Hugbúnaðarverkefni 2 / Software Project 2

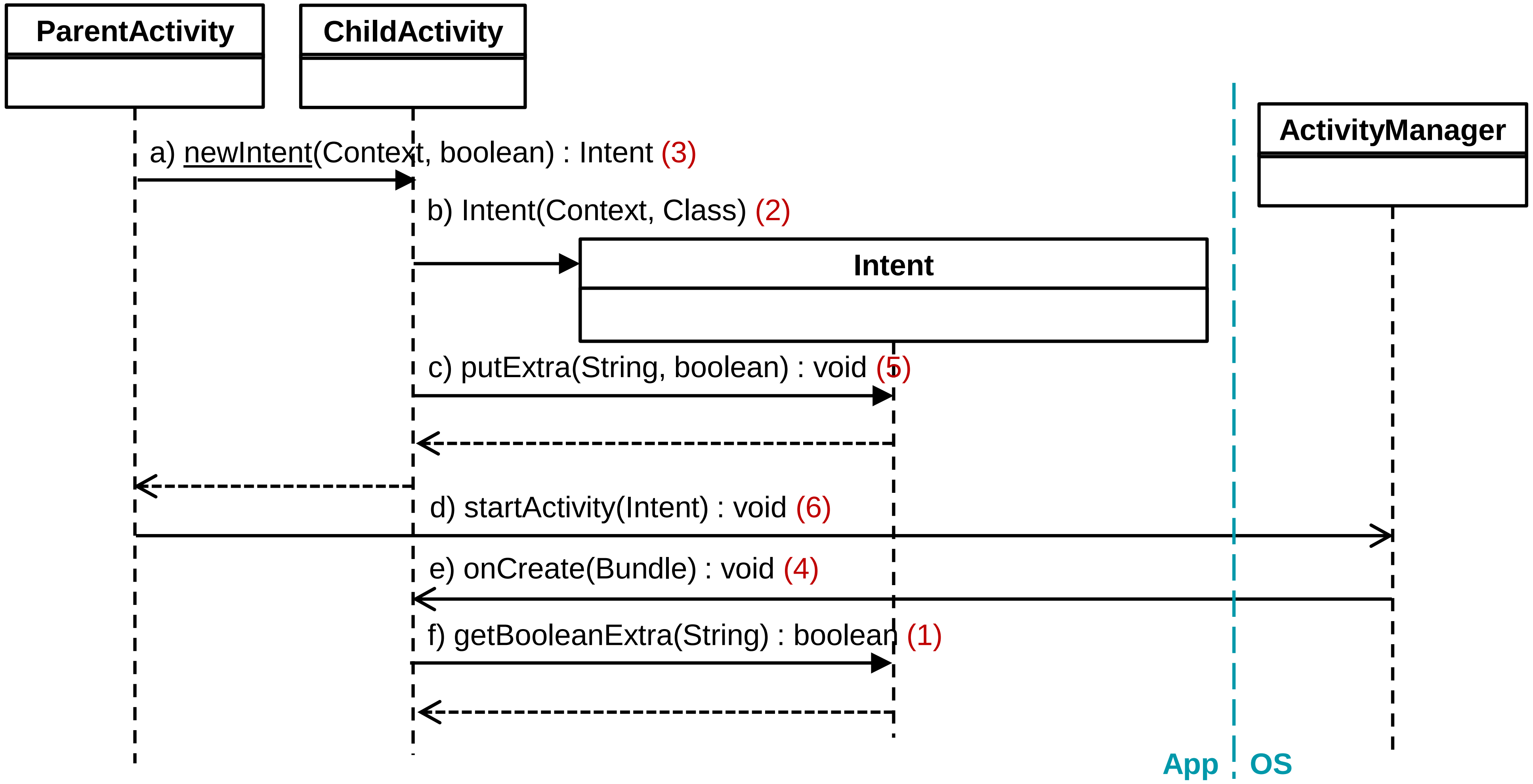
8. Android User Interfaces: Fragments

HBV601G – Spring 2019

Matthias Book

HÁSKÓLI ÍSLANDS
VERKFRÆÐI- OG NÁTTÚRUVÍSINDASVIÐ
IÐNAÐARVERKFRÆÐI-, VÉLAVERKFRÆÐI- OG TÖLVUNARFRÆÐIDEILD

In-Class Quiz 6 Solution: Communicating with Intents


ParentActivity
ChildActivity
ActivityManager
a) newIntent(Context, boolean) : Intent (3)
b) Intent(Context, Class) (2)
Intent
c) putExtra(String, boolean) : void (5)
d) startActivity(Intent) : void (6)
e) onCreate(Bundle) : void (4)
f) getBooleanExtra(String) : boolean (1)
App
OS

