picker	A control resembling several rotating wheels.
View Controller	Used to manage a single screen in an iOS app.
UIView	The most generic user interface class.

WK.12 / CH.11

❖ Persistent data

1) File Data Storage

- iOS enables saving data in files in regular text files or by archiving
- Apps are **sandboxed**, which means that each app is isolated from the other apps and from the operating system.
- Each app has simple file system that by default consists of: **Documents**, **Library**, and **tmp**
- Developer, can store files in the *Documents* and *tmp* folder, while read only *library* folder.
- The Documents folder is backed up when the device is backed up. The tmp folder is not.

Comparison between android and iOS File data storage:

IOS	ANDROID
store files in sequential file format or file storage	store files in serialization (archiving format)
store files in 3 folders, Library, Documents ,tmp.	Files written to internal or external storage.
Files private to app	Files private to app - stored internal storage Files are not private - stored external storage

2) User Defaults

- When you need to save a little bit of data in your app, the *NSUserDefaults* object is a very simple and easy way to do so.
- *NSUserDefaults* is a front-end to a key-value file (often referred to as Plist files because of the .plist extension) that is stored in the app's Preferences directory.
- There's only one file, but in this file you can store as many values as you want.
- When storing values in the file, you have to supply a key string to identify the value when retrieving it later.
- You can store many data types in NSUserDefaults (NSData , NSString , NSNumber ,NSDictionary , NSArray) Other data types can be archived and stored as an NSData object.
- NSUserDefaults = shared preferences in android

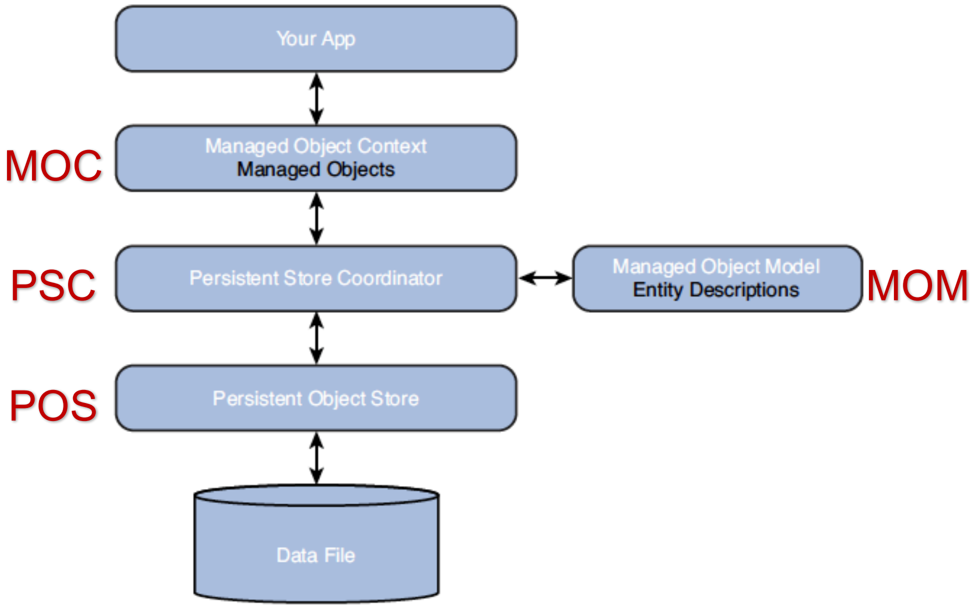
Ex. Write a code to store , retrieve and save the following values by using NSUserDefaults.

Key	value
refreshTimeOut	15
username	Hazazi
isAdmain	True

```
NSUserDefaults *settings = [NSUserDefaults standardUserDefaults];

[settings setInteger:15 forKey:@"refreshTimeOut"];
[settings setObject:@"Hazazi" forKey:@"username"];
[settings setBool:YES forKey:@"isAdmain"];

NSInteger *name1 = [settings integerForKey:@"refreshTimeOut" ];
NSObject  *name2 = [settings objectForKey:@"username" ];
Bool          *name3 = [settings boolForKey:@"isAdmain" ];

[settings synchronize];
```

3) Core Data

- Core Data is a powerful data persistence solution developed by Apple to provide object-oriented storage.



DATA FILE:
The framework stores data in files. The default is SQLite database, but you can also choose to use XML or binary data.

