Gradle- the basics

Gradle is a build system, like make, it can be very complex. Fortunately AS handles most of the gradle tasks for you with some exceptions. For instance changing what APIs your device targets, or what version of the appcompatibility library it uses. These usually come up when importing someone elses projects and you dont have the same SDKs installed on your machine that the original author does (like when you import my projects). This is a short guide on dealing with those situations.

Open Project

Open build.gradle (for the application)
first lib or application (see first line in gradle file)

```
apply plugin: 'com.android.library' or apply plugin: 'com.android.application'
```

Check your `compileSdkVersion`
is it installed ?(red squiggles if not) If not do so (SDK Manager) or change to one you have, careful when going backwards
use latest SDK build tools defined in the SDK manager that you have

defaultconfig
which versions you support min to target and all in between
buildtypes (not really relevant in this class)
used to support different project flavors for instance a freemium verses paid
proguard is security and obfuscation
dependencies
libraries you need
(see Support library setup
https://developer.android.com/topic/libraries/support-library/setup.html (Very important, especially the maven repo)

and Support library Packages
see https://developer.android.com/topic/libraries/support-library/packages.html
)

BTW you can see your downloaded support libs – look in your sdk location
$ANDROID_SDK/extras/android/m2repository/com/android/support/appcompat-v7
choose the latest with the major build number thats equiv to your `compileSdkVersion`
BTW if you change any of these settings resync gradle files and rebuild..
Weirdly the constraint layouts are located in another place
$ANDROID_SDK/extras/m2repository/com/android/support/constraint/ (Note the lack of the android directory.)
The maven plugin will take care of downloading these.

settings.gradle – whats in this project (project(s), libraries)

Want to add library as a dependency to an application?

- Create library (see https://developer.android.com/studio/projects/android-library.html)
- Add to project if you wish (see settings.gradle above)
- Add library as a build dependency, in the `com.android.application` gradle file add a dependency in the dependencies section, like so

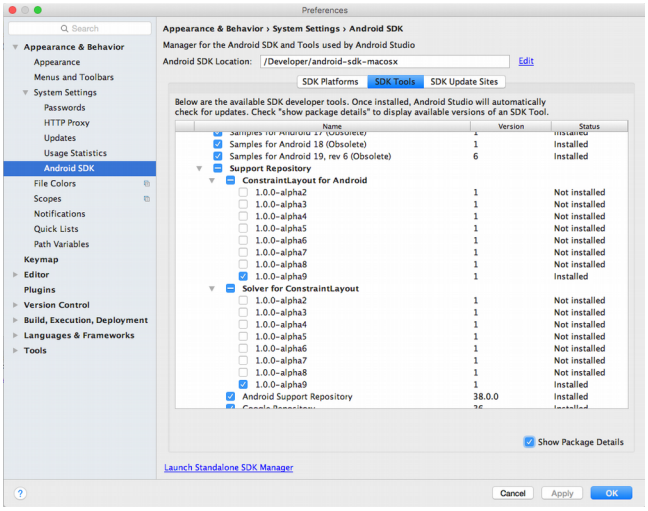
```
dependencies {
    compile fileTree(dir: 'libs', include: ['*.jar'])
    testCompile 'junit:junit:4.12'
```

```
    compile 'com.android.support:appcompat-v7:25.1.0'
    compile project(":bitmap_utilities")
}
```

ConstraintLayout

A lot of you are going to add a constraint layout to existing projects. No worries just see what version you have installed and note that version in your apps build.gradle file. (from stack overflow)

In my case, that support libraries for ConstraintLayout were installed, but I was adding the incorrect version of ConstraintLayout Library in my build.gradle file. In order to see what version have you installed, go to Preferences > Appearance & Behavior > System Settings > Android and move to SDK Tools tab. Check Show Package Details and take note of the version.



Finally you can add the correct version in the build.gradle file

```
dependencies {
        compile fileTree(dir: 'libs', include: ['*.jar'])
        compile 'com.android.support.constraint:constraint-layout:1.0.0-alpha9'
        testCompile 'junit:junit:4.12'
}
```

from Stack Overflow:

`compileSdkVersion` is the API version of Android that you compile against.

`buildToolsVersion` is the version of the compilers (aapt, dx, renderscript compiler, etc...) that you want to use. For each API level (starting with 18), there is a matching .0.0 version.
At IO 2014, we release API 20 and build-tools 20.0.0 to go with it.
Between Android releases we will release updates of the compilers, and so we'll release version .0.1, .0.2, etc...
Because we don't want to silently update these version under you, it's up to you to move to the new version when it's convenient for you. (KP use SDK Manager)
You can use a higher version of the build-tools than your compileSdkVersion, in order to pick up new/better compiler while not changing what you build your app against.