In-Class Quiz 7 Prep

- Please prepare a small scrap of paper with the following information:

ID: ____@hi.is Date: ________

a) ____	d) ____
b) ____	e) ____
c) ____	f) ____

- During class, I'll show you questions that you can answer with a number
- Hand in your scrap at end of class
- All questions in a quiz have same weight
- All quizzes (8-10 throughout semester) have the same weight
  - Your worst 2 quizzes will be disregarded
- Overall quiz grade counts as optional question worth 7.5% on final exam

Fragments

see also:

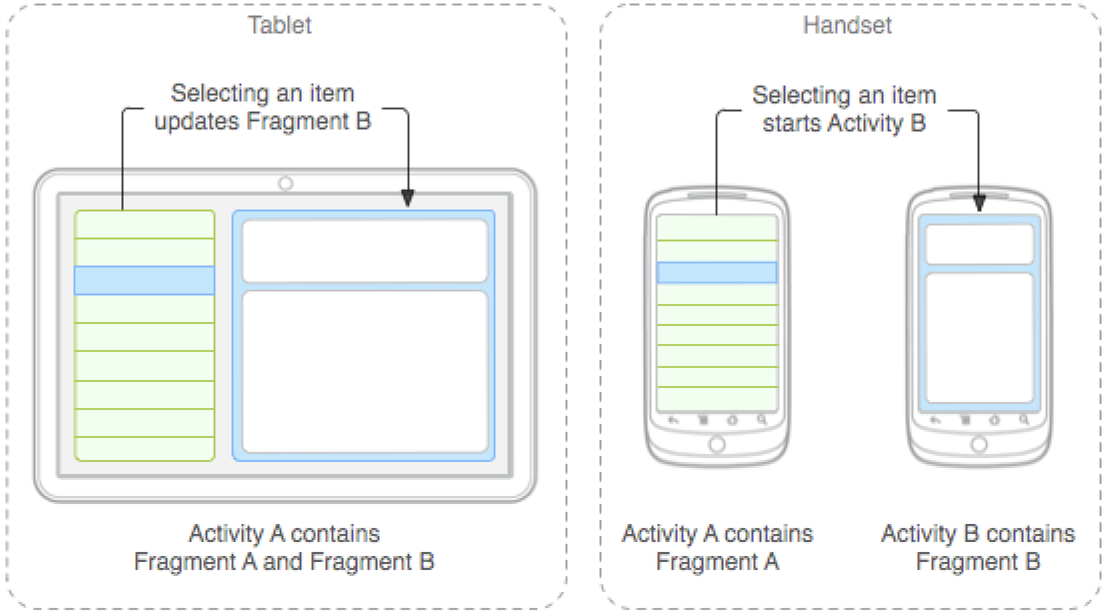
- Phillips et al.: Android Development, Ch. 7, 9 (2nd ed.) / 8 (3rd ed.), 10, 11, 17
- http://developer.android.com/guide/components/fragments.html