PERSISTENT OBJECT STORE:
Wraps around the data file and presents a common interface to the rest of the stack.

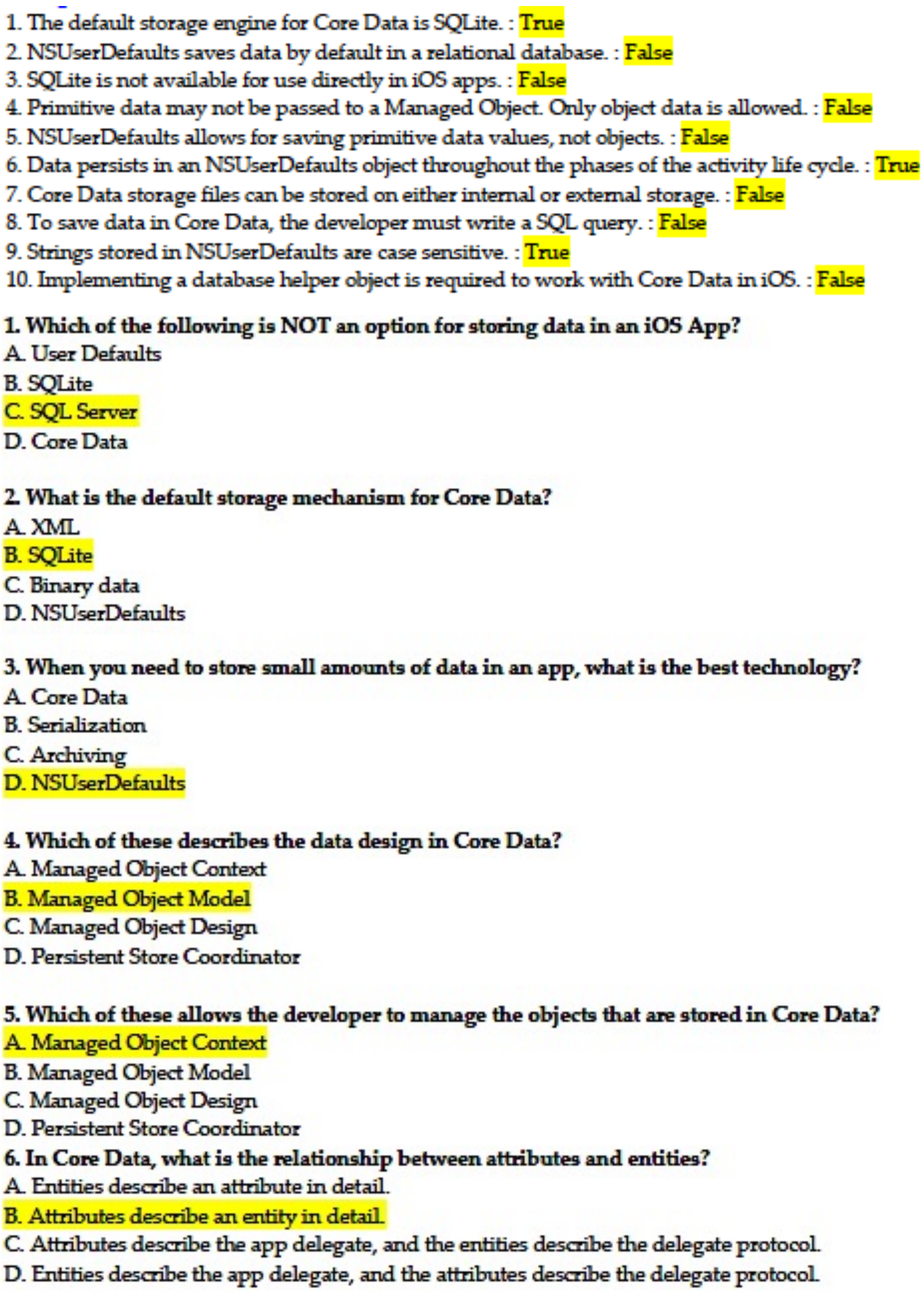
THE PERSISTENT STORE COORDINATOR:
Allows for having multiple data stores in the same app, and will then coordinate access to those stores.

THE MANAGED OBJECT MODEL:
Is where the description of the layout of the data is defined.

THE MANAGED OBJECT CONTEXT:
Which allows you to access the objects that are stored in the data file.

The Managed Object Context can keep track of multiple changes to the objects, and will periodically, or when instructed, save the changes to the persistent store.

