Flo Maldonado

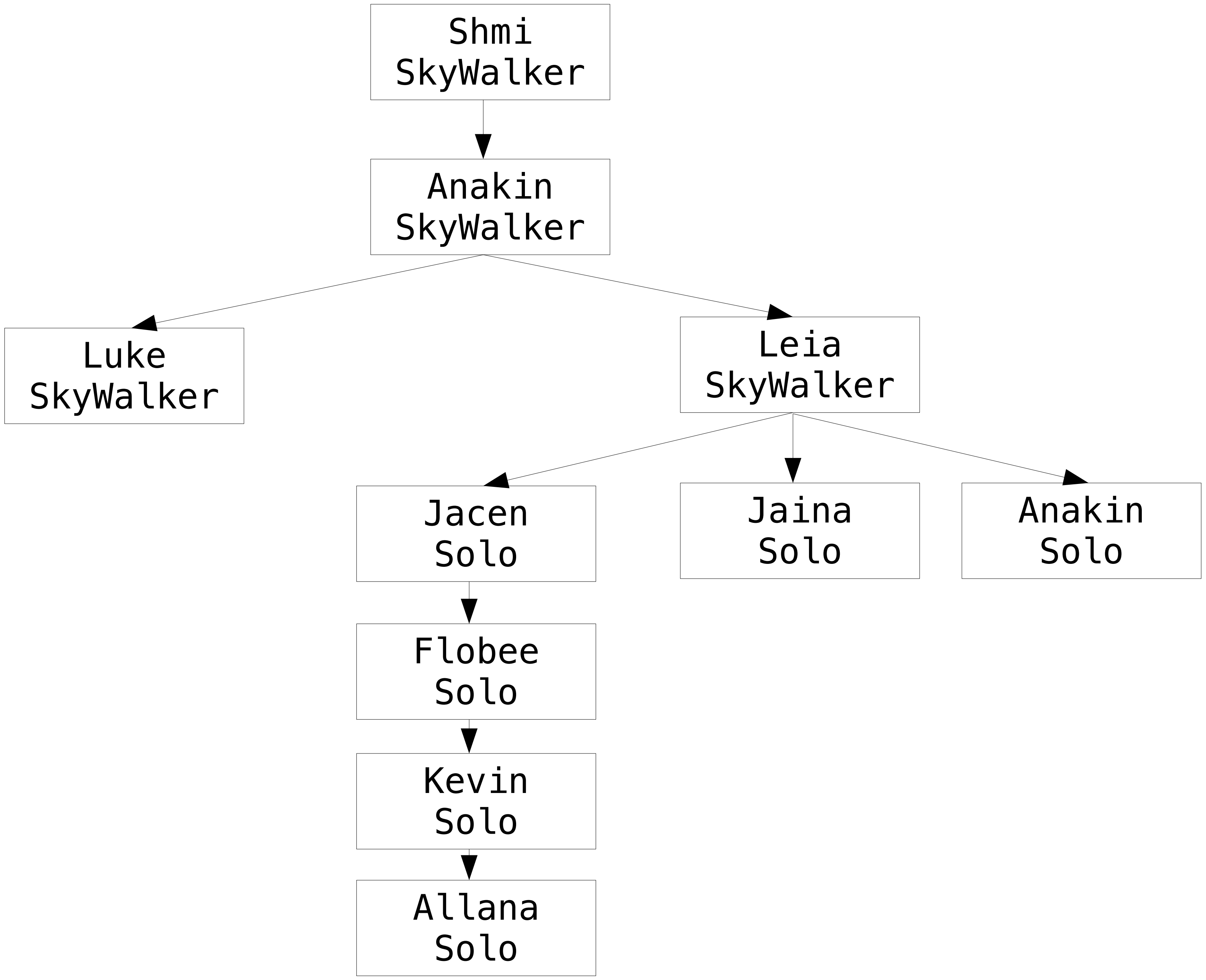
GDG-Houston

Turn off animations on your phone or emulator.

# Skywalker Lineage

Demonstration

PagerAdapters

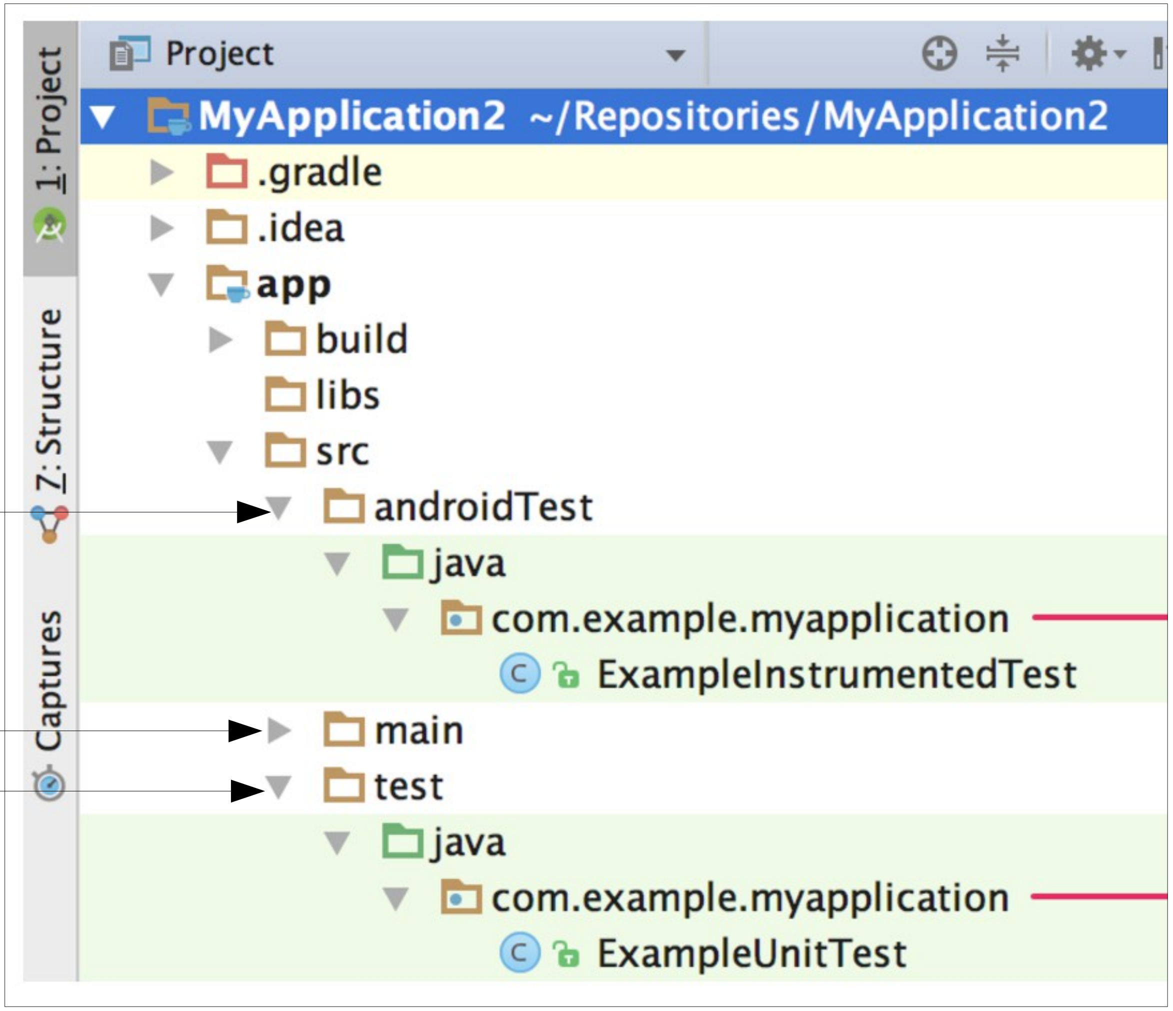


# Testing Folders

Instrumentation Tests:
Can instantiate Android classes.
Can use the emulator or phone.
Espresso simulates prescribed user interactions.