Recap: Activities vs. Fragments

One activity enabling paging through several fragments

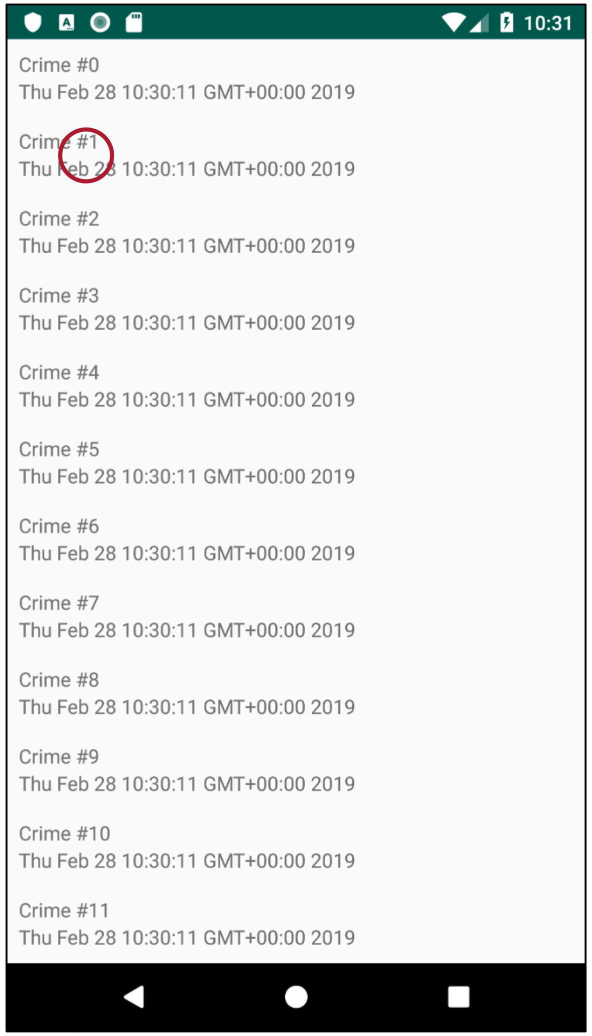
- An **activity** is the controller for one particular screen with a static layout
- A **fragment** is the controller for a quite independent user interface “module” that
  - is nested into a host activity
  - can be added to or deleted from the activity at runtime
  - has its own layout and behavior
  - receives its own input and lifecycle events
  - can be shown in combination with other fragments inside one activity
  - can be shown as a pop-up or tabbed dialog

➢Fragments enable dynamic UI changes.


Tablet
Selecting an item updates Fragment B
Activity A contains Fragment A and Fragment B
Handset
Selecting an item starts Activity B
Activity A contains Fragment A
Activity B contains Fragment B

Recap: List-Detail User Interface with Fragments

(Expected User Experience in Portrait Orientation)



10:31

Crime #0
Thu Feb 28 10:30:11 GMT+00:00 2019

Crime #1
Thu Feb 28 10:30:11 GMT+00:00 2019

Crime #2
Thu Feb 28 10:30:11 GMT+00:00 2019

Crime #3
Thu Feb 28 10:30:11 GMT+00:00 2019

Crime #4
Thu Feb 28 10:30:11 GMT+00:00 2019

Crime #5
Thu Feb 28 10:30:11 GMT+00:00 2019

Crime #6
Thu Feb 28 10:30:11 GMT+00:00 2019

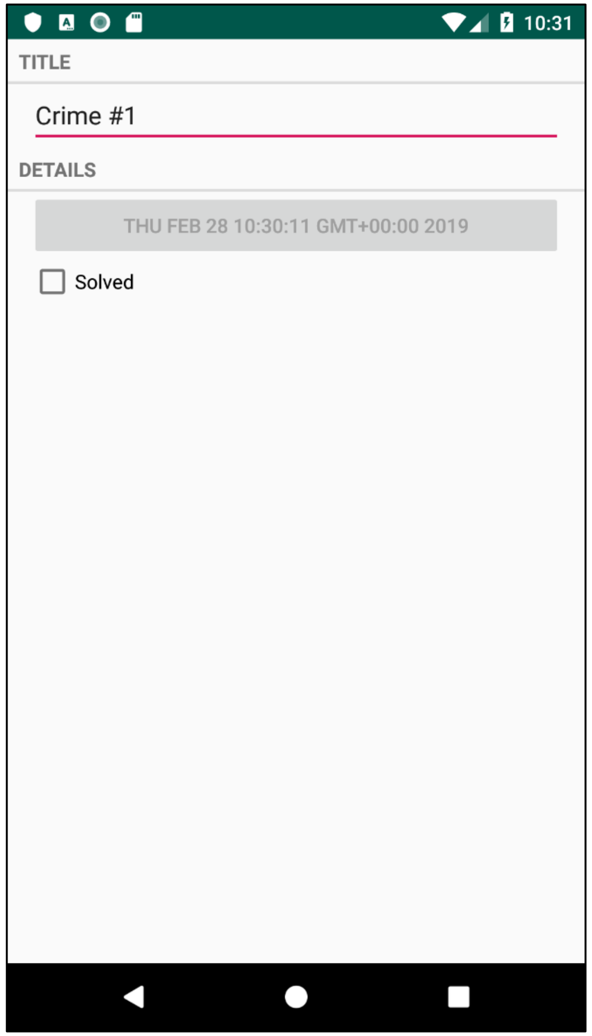
Crime #7
Thu Feb 28 10:30:11 GMT+00:00 2019