TB



1. The default storage engine for Core Data is SQLite. : True
2. NSUserDefaults saves data by default in a relational database. : False
3. SQLite is not available for use directly in iOS apps. : False
4. Primitive data may not be passed to a Managed Object. Only object data is allowed. : False
5. NSUserDefaults allows for saving primitive data values, not objects. : False
6. Data persists in an NSUserDefaults object throughout the phases of the activity life cycle. : True
7. Core Data storage files can be stored on either internal or external storage. : False
8. To save data in Core Data, the developer must write a SQL query. : False
9. Strings stored in NSUserDefaults are case sensitive. : True
10. Implementing a database helper object is required to work with Core Data in iOS. : False

1. Which of the following is NOT an option for storing data in an iOS App?
A. User Defaults
B. SQLite
C. SQL Server
D. Core Data

2. What is the default storage mechanism for Core Data?
A. XML
B. SQLite
C. Binary data
D. NSUserDefaults

3. When you need to store small amounts of data in an app, what is the best technology?
A. Core Data
B. Serialization
C. Archiving
D. NSUserDefaults

4. Which of these describes the data design in Core Data?
A. Managed Object Context
B. Managed Object Model
C. Managed Object Design
D. Persistent Store Coordinator

5. Which of these allows the developer to manage the objects that are stored in Core Data?
A. Managed Object Context
B. Managed Object Model
C. Managed Object Design
D. Persistent Store Coordinator

6. In Core Data, what is the relationship between attributes and entities?
A. Entities describe an attribute in detail.
B. Attributes describe an entity in detail.
C. Attributes describe the app delegate, and the entities describe the delegate protocol.
D. Entities describe the app delegate, and the attributes describe the delegate protocol.

7. What is the purpose of the delegate pattern?

A. Delay execution of certain tasks until later.

B. Allow one object to act on behalf of another object to speed up execution.

C. Allow one object to call methods in another object without knowing at design-time which object those methods would be available in.

D. It provides a very nice texture for user interface controls.

8. What is the purpose of a segue in an iOS app?

A. It handles switching between two tabs in the Tab Bar.

B. It manages the transition of data from the user interface to NSUserDefaults.

C. It manages the transition of data inside Core Data.

D. It manages the transition between two different screens in the user interface.

9. Consider the following code snippet:

```
UIBarButtonItem *saveButton = [[UIBarButtonItem alloc]
                                       initWithBarButtonSystemItem:UIBarButtonSystemItemSave
                                                            target:self
                                                            action:@selector(saveContact:)];
```

What is the meaning of the last line?

A. A method called saveContact: has to be created in the file, and this method will be called whenever the user interacts with the control.

B. A method called @selector has to be created in the file, and this method will be called whenever the user interacts with the control.

C. This will call the saveContact: method in the UIBarButtonItem class whenever the user interacts with the control.

D. This will call the saveContact: method on the selector object.

10. Which of the following lines of code will properly store the value "Jim" to the firstName attribute in a managed object referenced by the variable name contact?