Project Code

Unit Tests:
Have to mock Android classes.
Uses JUnit assert methods.
Mockito allows for mocking.
PowerMockito allows mocking static methods and much more.



## FirstPageTest.java

```
@RunWith(AndroidJUnit4.class)
public class FirstPageTest {

  private static final String firstCharacterName = "Shmi Skywalker";

  @Rule
  public ActivityTestRule mActivityRule =
    new ActivityTestRule(MainActivity.class);

  @Test
  public void firstPageStartsWithShmi () {
    onView(allOf(isDisplayed(), withId(R.id.character_name))).
      check(matches(withText(firstCharacterName)));

    onView(allOf(isDisplayed(), withId(R.id.character_pict))).
     check(matches(DrawableMatcher.
                 withCharacterName(firstCharacterName)));
  }

}
```

## FirstPageTest.java

Espresso Grammar

```
onView(ViewMatcher).check(ViewAssertion);

onView(withId(R.id.name)).check(matches(matcher      ));

onView(withId(R.id.name))).check(matches(isDisplayed());

onView(allOf(withId(R.id.name), withText('Luke')))
    .check(matches(isDisplayed());
```

## KinkedLineWIRTest

```
@RunWith(AndroidJUnit4.class)
public class KinkedLineWIRTest {
  private ViewPagerIdlingResource idlingResource;

  @Rule
  public ActivityTestRule<MainActivity> mActivityRule =
    new ActivityTestRule<>(
      MainActivity.class, true, false);

  @After
  public void tearDownIdlingResource () {
    unregisterIdlingResources(idlingResource);
  }

  @Test
  public void toBenThenToAllana () {
    Activity activity = startActivity();

    idlingResource = new ViewPagerIdlingResource((ViewPager)activity.
      findViewById(R.id.view_pager), "VPIR_0");
    registerIdlingResources(idlingResource);

    onView(isRoot()).perform(swipeLeft());
    onView(allOf(withId(R.id.offspring_button), withText(luke_s),
                 isDisplayed())).perform(click());
    onView(allOf(withId(R.id.character_name),withText(luke_s))).
      check(matches(isCompletelyDisplayed()));
    ...
  }
```

FirstPageTest.java

Espresso Grammar

```
onView(   ViewMatcher    ).perform(ViewAction);
onView(withId(R.id.name)).perform(    click()     );
onView(allOf(withId(R.id.name), withText('Luke'))).perform(click());
```

# KinkedLineWIRTest

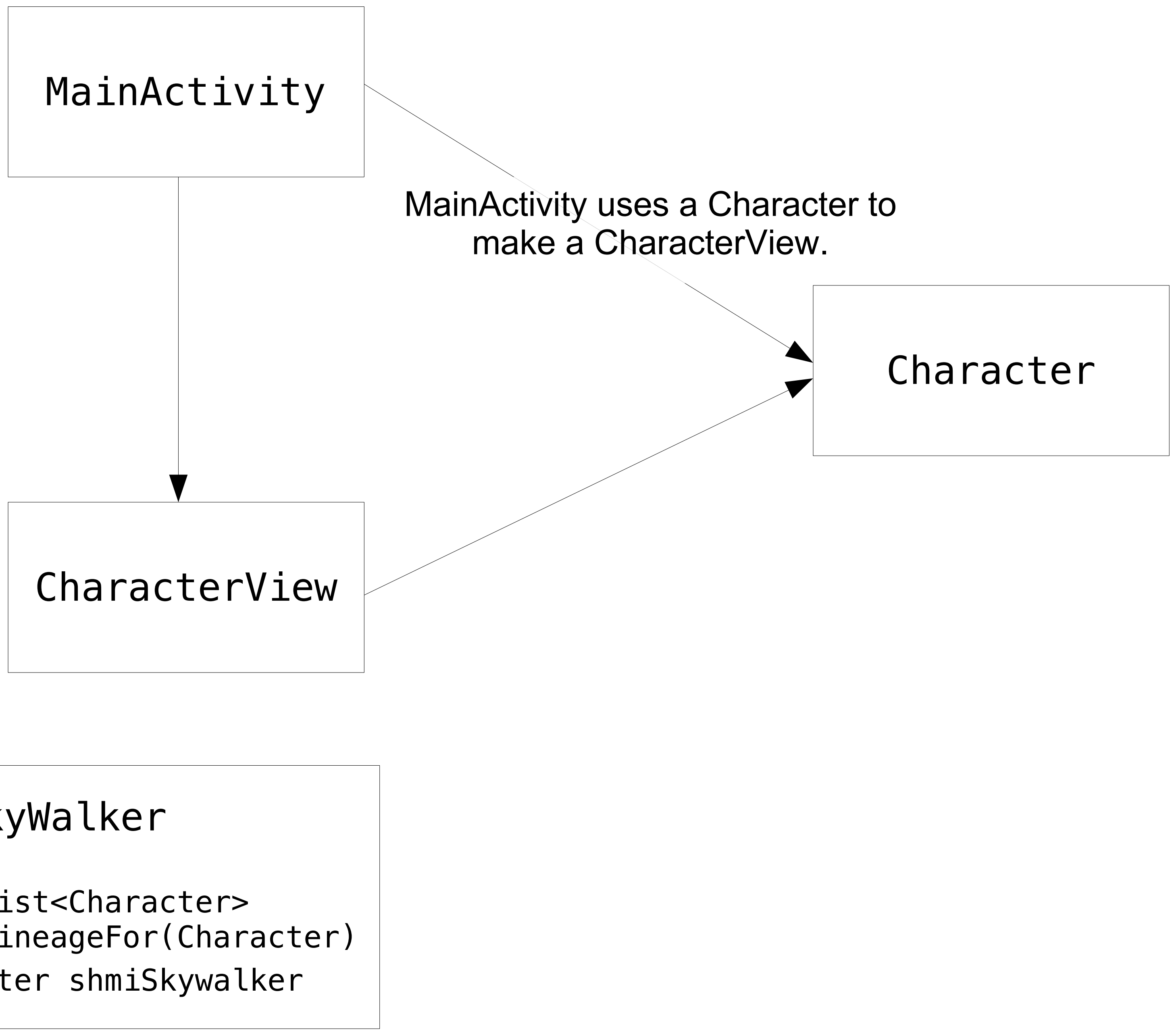
```
@RunWith(AndroidJUnit4.class)
public class KinkedLineWIRTest {
  private ViewPagerIdlingResource idlingResource;

  @Rule
  public ActivityTestRule<MainActivity> mActivityRule =
    new ActivityTestRule<>(
      MainActivity.class, true, false);

  @After
  public void tearDownIdlingResource () {
    unregisterIdlingResources(idlingResource);
  }

  @Test
  public void toBenThenToAllana () {
    Activity activity = startActivity();

    idlingResource = new ViewPagerIdlingResource((ViewPager)activity.
      findViewById(R.id.view_pager), "VPIR_0");
    registerIdlingResources(idlingResource);

    onView(isRoot()).perform(swipeLeft());
    onView(allOf(withId(R.id.offspring_button), withText(luke_s),
                 isDisplayed())).perform(click());
    onView(allOf(withId(R.id.character_name),withText(luke_s))).
      check(matches(isCompletelyDisplayed()));
    ...
  }
```

## Run Starting Tests

Run FirstPageTest in AndroidTest folder.

Run tests in test folder.
CharacterUTest1, CharacterUTest2, and CharacterViewUTest should pass!

