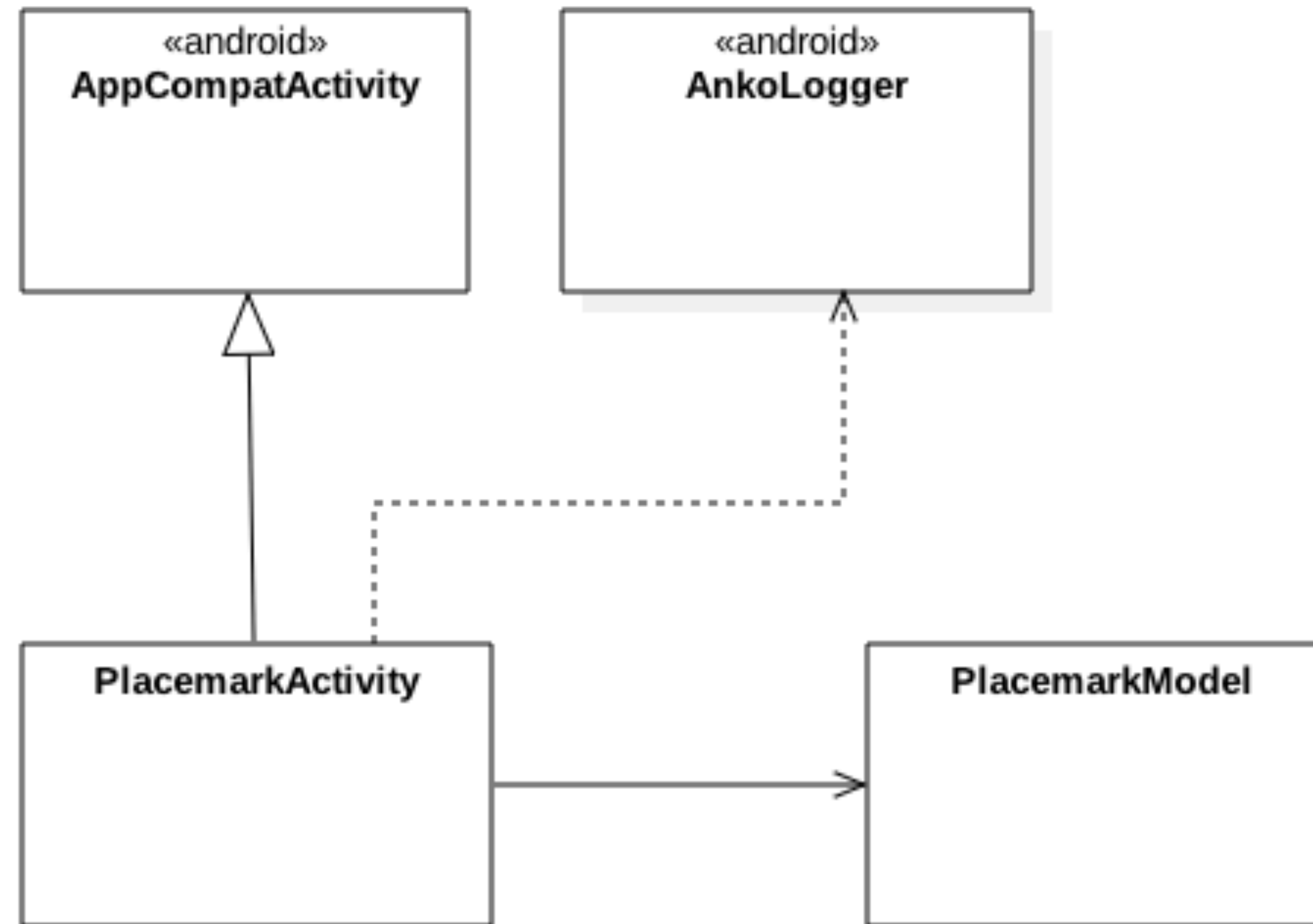# Application Objects

Android Application Object

This class enables a global application object to be defined, accessible from all activities