A. [contact setValue:jim forKey:firstName];

B. [contact setKey:@"Jim" forValue:@"firstName"];

C. contact.firstName = @"Jim";

D. [contact setValue:@"Jim" forKey:@"firstName"];

11. The synchronize: method is used to do what?

A. Synchronize the state of the app with iCloud.

B. Save any changes made to data in Core Data.

C. Save any changes made to data in NSUserDefaults.

D. Make sure that two screens in the app are updated to the same state.

12. With NSUserDefaults, data is stored using a Key-Value scheme. What does that mean?

A. It is used to store password values in the key chain.

B. Two items are stored: a key and a value. The key is used to identify the value.

C. Data is stored in an object graph where the key is used as an index to identify the value.

D. Two items are stored: a key and a value. The value is used to identify the key.

13. What is the advantage of declaring global constants in a separate file?

A. Avoids the programmer mistyping constant values in different parts of the program.

B. Reduces the programming effort required to create the program.

C. Improves performance of the app.

D. None. Global variables, constant or otherwise, should not be used in properly written code.

14. Why is it necessary to store the work the user is working on in an iOS app when the app becomes inactive?

A. To free up memory for the system to use

B. To increase performance of the device overall

C. Because the system can terminate the app in the background if the resources are needed

D. Because the protocol used to transfer the user interface data is stateless

15. When should a Core Data database be closed?

A. Never. The OS determines if it should be closed.

B. In the viewWillDisappear: method of the View Controller that uses the database.

C. In the viewDidUnload: method of the View Controller that uses the database.

D. As soon as the code is done using the database.

16. When data is made to exist through many iterations of an app's life cycle, it is said to ______?

A. Persist

B. Be stored

C. Archive

D. Continue

Protocol	Keeps track of which objects are stored on disk.
Managed Object Model	Describes the layout or design of data in Core Data.
Delegate	A specification of methods that must be implemented in a class.
Core Data	A group of fields to be stored in Core Data.
SQLite	A framework for storing large amounts of data in an app.
Entity	A transition between two screens in an iOS app.
User Defaults	A cross-platform database management system that can be added to mobile apps.
Attribute	An XML file for storing small amounts of data.
Managed Object Context	A field to be stored in Core Data.
Segue	An object that implements a protocol to make certain well-known features available to another object.

WK.13 / CH.15,16

❖ App Monetization Strategies

1) PAID APPS

- The price is advertised in the app store and the user decides, If the user buys the app, you get the money (70% to you , 30% for the store).

2) AD SUPPORTED APPS

- Make money from a free app by embeding ads within the app screens.
- Ads take up screen real-estate, so you will have to plan and code the user interface with this in mind.
- Google add is called *AddMob*, Apple has *iAds*, you can use both of them in Apple devices.
- Each click generates only a few cents, so you need a lot of clicks to make any real income.
- The money it generates is measured in dollars per month.
- Unlike Google’s ad service, iAd pays both for clicks on the ad and per impression ,the rate per impression is very small.