# Starting Code - UML Diagram


MainActivity
MainActivity uses a Character to make a CharacterView.
Character
CharacterView
SkyWalker
static ArrayList<Character> getLineageFor(Character)
static Character shmiSkywalker

## CharacterUTest1.java

using Junit4 static methods

```
public class CharacterUTest1 {

@BeforeClass
public static void initCharacters () {
  shmiS       = new Character("Shmi Skywalker");
  shmiSame    = new Character("Shmi Skywalker");
  shmiDiff    = new Character("Shmi Diff");
}

@Test
public void getName () {
  String name = "JacenSolo";
  Character jacenSolo = new Character(name);
  assertEquals(name, jacenSolo.getName());
}

@Test
public void equalsPositive () {
  assertTrue(shmiS.equals(shmiSame));
}

}
```

## Character.java

Starting Code

```java
public class Character implements Parcelable {
  private String name;
  private ArrayList<Character> children = new ArrayList<>();

  public Character(String name) {
    this.name = name;
  }

  public String getName() {
    return name;
  }

  public void setChildren(Character...childs) {
    children = new ArrayList<>();
    children.addAll(Arrays.asList(childs));
  }

  public ArrayList<Character> getChildren () {
    return children;
  }
```

## res/layout/character.xml

```
<LinearLayout>

  <ScrollView>

    <LinearLayout
      android:orientation="vertical">

      <TextView
        android:id="@+id/character_name"/>

      <ImageView
        android:id="@+id/character_pict"/>

      <LinearLayout
        android:orientation="vertical"
        android:id="@+id/children_buttons">
      </LinearLayout>

    </LinearLayout>

  </ScrollView>

</LinearLayout>
```

## CharacterView.java

```java
public class CharacterView extends LinearLayout {
  private String characterName;

  public CharacterView(Context context, AttributeSet attrs) {
    super(context, attrs);
    LayoutInflater inflater = (LayoutInflater)context.
      getSystemService(Context.LAYOUT_INFLATER_SERVICE);
    inflater.inflate(R.layout.character, this, true);
  }
  public void setCharacter (Character character) {
    characterName = character.getName();
    setName(character.getName());
    setDrawable(NamedDrawable.getDrawableFromName(this.getContext(),
                                             character.getName()));
    setChildren(character.getChildren(), character.getName());
  }

  private void setName (String characterName) {
    TextView nameView = (TextView)this.findViewById(R.id.character_name);
    nameView.setText(characterName);
  }

  private void setDrawable (Drawable drawable) {
    ImageView imageView = (ImageView)this.findViewById(R.id.character_pict);
    imageView.setImageDrawable(drawable);
  }
}
```

# CharacterUTest2.java

using Mockito

```
//MockitoJUnitRunner.class allows me to mock objects.
@RunWith(MockitoJUnitRunner.class)
public class CharacterUTest2 {

  String           mockParentName;
  @Mock Character mockFirstBorn;
  @Mock Character mockSecondBorn;
  @Mock Character mockBadSon;

  @Test
  public void setChildren () {
    Character parent = new Character(mockParentName);
    parent.setChildren(mockFirstBorn, mockSecondBorn);
    ArrayList<Character> twins = parent.getChildren();

    assertEquals(2, twins.size());
    assertTrue(twins.contains(mockFirstBorn));
    assertTrue(twins.contains(mockSecondBorn));
    assertFalse(twins.contains(mockBadSon));
  }

}
```

## MainActivity.java
## activity_main.xml

```java
public class MainActivity extends AppCompatActivity {
  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);


    CharacterView characterView =
     (CharacterView)findViewById(R.id.character_view);
    characterView.setCharacter(SkyWalker.shmiSkywalker);

  }
}
```

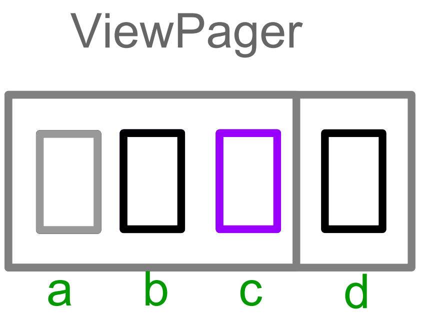
activity_main.xml

```xml
<FrameLayout
  android:id="@+id/activity_main"
  android:layout_width="match_parent"
  android:layout_height="match_parent">

  <flobee.myapplication.CharacterView
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:id="@+id/character_view">
  </flobee.myapplication.CharacterView>

</FrameLayout>
```

## FirstPageTest.java

```
@RunWith(AndroidJUnit4.class)
public class FirstPageTest {

  private static final String firstCharacterName = "Shmi Skywalker";

  @Rule
  public ActivityTestRule mActivityRule =
    new ActivityTestRule(MainActivity.class);

  @Test
  public void firstPageStartsWithShmi () {
    onView(allOf(isDisplayed(), withId(R.id.character_name))).
      check(matches(withText(firstCharacterName)));
  }

}
```

## PagerAdapter and ViewPager



Methods ViewPager calls on PagerAdapter (there are more):

```
Object instantiateItem(ViewGroup container, int pos) {
   View view = create new View.
   container.addChild(view);
   return view's unique identifier
}

 void destroyItem(ViewGroup container, int position, Object object){
   Find view inside container.
   container.remove(view);
}
```

## MyPlainPagerAdapter Intro

### MainActivity.java and activity_main.xml

```java
public class MainActivity extends AppCompatActivity {
  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    ViewPager viewPager = (ViewPager)findViewById(R.id.view_pager);
    CharacterAdapter characterAdapter = new SkywalkerAdapter();
    characterAdapter.addCharacters(
        SkyWalker.getLineageFor(SkyWalker.allanaSolo);
    PagerAdapter plainAdapter = new
      MyPlainPagerAdapter(characterAdapter);
    viewPager.setAdapter(plainAdapter);
  }
}
```

```xml
<FrameLayout
  android:id="@+id/activity_main"
  android:layout_width="match_parent"
  android:layout_height="match_parent">

  <android.support.v4.view.ViewPager
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:id="@+id/view_pager">
  </android.support.v4.view.ViewPager>

</FrameLayout>
```

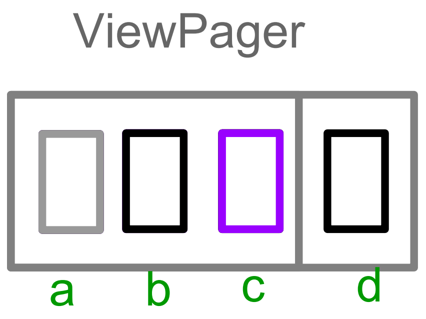
# JediAndLightsabers

using Mockito

```
@RunWith(MockitoJUnitRunner.class)
public class BlasterUTest {
  @Mock Person  leia;
  @Mock Person  anyoneElse;

  @Test
  public void verifyTurnsSaberOn ()
{
    when(leia.getName())
      .thenReturn("Leia");
    when(anyoneElse.getName())
      .thenReturn("anything");

    Blaster blaster = new Blaster();

    blaster.shoot(leia);
    verify(leia, times(0)).hit();

    blaster.shoot(anyoneElse);
    verify(anyoneElse).hit();

  }
}
```

```
class Blaster {
  void shoot (Person person) {
    if (!person.getName().equals("Leia"))
      person.hit();
  }
}

class Person {
  String name;
  boolean isHit = false;

  public Person (String name) {
    this.name = name;
  }
  void hit () {
    isHit = true;
  }

  String getName () {
    return name;
  }
}
```