Current Model (Lab 01)

```kotlin
class PlacemarkActivity : AppCompatActivity(), AnkoLogger {

    var placemark = PlacemarkModel()
    val placemarks = ArrayList<PlacemarkModel>()

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_placemark)

        btnAdd.setOnClickListener() {
            placemark.title = placemarkTitle.text.toString()
            if (placemark.title.isNotEmpty()) {
                placemarks.add(placemark)
                info("add Button Pressed: $placemarkTitle")
                placemarks.forEach { info("add Button Pressed: ${it.title}")}
            }
            else {
                toast ("Please Enter a title")
            }
        }
    }
}
```
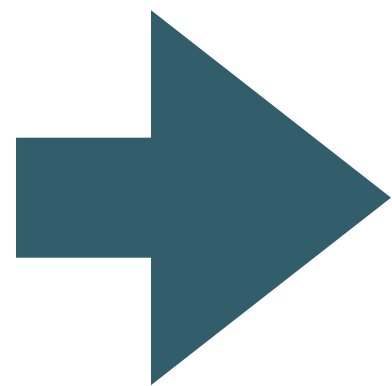
Create placemarks array

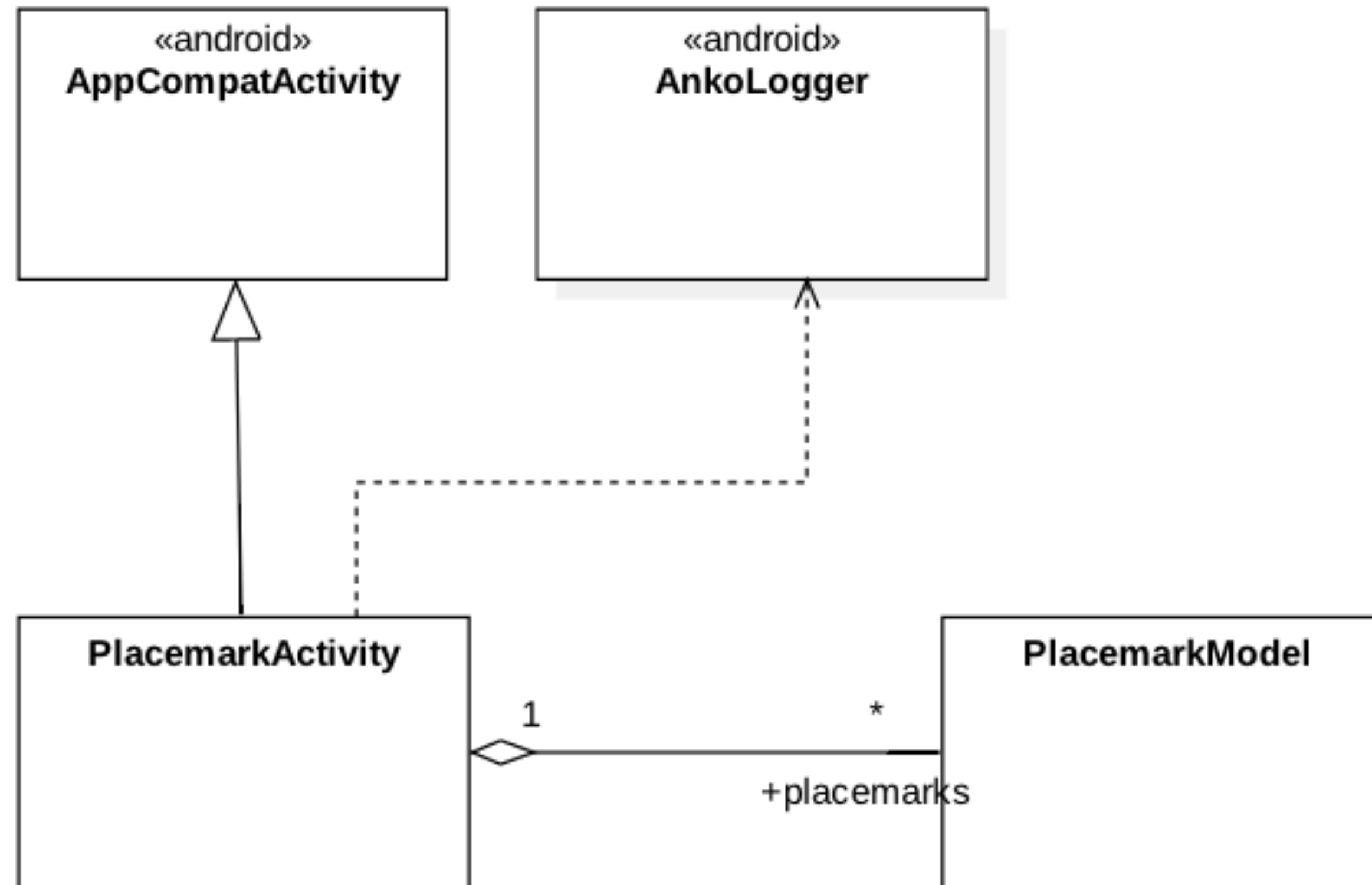..add placemark to array

```
placemarks.add(placemark)
```

```
placemarks.add(placemark.copy())
```