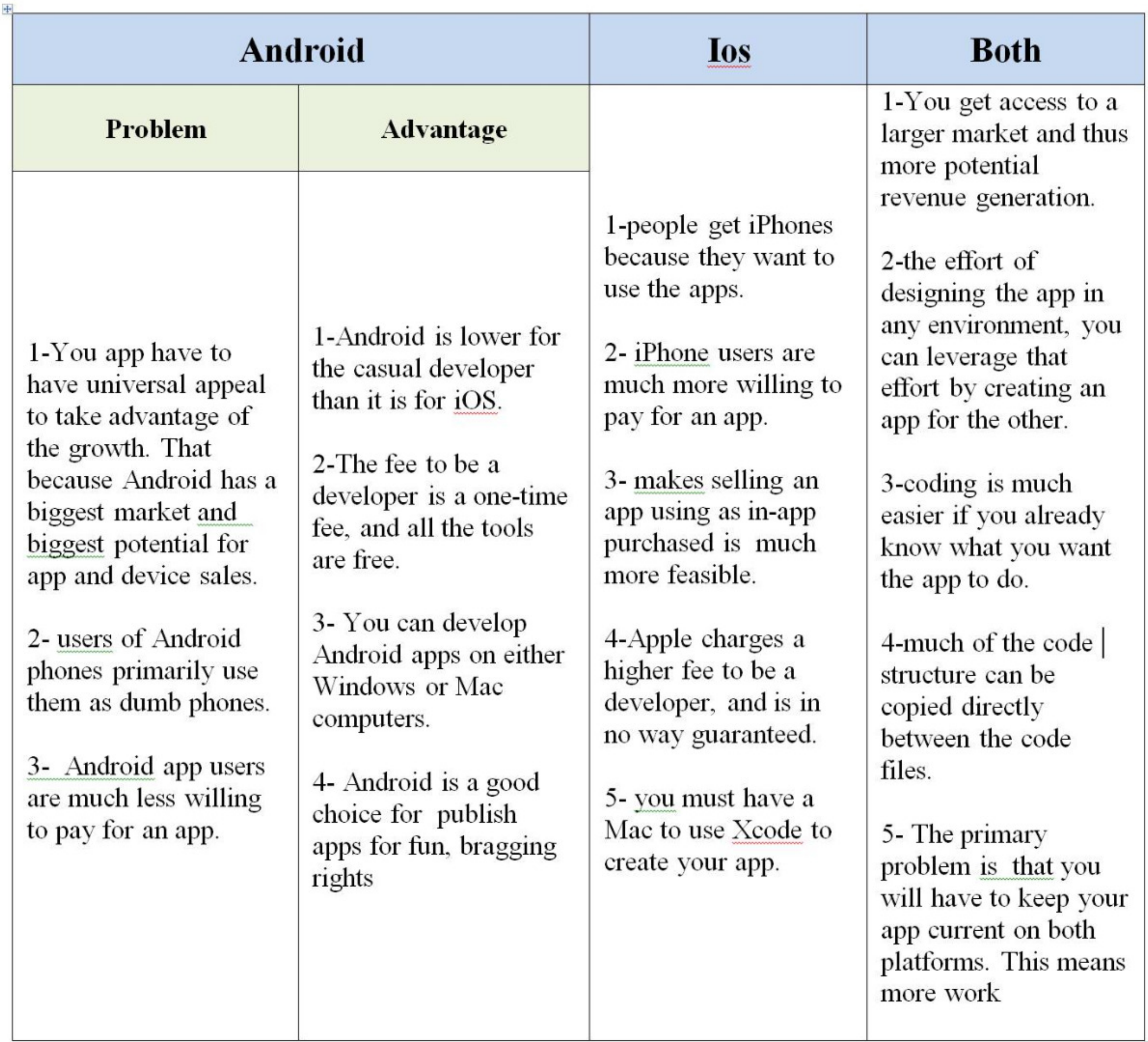
3) IN-APP PURCHASES

- You make free app, get the users hooked, and then allow them to add features by advertising the feature in the app. The sale is made during use of the app.
- More than 75% of the revenue going to iPhone developers in February 2013 came from in-app purchases.
- in-app purchasing opens up the possibility of a regular revenue stream from the same user, instead of relying on a single purchase up front.
- Often an in-app purchase strategy is combined with an ad-supported strategy. The free version includes ads that are eliminated as a bonus for an in-app purchase.

❖ The Economics of App Stores

The app stores take a 30% cut of revenue that apps generate.

❖ What Platform Should You Develop?



Android		Ios	Both
Problem	Advantage		
1-You app have to have universal appeal to take advantage of the growth. That because Android has a biggest market and biggest potential for app and device sales. 2- users of Android phones primarily use them as dumb phones. 3- Android app users are much less willing to pay for an app.	1-Android is lower for the casual developer than it is for iOS. 2-The fee to be a developer is a one-time fee, and all the tools are free. 3- You can develop Android apps on either Windows or Mac computers. 4- Android is a good choice for publish apps for fun, bragging rights	1-people get iPhones because they want to use the apps. 2- iPhone users are much more willing to pay for an app. 3- makes selling an app using as in-app purchased is much more feasible. 4-Apple charges a higher fee to be a developer, and is in no way guaranteed. 5- you must have a Mac to use Xcode to create your app.	1-You get access to a larger market and thus more potential revenue generation. 2-the effort of designing the app in any environment, you can leverage that effort by creating an app for the other. 3-coding is much easier if you already know what you want the app to do. 4-much of the code structure can be copied directly between the code files. 5- The primary problem is that you will have to keep your app current on both platforms. This means more work

❖ App Distribution through the App/Play Stores

- Developers must adhere to the store requirements
- Apple will review the app before it is published. If your app does not conform, it will be rejected.
- Google will publish the application that does not meet its store requirements, but will remove the app from the store later if it finds that the app violates its rules.

- **The steps of publishing the app**

1. **Prepare an icon for the app.**
    - Android requires an app icon sized to 512 x 512.
    - iOS requires the icon to be 1024 x 1024 pixel.

2. **Prepare screen shots of the app.**
    - Android required size of the screen shot is 320 x 480.
    - iOS regular size phone (640 x 960) and the 4-inch display phone (640 x 1136).

3. **Determine price.**
    - Android, you enter the price you want to charge for the app.
    - iOS you will be prompted to select from set of pricing.

4. **Establish category. for example, game, sports, tool, and so on.**

5. **Determine countries for app.**

6. **Licensing Library and copyrights.**

7. **Compile app and sign into application.**

8. **Debug code.**

9. **Set a private key.**

- **App distribution within an organization**

Android:

- Prepare an APK and give it to your users.
- Send users an email with the APK attached.
- Set up an internal website to distribute the app.
- Organizations implement Mobile Device Management (MDM) solutions to manage their mobile devices and app distribution.

iOS Enterprise Distribution:

- Get an iOS Enterprise Developer license.
- The cost of the license is $299 per year but Allows unlimited distribution of apps within the organization.
- You cannot sell apps in the App Store with this license.
- Organizations that do internal and public development need an Enterprise Developer license and an iOS Developer license.
- Set up an enterprise distribution certificate and an enterprise distribution provisioning profile.