Crime #8
Thu Feb 28 10:30:11 GMT+00:00 2019

Crime #9
Thu Feb 28 10:30:11 GMT+00:00 2019

Crime #10
Thu Feb 28 10:30:11 GMT+00:00 2019

Crime #11
Thu Feb 28 10:30:11 GMT+00:00 2019



10:31

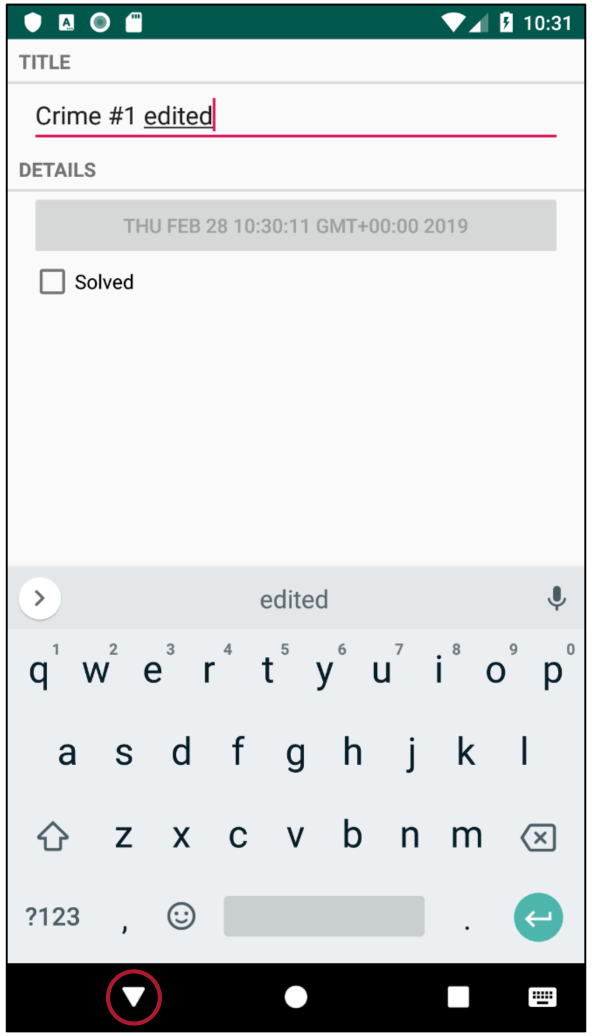
TITLE

Crime #1

DETAILS

THU FEB 28 10:30:11 GMT+00:00 2019

☐ Solved



10:31

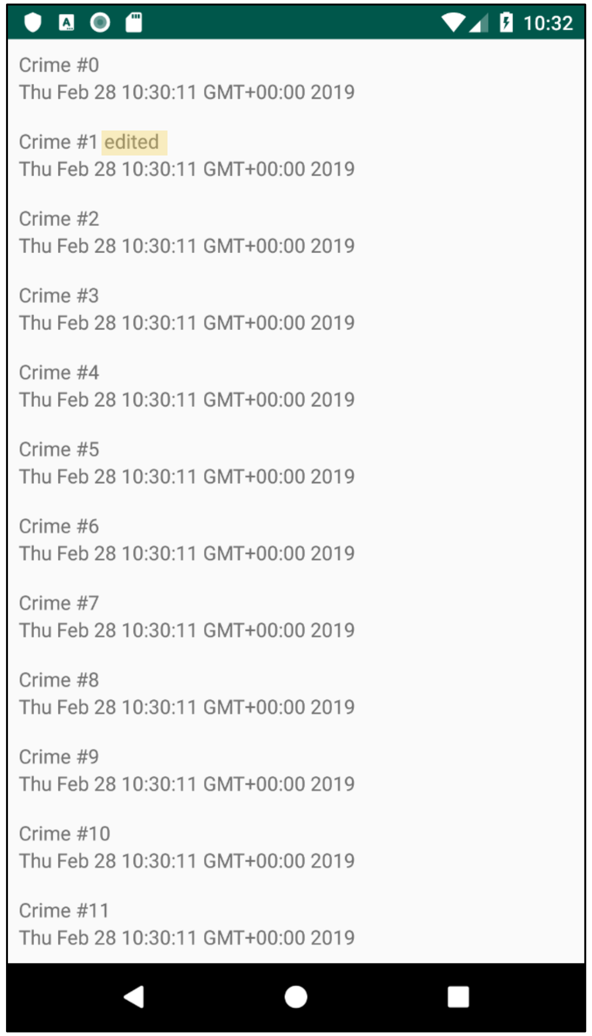
TITLE

Crime #1 edited

DETAILS

THU FEB 28 10:30:11 GMT+00:00 2019

☐ Solved



10:32

Crime #0
Thu Feb 28 10:30:11 GMT+00:00 2019

Crime #1 edited
Thu Feb 28 10:30:11 GMT+00:00 2019

Crime #2
