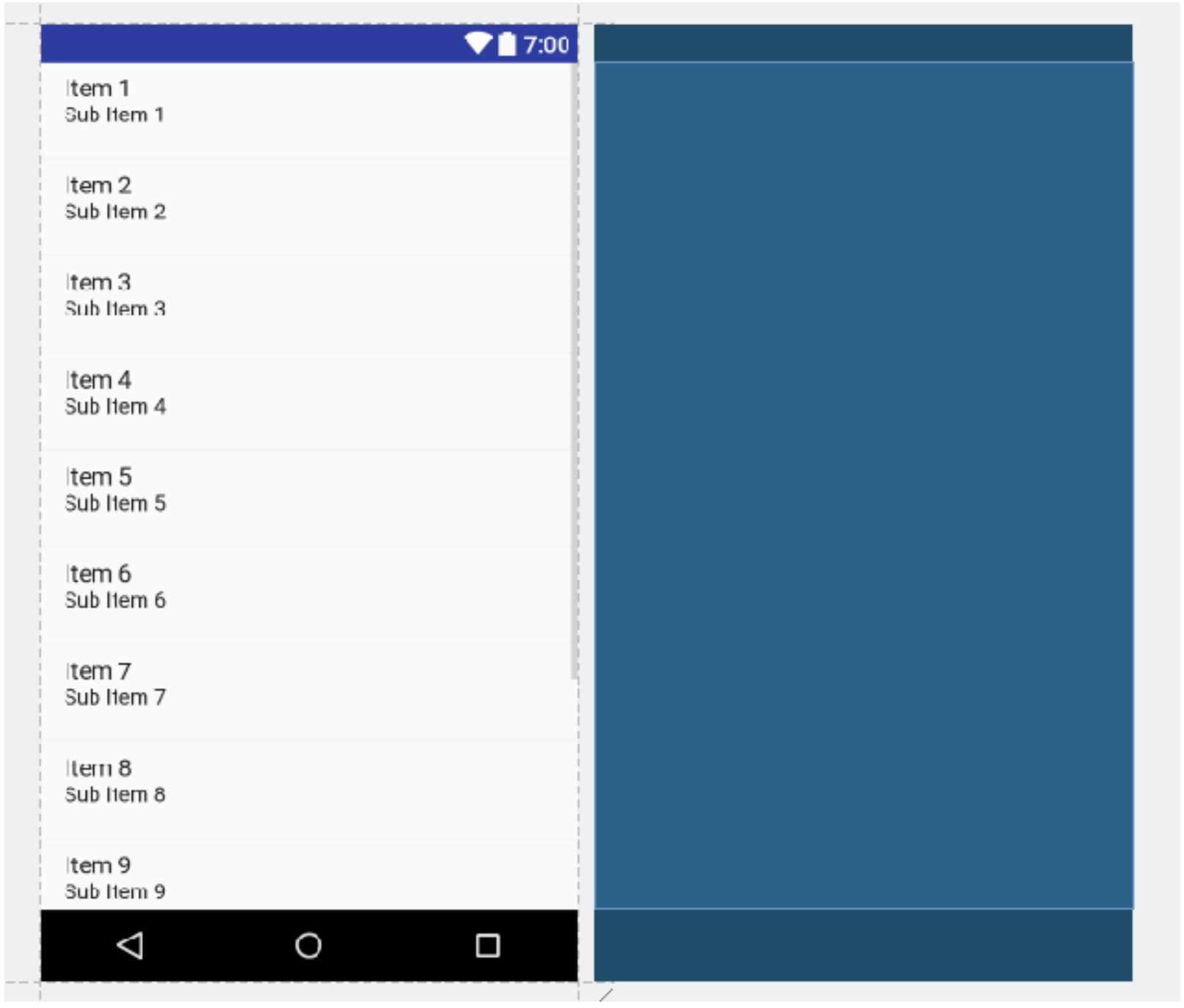
ListView와 Adapter

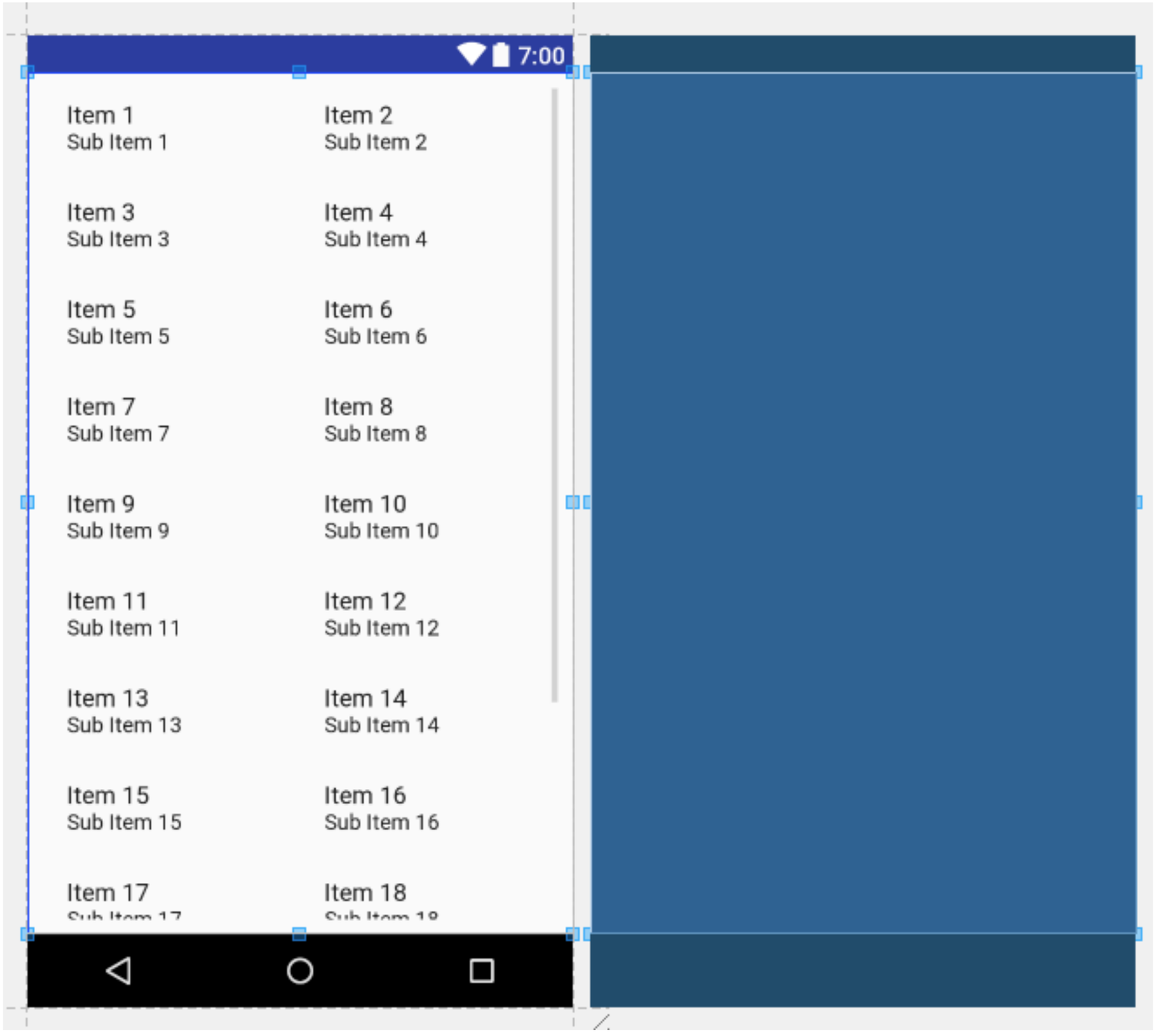
ListView


7:00
Item 1
Sub Item 1
Item 2
Sub Item 2
Item 3
Sub Item 3
Item 4
Sub Item 4
Item 5
Sub Item 5
Item 6
Sub Item 6
Item 7
Sub Item 7
Item 8
Sub Item 8
Item 9
Sub Item 9

ListAdapter

```
@Override
public int getCount() {
    return 0; //리스트에 들어갈 아이템의 개수
}

@Override
public Object getItem(int position) {
    return null; // 리스트의 각 포지션에 들어 있는 객체
}

@Override
public long getItemId(int position) {
    return 0; // 각 아이템의 인덱스 값
}

@Override
public View getView(int position, View convertView, ViewGroup parent) {

    if(row == null) {
        LayoutInflater layoutInflater =
                        ((Activity)context).getLayoutInflater();
        row = layoutInflater.inflate(R.layout.activity_list, parent, false);
    }

    return null; // 만든 커스텀뷰와 오브젝트를 연결
}
```

GridView


7:00
Item 1
Sub Item 1
Item 2
Sub Item 2
Item 3
Sub Item 3
Item 4
Sub Item 4
Item 5
Sub Item 5
Item 6
Sub Item 6
Item 7
Sub Item 7
Item 8
Sub Item 8
Item 9
Sub Item 9
Item 10
Sub Item 10
Item 11
Sub Item 11
Item 12
Sub Item 12
Item 13
Sub Item 13
Item 14
Sub Item 14
Item 15
Sub Item 15
Item 16
Sub Item 16
Item 17
Item 18

이미지 라이브러리

http://square.github.io/picasso/

```
Picasso.with(context).load("http://i.imgur.com/DvpvklR.png").into(imageView);
```

이미지 라이브러리

Build.gradle(app)

```
dependencies {
    compile fileTree(dir: 'libs', include: ['*.jar'])
    androidTestCompile('com.android.support.test.espresso:espresso-core:2.2.2', {
        exclude group: 'com.android.support', module: 'support-annotations'