# PagerAdapter and ViewPager



```
public Object instantiateItem(ViewGroup container, int position)
```

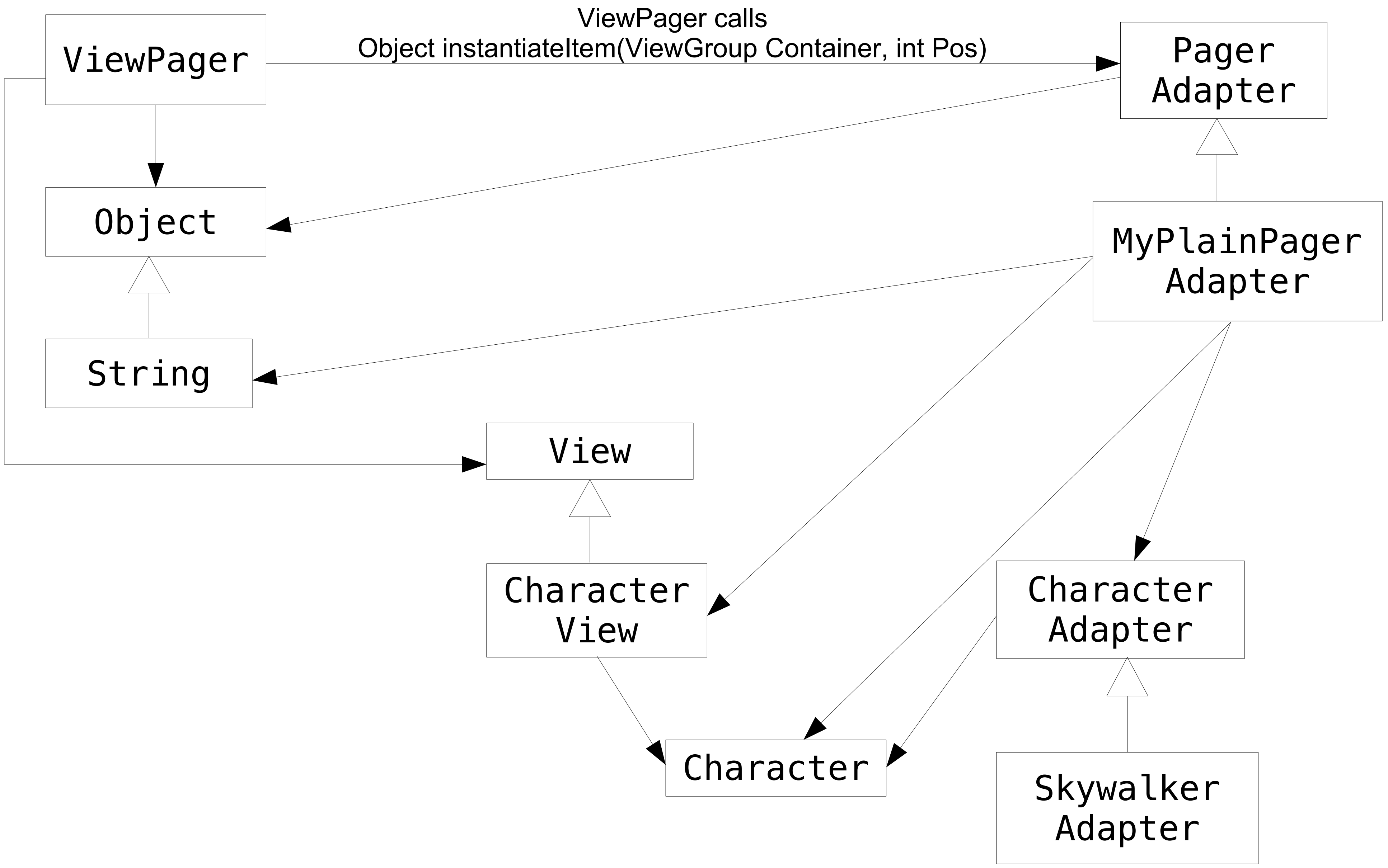
```
public void destroyItem(ViewGroup container, int position, Object object)
```

## PagerAdapter and ViewPager

Methods that ViewPager calls on PagerAdapter.

```
Object instantiateItem(ViewGroup container, int pos) {
}

void destroyItem(ViewGroup container, int position, Object object) {
}

int getItemPosition (Object object) {
  Right now just returning UNCHANGED.
}

boolean isViewFromObject (View view, Object object) {
}

int getCount () {
}
```

# MyPlainPager - UML Diagram



MyPlainPagerAdapterUTest1.java

Testing getCount()

```
@RunWith(MockitoJUnitRunner.class)
public class MyPlainPagerAdapterUTest1 {

  @Mock CharacterAdapter mockCharacterAdapter;
  int   count = 5;

  @Test
  public void returnsCountFromDataAdapter () {
    when(mockCharacterAdapter.getCount()).thenReturn(count);

    PagerAdapter pagerAdapter =
      new MyPlainPagerAdapter(mockCharacterAdapter);

    assertEquals(count, pagerAdapter.getCount());
  }
}
```

## MyPlainPagerAdapterUTest2.java

### instantiateView(ViewGroup container, int position)

```
@RunWith(PowerMockRunner.class)
public class MyPlainPagerAdapterUTest2 {
@Mock CharacterAdapter mockCharacterAdapter;
@Mock Context          mockContext;
@Mock CharacterView    mockCharacterView;
@Before
public void initCharacters () {
 when(mockContainerView.getContext())
    .thenReturn(mockContext);
 whenNew(CharacterView.class).withArguments(mockContext,
   mockAttributeSet).thenReturn(mockCharacterView);...
}

// Tests instantiateView(ViewGroup container, int mockPosition)
// Character View is made and is added to ViewGroup container.
@Test
public void characterViewAttributesAddedToCharacterView () {
  PagerAdapter pagerAdapter = new
    MyPlainPagerAdapter(mockCharacterAdapter);

  pagerAdapter.instantiateItem(mockContainerView, mockPosition);

  verify(mockContainerView).addView(mockCharacterView);
}
```

MyPlainPagerAdapterUTest2.java

MyPlainPagerAdapter

instantiateView(ViewGroup container, int position)

PagerAdapters

in MyPlainPagerAdapterUTest2.java

```
@RunWith(PowerMockRunner.class)
public class MyPlainPagerAdapterUTest2 {
@Mock CharacterAdapter mockCharacterAdapter;
@Mock Character        mockLeia;
@Mock int              mockPosition;
@Mock String           mockAName;

@Before
public void initCharacters () {
  when(mockCharacterAdapter.getCharacterAt(mockPosition))
   .thenReturn(mockLeia);
  when(mockLeia.getName()).thenReturn(leiaName);
}

// Tests instantiateView(ViewGroup container, int mockPosition)
// Leia's name is returned.
@Test
public void returnsCharacterName () {
  PagerAdapter pagerAdapter = new
    MyPlainPagerAdapter(mockCharacterAdapter);
  Object returnedObject =
    pagerAdapter.instantiateItem(mockContainerView, mockPosition);
  assertEquals(leiaName, returnedObject);
}
```

# MyPlainPagerAdapter

MyPlainPagerAdapterUTest2.java
instantiateView(ViewGroup container, int position)

```java
@RunWith(MockitoJUnitRunner.class)
public class MyPlainPagerAdapterUTest {

@Before
public void initCharacters () {
  when(mockCharacterAdapter.getCharacterAt(position)).thenReturn(mockLeia);
  when(mockContainerView.getContext()).thenReturn(mockContext);
}

// Tests destroyItem(View container, int mockPosition, Object view)
// View is removed from its container
@Test
public void testRemovesViewFromContainer () {
  when(mockContainerView.getChildCount()).thenReturn(4);
  when(mockContainerView.getChildAt(0)).thenReturn(mockCharacterView);
  when(mockCharacterView.getName()).thenReturn(leiaName);

  PagerAdapter pagerAdapter = new MyPlainPagerAdapter(mockCharacterAdapter);
  pagerAdapter.destroyItem(mockContainerView, mockPosition, leiaName);

  verify(mockContainerView).removeView(mockCharacterView);
}
```

# MyPlainPagerAdapter.java

```java
@Override
public Object instantiateItem(ViewGroup container, int position) {
  Context context = container.getContext();
  CharacterView characterView = new CharacterView(context, null);
  Character character = characterAdapter.getCharacterAt(position);
  characterView.setCharacter(character);
  container.addView(characterView);
  return character.getName();
}
```