Thu Feb 28 10:30:11 GMT+00:00 2019

Crime #3
Thu Feb 28 10:30:11 GMT+00:00 2019

Crime #4
Thu Feb 28 10:30:11 GMT+00:00 2019

Crime #5
Thu Feb 28 10:30:11 GMT+00:00 2019

Crime #6
Thu Feb 28 10:30:11 GMT+00:00 2019

Crime #7
Thu Feb 28 10:30:11 GMT+00:00 2019

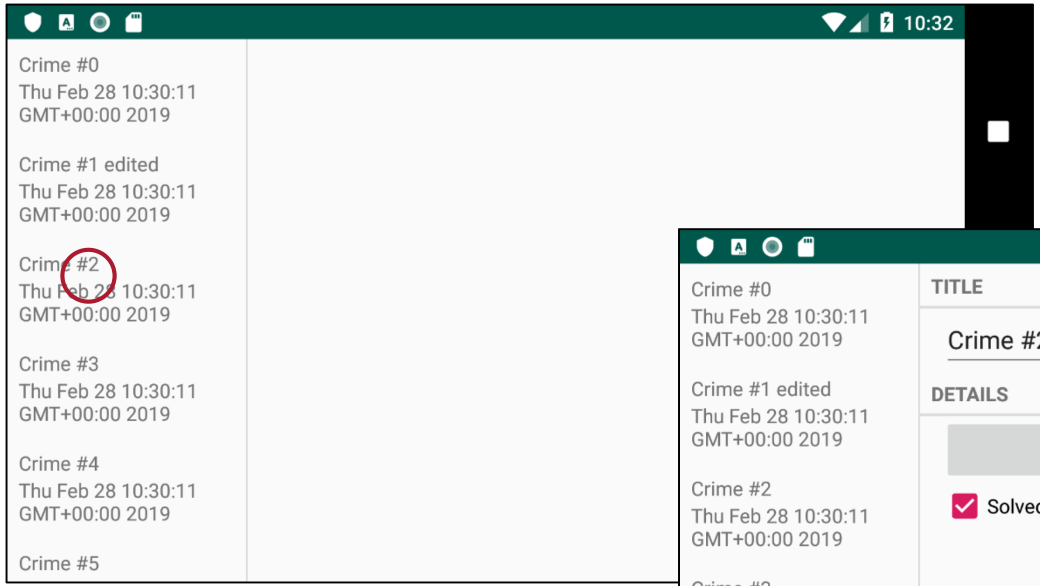
Crime #8
Thu Feb 28 10:30:11 GMT+00:00 2019

Crime #9
Thu Feb 28 10:30:11 GMT+00:00 2019

Crime #10
Thu Feb 28 10:30:11 GMT+00:00 2019

Crime #11
Thu Feb 28 10:30:11 GMT+00:00 2019

Recap: List-Detail UI with Fragments (Expected Landscape UX)