    })
    compile 'com.android.support:appcompat-v7:25.3.1'
    compile 'com.android.support.constraint:constraint-layout:1.0.2'
    compile 'com.squareup.picasso:picasso:2.5.2'

    testCompile 'junit:junit:4.12'
}
```

AndroidManifest.xml

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

Local Database 사용하기

- SQLite 사용하기
- Realm 사용하기

Realm 사용하기

Build.gradle(Project : ~~~)

```
buildscript {
    repositories {
        jcenter()
    }
    dependencies {
        classpath 'com.android.tools.build:gradle:2.3.2'
        classpath "io.realm:realm-gradle-plugin:3.3.1"

        // NOTE: Do not place your application dependencies here; they belong
        // in the individual module build.gradle files
    }
}
```

Build.gradle(app)

```
apply plugin: 'com.android.application'
apply plugin: 'realm-android'
```

Realm 사용하기

```
// Obtain a Realm instance
Realm realm = Realm.getDefaultInstance();

realm.beginTransaction();

//... add or update objects here ...

realm.commitTransaction();
```

Realm 사용하기

```
// Build the query looking at all users:
RealmQuery<User> query = realm.where(User.class);

// Add query conditions:
query.equalTo("name", "John");
query.or().equalTo("name", "Peter");

// Execute the query:
RealmResults<User> result1 = query.findAll();

// Or alternatively do the same all at once (the "Fluent interface"):
RealmResults<User> result2 = realm.where(User.class)
                                  .equalTo("name", "John")
                                  .or()
                                  .equalTo("name", "Peter")
                                  .findAll();
```