# MyPlainPagerAdapter.java

```java
//position is the original position from the adapter, not the position
//in the container.
@Override
public void destroyItem(ViewGroup container, int position, Object object) {
  for (int ii=0; ii< container.getChildCount(); ii++) {
    View view = container.getChildAt(ii);
    if (((CharacterView)view).getName().equals(object)) {
      container.removeView(view);
      return;
    }
  }
}
```

# MyPlainPagerAdapter

Extends PagerAdapter!!

```
public class MyPlainPagerAdapter extends PagerAdapter {
  CharacterAdapter characterAdapter;

  public MyPlainPagerAdapter(CharacterAdapter characterAdapter) {
    this.characterAdapter = characterAdapter;
  }

  @Override
  public int getCount() {
    return characterAdapter.getCount();
  }

  @Override
  public boolean isViewFromObject(View view, Object object) {
    CharacterView characterView = (CharacterView)view;
    return (characterView.getName().equals(object));
  }

}
```

*From PagerAdapter:*

```
public int getItemPosition(Object object) {
    return POSITION_UNCHANGED;
}
```

## StraightLineWIRTest

MyPlainPagerAdapter

PagerAdapters

```
@RunWith(AndroidJUnit4.class)
public class StraightLineWIRTest {
  private ViewPagerIdlingResource idlingResource;

  @Rule
  public IntentsTestRule<MainActivity> mActivityRule =
    new IntentsTestRule(MainActivity.class, true, false);

  @After
  public void tearDownIdlingResource () {
    unregisterIdlingResources(idlingResource);
  }

  @Test
  public void firstSwipe () {
    Activity activity = startActivity();

    idlingResource = new ViewPagerIdlingResource((ViewPager)activity.
      findViewById(R.id.view_pager), "VPIR_0");
    registerIdlingResources(idlingResource);

    onView(isRoot()).perform(swipeLeft());
    onView(allOf(withId(R.id.character_name),withText(anakin_s))).
      check(matches(isCompletelyDisplayed()));
    onView(allOf(withId(R.id.character_name),withText(shmi_s))).
      check(matches(not(isCompletelyDisplayed())));
  }
```

## MyPlainPagerAdapter

Ready to Write MyPlainPagerAdapter

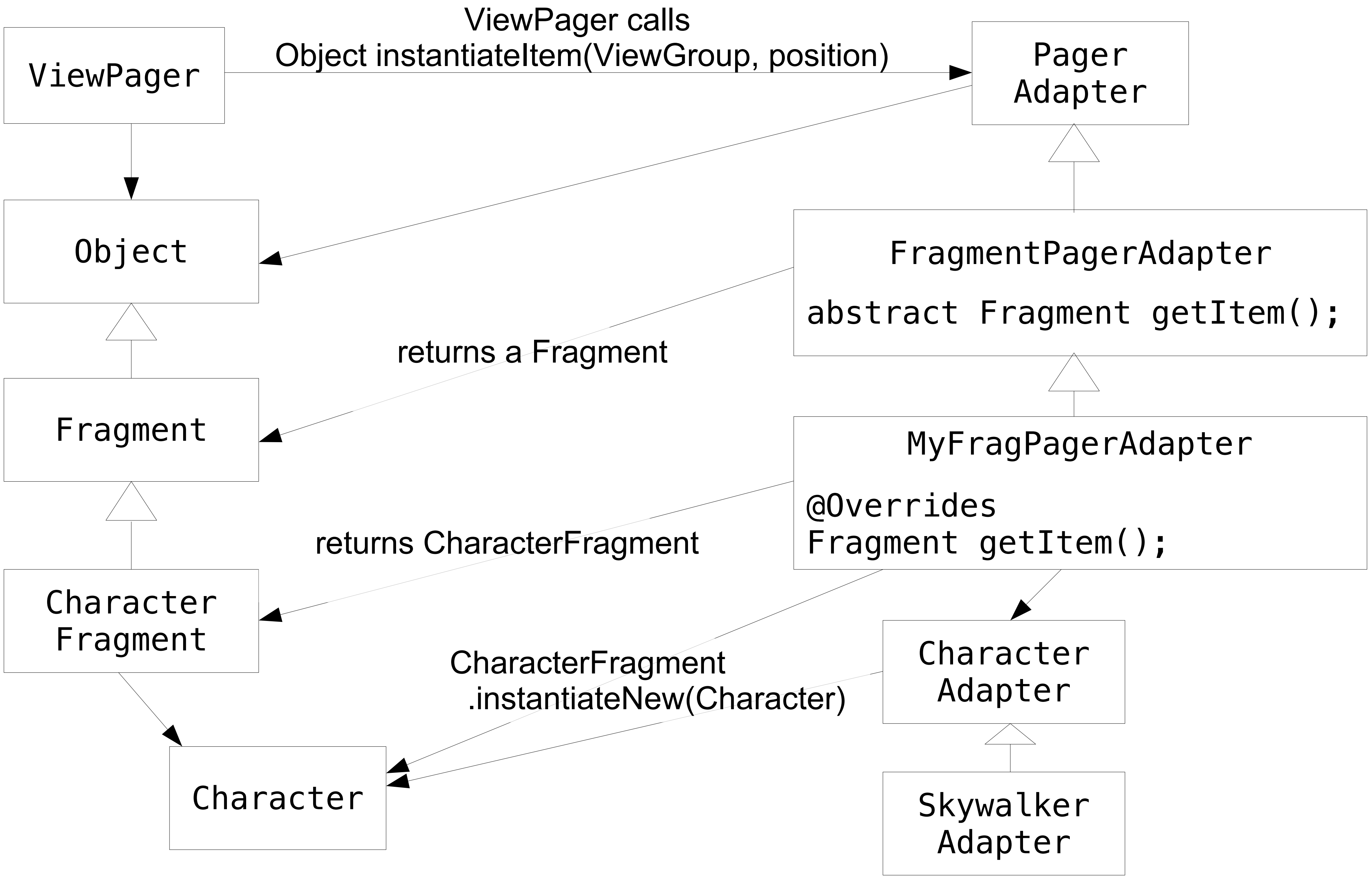
and Run Tests.

FragmentPagerAdapter

an Android class

```
@Override
public Object instantiateItem(ViewGroup container, int position) {
    if (mCurTransaction == null) {
        mCurTransaction = mFragmentManager.beginTransaction();
    }

    final long itemId = getItemId(position);

    // Do we already have this fragment?
    String name = makeFragmentName(container.getId(), itemId);
    Fragment fragment = mFragmentManager.findFragmentByTag(name);
    if (fragment != null) {
        mCurTransaction.attach(fragment);
    } else {
        fragment = getItem(position);
        mCurTransaction.add(container.getId(), fragment,
                makeFragmentName(container.getId(), itemId));
    }
    if (fragment != mCurrentPrimaryItem) {
        fragment.setMenuVisibility(false);
        fragment.setUserVisibleHint(false);
    }

    return fragment;
}
```

# MyFragPagerAdapter - UML Diagram



## FragmentPagerAdapter

an Android class

```
@Override
public Object instantiateItem(ViewGroup container, int position) {
    if (mCurTransaction == null) {
        mCurTransaction = mFragmentManager.beginTransaction();
    }

    final long itemId = getItemId(position);

    // Do we already have this fragment?
    String name = makeFragmentName(container.getId(), itemId);
    Fragment fragment = mFragmentManager.findFragmentByTag(name);
    if (fragment != null) {
        mCurTransaction.attach(fragment);
    } else {
        fragment = getItem(position);
        mCurTransaction.add(container.getId(), fragment,
                makeFragmentName(container.getId(), itemId));
    }
    if (fragment != mCurrentPrimaryItem) {
        fragment.setMenuVisibility(false);
        fragment.setUserVisibleHint(false);
    }

    return fragment;
}
```

# MyFragPagerAdapter.java

```java
public class MyFragPagerAdapter extends FragmentPagerAdapter {

  CharacterAdapter characterAdapter;

  MyFragPagerAdapter(FragmentManager  fragmentManager,
                     CharacterAdapter adapter) {
    super(fragmentManager);
    characterAdapter = adapter;
  }

  @Override
  public int getCount() {
    return characterAdapter.getCount();
  }

  @Override
  public Fragment getItem(int position) {
    return CharacterFragment
             .newInstance(characterAdapter.getCharacterAt(position));
  }
}
```

## MyFragPagerAdapter

That's it!