Chapter 15 Monetizing Apps

1. Google takes 30% of the app sales price for apps in the Play Store. : True
2. After buying an app, the user will get free updates as long as the app stays in the app store. : True
3. Developers can make money from apps without selling the apps. : True
4. Ads on Android devices pay for both clicks and impressions. : False
5. Certain features in an app can be unlocked after it's purchased. : True
6. Google Play Billing Library is needed for ad-supported apps on Android. : False
7. To avoid having Apple or Google take a cut of a developer's revenue, users can buy subscriptions on the developer's website. : True
8. Apps developed for one mobile platform can be recompiled to work on other platforms. : False
9. All the necessary development tools to create iOS apps are free. : True
10. Freelancing is a viable approach to making money developing apps. : True

1. If an app developer raises the price of an app in the app store, what is likely to happen to the number of purchases of the app?

A. It would go down.
B. It would go up.
C. It would stay roughly the same.
D. Impossible to say without knowing more about the app.

2. Which type of app is likely to be able to command a price premium?

A. High-graphic games
B. Contact List apps
C. Apps that take advantage of most of the sensors on the device
D. Apps that solve business problems

3. Which of the following is NOT required to launch an ad-supported app?

A. Submitting bank information to Apple or Google
B. Design ads
C. Submitting a W-9 tax form to Apple or Google
D. Adding functionality to the app to serve up ads

4. Which framework allows for serving up ads on Android devices?

A. AdMob
B. iAd
C. gAd
D. AdSense

5. Which framework allows for serving up ads on iOS devices?

A. AdWorld
B. iAd
C. gAd
D. AdSense

6. What percentage do Apple and Google take on app purchases?

A. 5%
B. 10%
C. 30%
D. 50%

7. What percentage do Apple and Google take on in-app purchases?
A. 5%
B. 10%
C. 30%
D. 50%

8. Which of the following generates the most revenue for developers?
A. Paid apps
B. In-app purchasing
C. Ad-supported apps
D. Subscription

9. Which framework is needed to support in-app purchases on Android?
A. Google StoreKit
B. Google Purchasing API
C. Google Play Billing Library
D. AdMob

10. Which framework is needed to support in-app purchases on iOS?
A. Google Play Billing Library
B. iAd
C. InAppKit

D. StoreKit

11. Which of the following is a significant reason to set up a company for selling apps?
A. Clean separation of personal and business assets.
B. It's required by Apple and Google.
C. It's required by the Internal Revenue Service.
D. Potential for making more money.

12. Which is the surest way to make money for creating apps?
A. Paid apps
B. In-app purchasing
C. Freelancing
D. Contract development

13. Which mobile platform has the largest market share in terms of devices sold?
A. iOS
B. Android
C. Blackberry
D. Windows Phone

14. Which mobile platform has the largest potential for generating revenue for developers?
A. iOS
B. Android
C. Blackberry
D. Windows Phone

15. Which of these do NOT involve extra work when developing an app for multiple platforms compared to a single platform?
A. Programming the business logic
B. Keeping apps updated in multiple marketplaces
C. Designing user experience and interface
D. Learning multiple development environments

Android	A. Making money from apps.
Subscription	B. A small percentage of users finance free service for everyone else.
LLC	C. Mobile platform with the largest number of devices.
Monetization	D. Users pay for an app before starting to use it.
iOS	E. Business model where customers pay the same amount on a regular basis.
Paid apps	F. Type of business that shields your personal assets.
Platform	G. Mobile platform with potential for highest per-user revenue.
Ad supported	H. Free apps financed by third-party companies.
Freemium	I. Users can buy upgrades to an app.
In-app purchase	J. A description of a group of similar type of devices.