Make copy of placemark first, and then add the copy

Otherwise - we are adding same object, overwriting properties of previous object each time

Revised model

Exercise 2:

Create new text field **description** + log when entered. This will require you to:

- and a new field in the layout
- add a new entry in the strings.xml file
- expend the model
- recover the field in the event handler and include in the model objects
- log the new field

PlacemarkModel.kt

```kotlin
data class PlacemarkModel(var title: String = "",
                          var description: String = "")
```

PlacemarkActivity.kt

```kotlin
...
    placemark.description = description.text.toString()
...
```

strings.xml

```xml
...
  <string name="hint_placemarkDescription">Description </string>
...
```

activity_placemark.xml

```xml
...
        <android.support.design.widget.TextInputEditText
            android:id="@+id/description"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_margin="8dp"
            android:hint="@string/hint_placemarkDescription"
            android:inputType="text"
            android:maxLength="25"
            android:maxLines="1"
            android:padding="8dp"
            android:textColor="@color/colorPrimaryDark"
            android:textSize="14sp"/>
...
```

Introduce MainApp
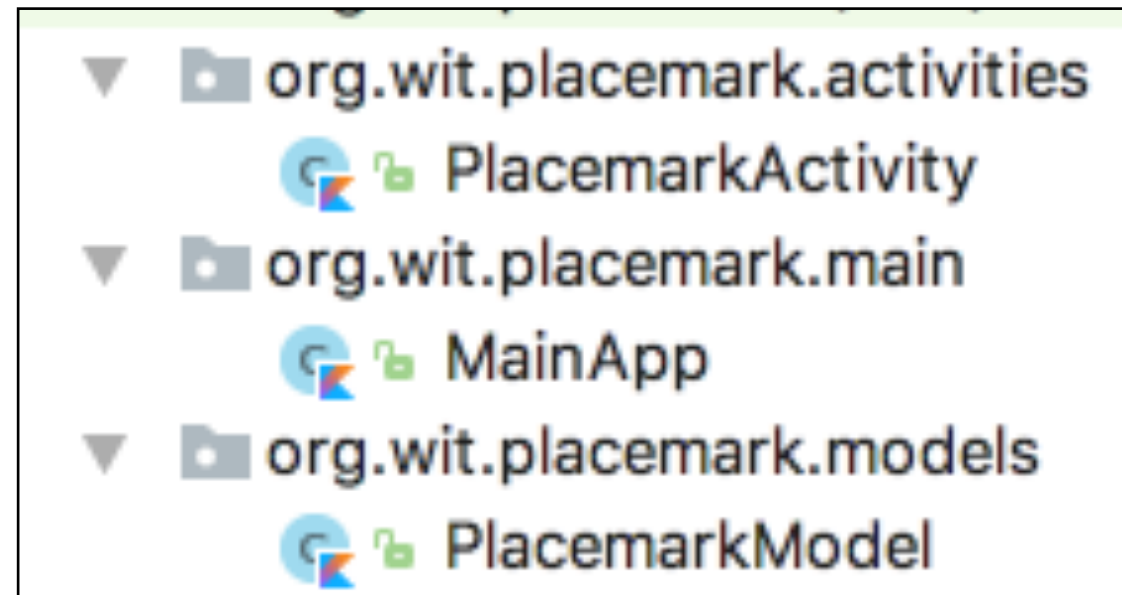

```
...
  <application
       android:name="org.wit.placemark.main.MainApp"
...
```

Include in AndroidManifest.xml

One instance of this class created for the application

```kotlin
class MainApp : Application(), AnkoLogger {

    override fun onCreate() {
        super.onCreate()
        info("Placemark started")
    }
}
```

Move placemarks Array into this class

It can be accessed from multiple Activities from here

```kotlin
class MainApp : Application(), AnkoLogger {

  val placemarks = ArrayList<PlacemarkModel>()

  override fun onCreate() {
    super.onCreate()
    info("Placemark started")
  }
}
```