`Add code.`

and Run Tests.

## MyFragPagerAdapter

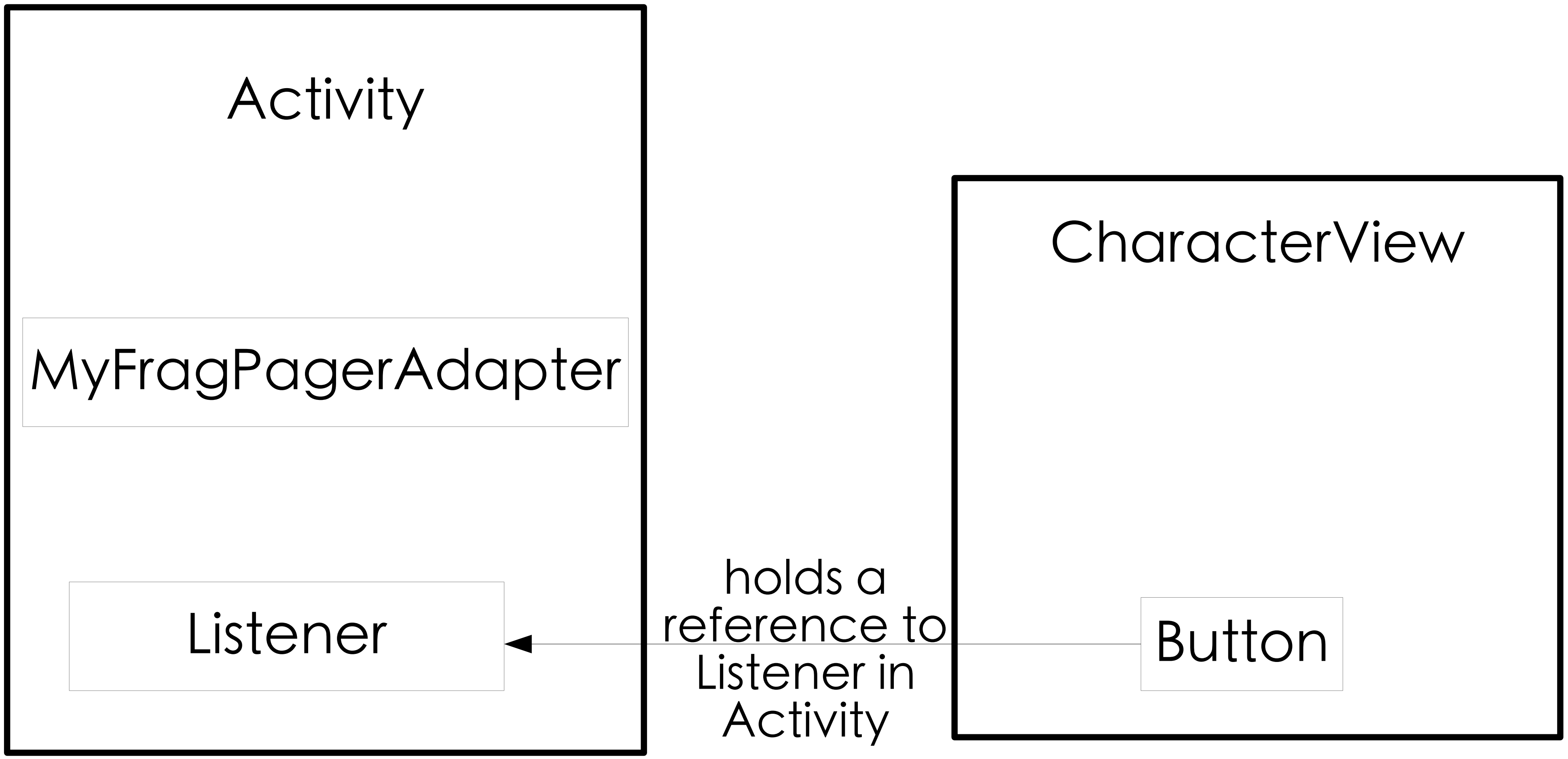
Kinked Line

MyFragPagerAdapter

```
void changeChildTo(String parent, String nextChild)
```

MyFragPagerAdapter - Kinked Line

ChildListener



## MyFragPagerAdapter Kinked Line

MainActivity calling
notifyDataSetChanged()

```
public ChildButtonListener getChildListener () {
  if (listener != null)
    return listener;
  else {
    return new ChildButtonListener() {
      @Override
      public void changeChildTo(String parent, String nextChild) {
        ViewPager viewPager
          = (ViewPager) findViewById(R.id.view_pager);
        if (viewPager!= null) {
          MyFragPagerAdapter adapter =
           (MyFragPagerAdapter)viewPager.getAdapter();
          adapter.changeChildTo(parent, nextChild);
          adapter.notifyDataSetChanged();
          viewPager.setCurrentItem(viewPager.getCurrentItem() + 1);
        }
      }
    };
  }
}
```

# MyFragPagerAdapter - Kinked Line

FragPagerAdapter methods that need to be updated after notifyDataSetChanged()

```
int getItemPosition (Object object) {
  We can't always return UNCHANGED anymore!
}

int getItemId (int position) {
  FragmentPagerAdapter uses the result of this method to make the id tags
  for its fragments. Before position was adequate to distinguish
  fragments. Now different fragments will have the same position.
}
```

# MyFragPagerAdapter Kinked Line

## SkywalkerAdapter Diagram

`Needed results for instantiateItem() and getItemPosition()`

Family Lines

| 0 | 1 | 2 |
|---|---|---|
| Shmi | Shmi | Shmi |
| * Anakin (click Luke) | * Anakin (click Leia) | * Anakin |
| **Leia** | Luke | **Leia** |
| **Jacen** | Ben | Jaina |
| Flo | | |

| 0 | 1 | 1 | 2 | 2 |
|---|---|---|---|---|
| Shmi Fragment | UN CHANGED | | UN CHANGED | |
| Anakin Fragment | UN CHANGED | | UN CHANGED | |
| Leia Fragment | NONE | Luke Fragment | NONE | Leia Fragment |
| | | | | |
| | | | | |

## MyFragPagerAdapter Kinked Line
## after adding notifyDataSetChanged()

FragmentPagerAdapter methods that need to be updated after notifyDataSetChanged()

```
int getItemPosition (Object object) {
  We can't always return UNCHANGED anymore!
}

int getItemId (int position) {
  getItemId is used to make fragment tags! Tags have to be unique!
}
```

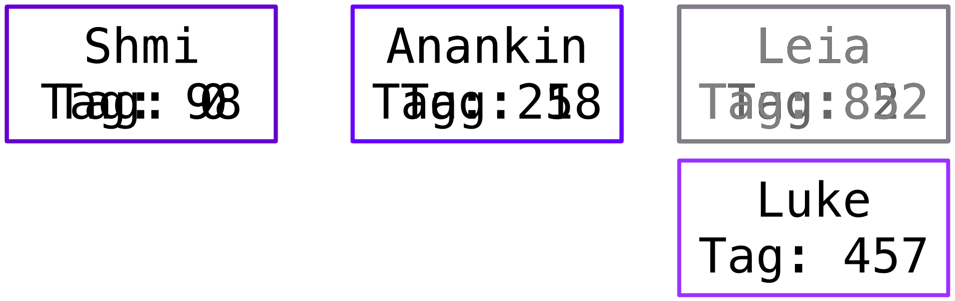
# FragPagerAdapter Kinked Line

after notifyDataSetChanged()