Chapter 16 Publishing Apps

1. Both Apple and Google require you to pay for each app you publish in the Play and App stores. False

2. Apple has strict requirements for apps published in the App Store, but Google does not have any for the Play Store. : False

3. Both Apple and Google require you to provide at least one screen shot for every app you publish in the Play and App stores. : True

4. An Android app requires extra programming by developers if they want it copy protected. An iOS app does not. : True

5. Neither Google nor Apple require developers to sign their apps prior to publication. : False

6. Apple reviews all apps prior to publishing them in the App Store. Google does not review apps prior to publication in the Play Store. : True

7. Publishing iOS apps in an organization requires a different license than the one needed to publish an app in the App Store. : True

8. Publishing Android apps in the Play Store is free for any developer, but publishing in an organization requires an Enterprise License. : False

9. Testing an Android on a real device is required prior to publication. However, the Xcode simulator is enough testing for an iOS app. : False

10. Published apps are automatically updated for the latest iOS by Apple. However, Android apps must be updated by the developer. : False

1. What is the name of the iOS app market?

A. iTunes

B. Play Store

C. App Store

D. Apple Market (iMarket)

2. What is the smallest number of screenshots required to publish an Android app?

A. 1

B. 2

C. 5

D. 10

3. How do you tell the App Store how much you want to charge for an app?

A. You enter the exact price.

B. You complete a short questionnaire about the app characteristics and purpose, and a price is selected for you.

C. You select a pricing tier from the options offered by Apple.

D. You are not allowed to charge for iOS apps.

4. How are apps protected from illegal copying in Android?
A. There is no copy protection on Android.
B. The app is encrypted.
C. An encrypted key is generated and passed to the device when the app is purchased. The operating system uses this key to unlock the app.
D. The app is coded to check if the app has been legally purchased from the Play Store.

5. Which best describes how iOS apps are distributed in an organization?
A. The company purchases the iTunes Distribution System from Apple.
B. A distribution profile and provisioning profile is packaged with the app, and then the user can get it any way the organization specifies.
C. There are many third-party solutions that an organization can purchase to do this from its own servers.
D. It is emailed to the users.

6. Which best describes how Android apps are distributed in an organization?
A. From the Google developer console.
B. From a website purchased from Google.
C. However the organization wants the distribution.
D. It is emailed to the users.

7. Which of the following is the best description of fragmentation?
A. Multiple device configurations including OS versions, screen sizes, and resolutions.
B. Frequent OS releases.
C. Multiple versions of the same app available on the Play and App stores.
D. Testing Minimum and Target SDKs.

8. If an input asks the user for the day of the month, and the developer tests it by entering 31 and 32, what type of testing is the developer doing?
A. Equivalence partitioning
B. Boundary value analysis
C. Cause-effect graphing
D. Usability

9. Who performs Alpha testing?
A. The Play and App Stores
B. Potential users
C. The organization that produced the app
D. Management

10. What is the minimum number of real devices that an iOS developer should test his/her app on?
A. 0
B. 1
C. 2 (iPad and iPhone)
D. Each device type and iOS version the app targets

11. How are apps protected from illegal copying in iOS?
A. There is no copy protection on iOS.
B. The app can be acquired only through the App Store.
C. An encrypted key is generated and passed to the device when the app is purchased. The operating system uses this key to unlock the app.
D. The app is coded to check if the app has been legally purchased from the App Store.

12. When are Android apps rejected because of policy violations?
A. Prior to publication if they fail a review by Google employees
B. Prior to publication if they fail an automated review that is executed on the uploaded binary
C. Post publication if they fail a review by Google employees
D. Post publication if there are complaints and they fail a review initiated because of complaints

13. Which of the following best describes unit testing?

A. Testing to ensure that the app meets all specifications
B. Testing to ensure that individual components of an app meet all functional specifications
C. Testing to make sure every button works
D. Testing to make sure the user can use the app effectively

14. What is the minimum number of real devices that Android developers should test their apps on?

A. 1
B. 2 (phone and pad)
C. 4 (minimum and target SDK on each phone and pad)
D. Each device type and OS version the app targets

15. What are the required app icon sizes for Android and iOS?

A. 1024 px for both
B. 512 and 1024 px, respectively
C. 72 and 144 px, respectively
D. 512 px for both

Play Store	The marketplace for apps provided and managed by Google
Fragmentation	The term used to describe an environment where there are many devices with different characteristics and operating system versions
Beta testing	Testing of an app by actual potential users of the app
Keystore	A file used to sign an Android app
Publishing	The term used to describe the act of making an app available on a market.
Unknown sources	What Android apps that don't come from the Play Store are designated as coming from.
Unit testing	Apps that do not come from the Play Store
APK	Testing individual bits of app function
Archive	The term used to describe compiling an iOS application for release.
Boundary value analysis	A type of unit testing.