Any Activity can ask for a reference to this MainApp object

Locate placemarks array through the app object

```kotlin
class PlacemarkActivity : AppCompatActivity(), AnkoLogger {

    var placemark = PlacemarkModel()
    var app : MainApp? = null

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_placemark)
        app = application as MainApp

        btnAdd.setOnClickListener() {
            placemark.title = placemarkTitle.text.toString()
            placemark.description = description.text.toString()
            if (placemark.title.isNotEmpty()) {
                app!!.placemarks.add(placemark.copy())
                info("add Button Pressed: $placemarkTitle")
                app!!.placemarks.forEach { info("add Button Pressed: ${it}")}
            }
            else {
                toast ("Please Enter a title")
            }
        }
    }
}
```

Declare the **app** object ➡️
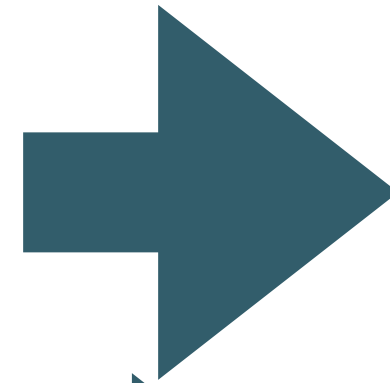
```
var app : MainApp? = null
```
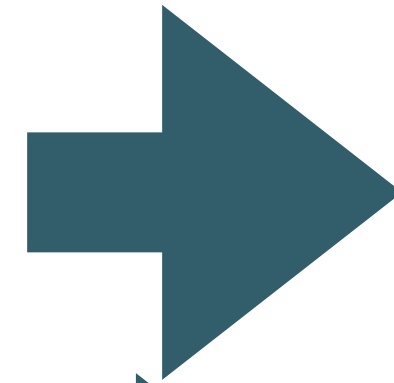
Initialise the **app** object ➡️
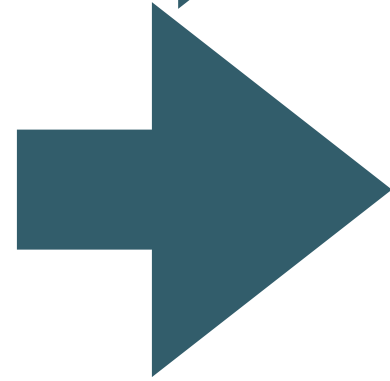
```
app = application as MainApp
```

```
app!!.placemarks.add(placemark.copy())
info("add Button Pressed: $placemarkTitle")
app!!.placemarks.forEach { info("add Button Pressed: ${it}")}
```
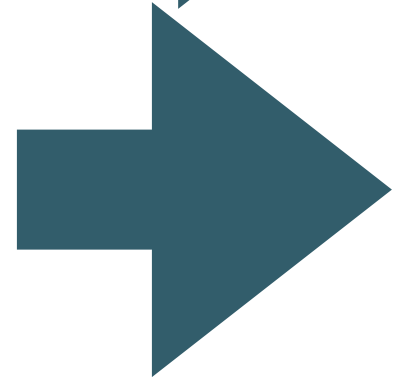
Use the **app** object to store the placemarks

Declare the **app** object as 'lateinit'

```
lateinit var app : MainApp
```

Initialise the **app** object

```
app = application as MainApp
```

```
app.placemarks.add(placemark.copy())
info("add Button Pressed: $placemarkTitle")
app.placemarks.forEach { info("add Button Pressed: ${it}")}
```

Because the app object is declared as 'lateinit' - we can assume it has been initialised, and we access without !! operators

```kotlin
class PlacemarkActivity : AppCompatActivity(), AnkoLogger {

    var placemark = PlacemarkModel()
    lateinit var app : MainApp

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_placemark)
        app = application as MainApp

        btnAdd.setOnClickListener() {
            placemark.title = placemarkTitle.text.toString()
            placemark.description = description.text.toString()
            if (placemark.title.isNotEmpty()) {
                app.placemarks.add(placemark.copy())
                info("add Button Pressed: $placemarkTitle")
                app.placemarks.forEach { info("add Button Pressed: ${it}")}
            }
            else {
                toast ("Please Enter a title")
            }
        }
    }
}
```

▼ 📁 org.wit.placemark.activities
   🅺🔓 PlacemarkActivity
▼ 📁 org.wit.placemark.main
   🅺🔓 MainApp
▼ 📁 org.wit.placemark.models
   🅺🔓 PlacemarkModel