10:32

Crime #0
Thu Feb 28 10:30:11 GMT+00:00 2019

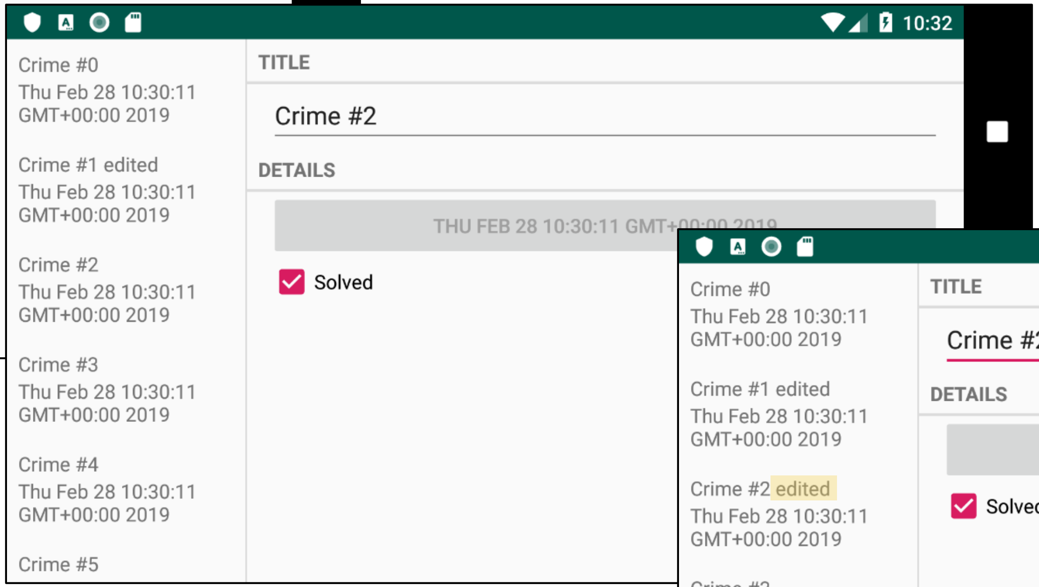
Crime #1 edited
Thu Feb 28 10:30:11 GMT+00:00 2019