FragmentPagerAdapter

```
Object instantiateItem (Container, position) {

  fragmentName = getItemId(position);
  fragment     = findFragmentByTag(fragmentName);
  if (fragment == null) {
    fragment = getItem();
    currTransaction.add(container, fragment, fragmentName);
  }
  else {
    currTransaction.attach(fragment);

}

long getItemId(int position) {
```

```
@Override
long getItemId(int position) {
character = characterAdapter.getCharacterAt(position);
return character.getCharacterNameHashCode();
```



# FragmentPagerAdapter

An android class

```
@Override
public void destroyItem(ViewGroup container, int position, Object object) {
    if (mCurTransaction == null) {
        mCurTransaction = mFragmentManager.beginTransaction();
    }

    mCurTransaction.detach((Fragment)object);
}
```

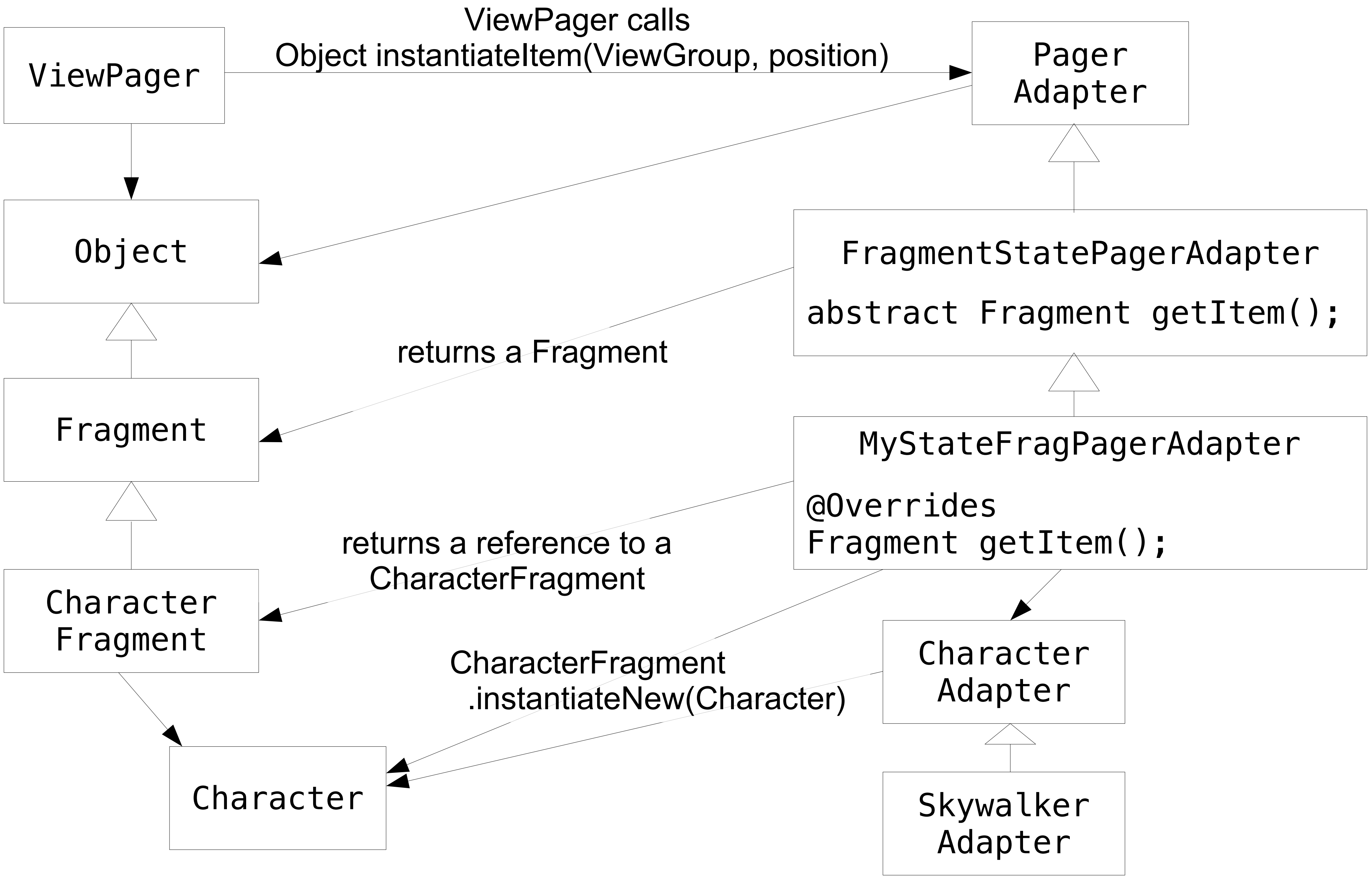
## MyFragPagerAdapter
### Kinked Line

Run Tests

and

**Run Application.**

# MyFragStatePagerAdapter - UML Diagram


ViewPager calls
Object instantiateItem(ViewGroup, position)
ViewPager
Pager Adapter
Object
FragmentStatePagerAdapter
abstract Fragment getItem();
returns a Fragment
Fragment
MyStateFragPagerAdapter
@Overrides
Fragment getItem();
returns a reference to a CharacterFragment
Character Fragment
CharacterFragment .instantiateNew(Character)
Character Adapter
Character
Skywalker Adapter

## MyFragStatePagerAdapter

### SetUp

```
@Override
protected void onCreate(Bundle savedInstanceState) {
  super.onCreate(savedInstanceState);
  setContentView(R.layout.activity_main);

  ViewPager viewPager = (ViewPager)findViewById(R.id.view_pager);
  CharacterAdapter characterAdapter = new SkywalkerAdapter();
  characterAdapter.addCharacters(
    SkyWalker.getLineageFor(SkyWalker.allanaSolo));
  characterAdapter.addCharacters(
    SkyWalker.getLineageFor(SkyWalker.benSkywalker));
  characterAdapter.addCharacters(
    SkyWalker.getLineageFor(SkyWalker.jainaSolo));
  characterAdapter.addCharacters(
    SkyWalker.getLineageFor(SkyWalker.anakinSolo));
  PagerAdapter myFragStatePagerAdapter =
    new MyFragStatePagerAdapter(this.getSupportFragmentManager(),
                                characterAdapter);
  viewPager.setAdapter(myFragStatePagerAdapter);
  ...
}
```

## Array of Fragments, Array of FragmentStates

MyFragmentStatePagerAdapter

Run Tests

FragmentStatePagerAdapter

Not Working!!

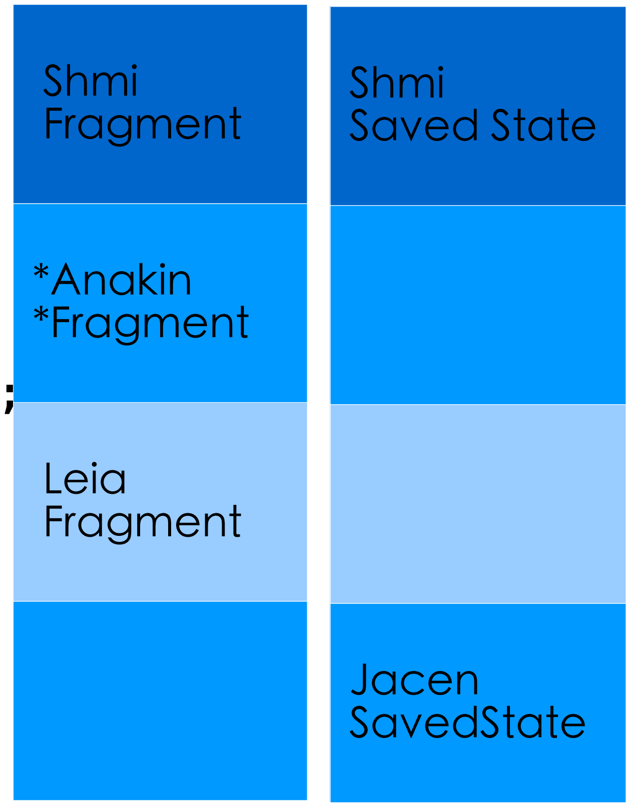
## Array of Fragments, Array of FragmentStates

```
@Override
public void destroyItem(ViewGroup container, int position, Object object) {
    Fragment fragment = (Fragment) object;

    if (mCurTransaction == null) {
        mCurTransaction = mFragmentManager.beginTransaction();
    }
    while (mSavedState.size() <= position) {
        mSavedState.add(null);
    }
    mSavedState.set(position, fragment.isAdded()
            ? mFragmentManager.saveFragmentInstanceState(fragment) : null);
    mFragments.set(position, null);

    mCurTransaction.remove(fragment);
}
```

## Array of Fragments, Array of FragmentStates

### MyFragmentStatePagerAdapter

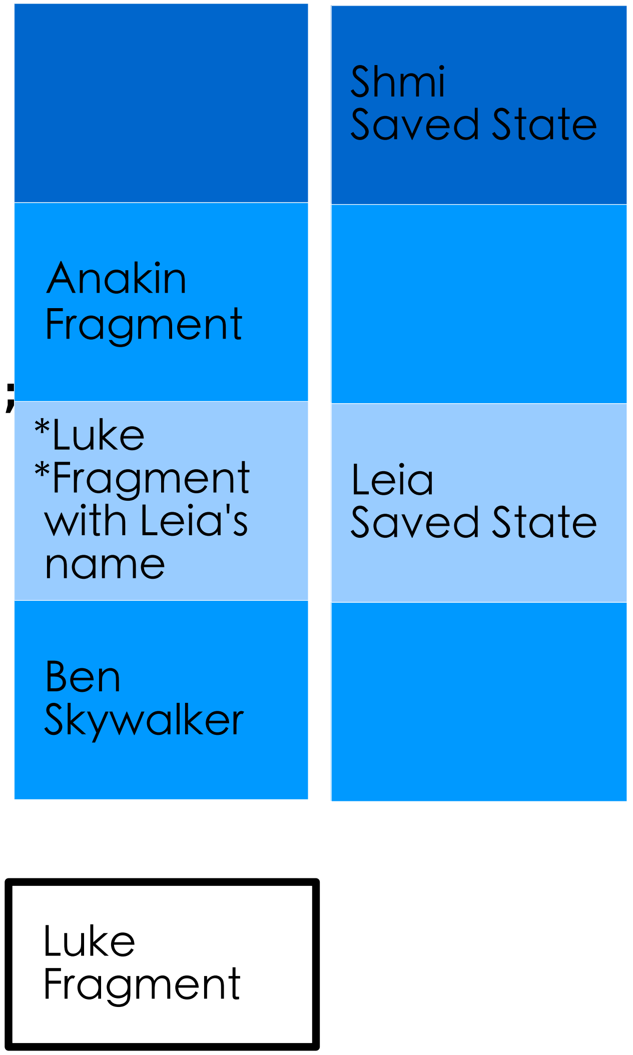
```
@Override
public Object instantiateItem(ViewGroup container, int position) {
  if (mFragments.size() > position) {
    Fragment f = mFragments.get(position);
    if (f != null) {
      return f;
    }
  }
  Fragment fragment = getItem(position);
  if (mSavedState.size() > position) {
    Fragment.SavedState fss = mSavedState.get(position);
    if (fss != null) {
      fragment.setInitialSavedState(fss);
    }
  }
  while (mFragments.size() <= position) {
    mFragments.add(null);
  }
  mFragments.set(position, fragment);
  mCurTransaction.add(container.getId(), fragment);

  return fragment;
}
```



## Array of Fragments, Array of FragmentStates

## MyFragmentStatePagerAdapter

```
@Override
public Object instantiateItem(ViewGroup container, int position) {
  if (mFragments.size() > position) {
    Fragment f = mFragments.get(position);
    if (f != null) {
      return f;
    }
  }
  Fragment fragment = getItem(position);
  if (mSavedState.size() > position) {
    Fragment.SavedState fss = mSavedState.get(position);
    if (fss != null) {
      fragment.setInitialSavedState(fss);
    }
  }
  while (mFragments.size() <= position) {
    mFragments.add(null);
  }
  mFragments.set(position, fragment);
  mCurTransaction.add(container.getId(), fragment);

  return fragment;
}
```



FragmentStateAdapter2

MyFragStatePagerAdapter

SetUp

PagerAdapters

```
@Override
public void notifyDataSetChanged() {
  Fragment currF = mCurrentPrimaryItem;
  List<Integer> toRemove = new LinkedList<>();
  if (currF != null) {
    int currPosition = (findFragment(currF));
    if (currPosition != -1) {
      for (int ii=currPosition+1; ii<mFragments.size(); ii++) {
        Fragment existingFragment = mFragments.get(ii);
        if (existingFragment == null)
          toRemove.add(ii);
        else if(getItemPosition(existingFragment) == POSITION_NONE)
          toRemove.add(ii);
      }
      for (Integer toGo : toRemove) {
        removeFragment(toGo);
      }
    }
  }
  super.notifyDataSetChanged();
}
```

## FragmentStateAdapter2

SetUp

```
private void removeFragment (int position) {
  if (mFragments.size() <= position)
    return;
  if (mCurTransaction == null) {
    mCurTransaction = mFragmentManager.beginTransaction();
  }
  Fragment wrongFragment = mFragments.get(position);
  if (wrongFragment != null) {
    mCurTransaction.remove(wrongFragment);
    mFragments.set(position, null);
  }
  if (mSavedState.size() > position) {
    mSavedState.set(position, null);
  }
}
```

FragmentStateAdapter2

SetUp

```
@Override
public void destroyItem(ViewGroup container, int position, Object object) {
  Fragment fragment = (Fragment) object;

  if (mCurTransaction == null) {
    mCurTransaction = mFragmentManager.beginTransaction();
  }
  while (mSavedState.size() <= position) {
    mSavedState.add(null);
  }
  int isInLineFragment = getItemPosition(object);
  if (isInLineFragment != PagerAdapter.POSITION_NONE) {
    mSavedState.set(position, fragment.isAdded()
      ? mFragmentManager.saveFragmentInstanceState(fragment) : null);
    mFragments.set(position, null);
  }
  else {
    mSavedState.set(position, null);
  }

  mCurTransaction.remove(fragment);
}
```

# CharacterViewUTest.java

using PowerMockito

```
@RunWith(PowerMockRunner.class)
@PrepareForTest({NamedDrawable.class, CharacterView.class})
public class CharacterViewUTest {

  @Mock Context  mockContext;
  @Mock TextView mockNameView;
  private String anakinName = SkyWalker.anakinSkywalker.getName();

  @Before
  public void init () throws Exception {
    suppress(methodsDeclaredIn(LayoutInflater.class));
  }

  @Test
  public void setViewsFromCharacter () throws Exception  {
    CharacterView characterView = new CharacterView(mockContext, null);
    CharacterView spyCharView   = spy(characterView);
    when(spyCharView.findViewById(R.id.character_name))
      .thenReturn(mockNameView);

    spyCharView.setCharacter(SkyWalker.getCharacter(anakin));

    verify(mockNameView).setText(anakinName);
  }
```