Crime #2
Thu Feb 28 10:30:11 GMT+00:00 2019

Crime #3
Thu Feb 28 10:30:11 GMT+00:00 2019

Crime #4
Thu Feb 28 10:30:11 GMT+00:00 2019

Crime #5


10:32

Crime #0
Thu Feb 28 10:30:11 GMT+00:00 2019

Crime #1 edited
Thu Feb 28 10:30:11 GMT+00:00 2019

Crime #2
Thu Feb 28 10:30:11 GMT+00:00 2019

Crime #3
Thu Feb 28 10:30:11 GMT+00:00 2019

Crime #4
Thu Feb 28 10:30:11 GMT+00:00 2019

Crime #5

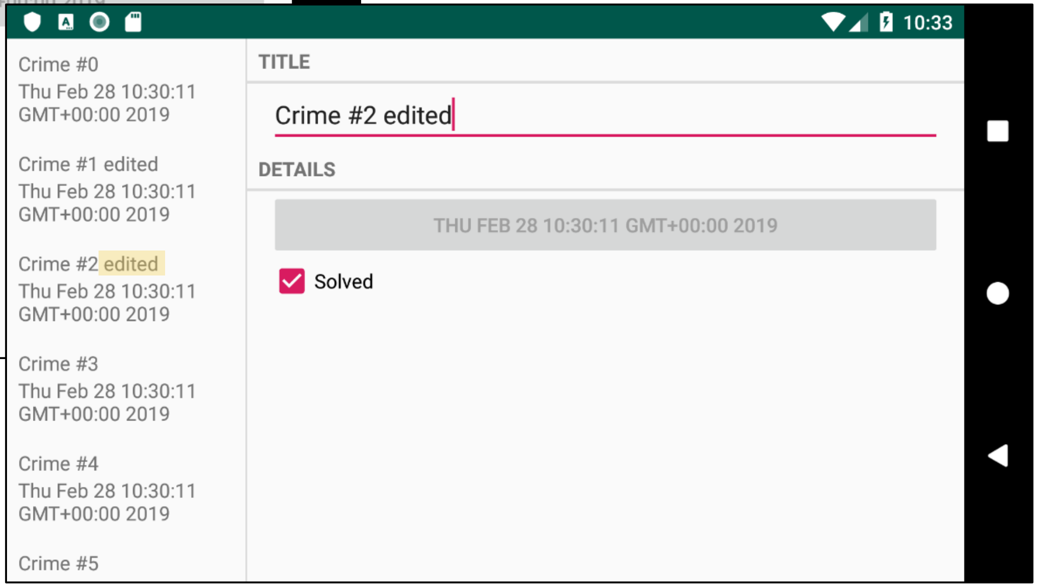
TITLE

Crime #2

DETAILS

THU FEB 28 10:30:11 GMT+00:00 2019

✓ Solved


10:33

Crime #0
Thu Feb 28 10:30:11 GMT+00:00 2019

Crime #1 edited
Thu Feb 28 10:30:11 GMT+00:00 2019

Crime #2 edited
Thu Feb 28 10:30:11 GMT+00:00 2019

Crime #3
Thu Feb 28 10:30:11 GMT+00:00 2019

Crime #4
Thu Feb 28 10:30:11 GMT+00:00 2019

Crime #5

TITLE

Crime #2 edited

DETAILS

THU FEB 28 10:30:11 GMT+00:00 2019

✓ Solved

Recap: List-Detail UI Design Model

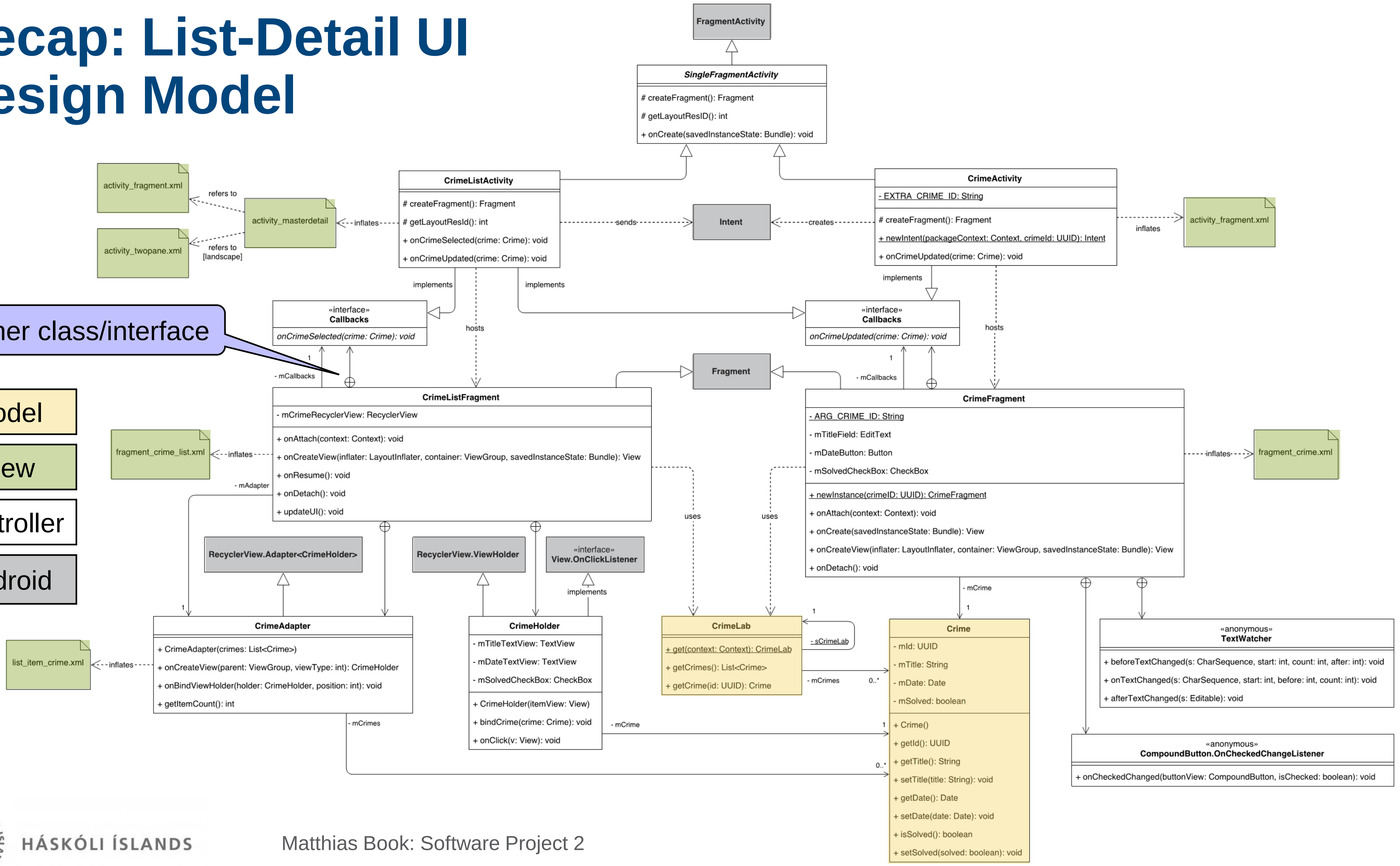
Inner class/interface

Model

View

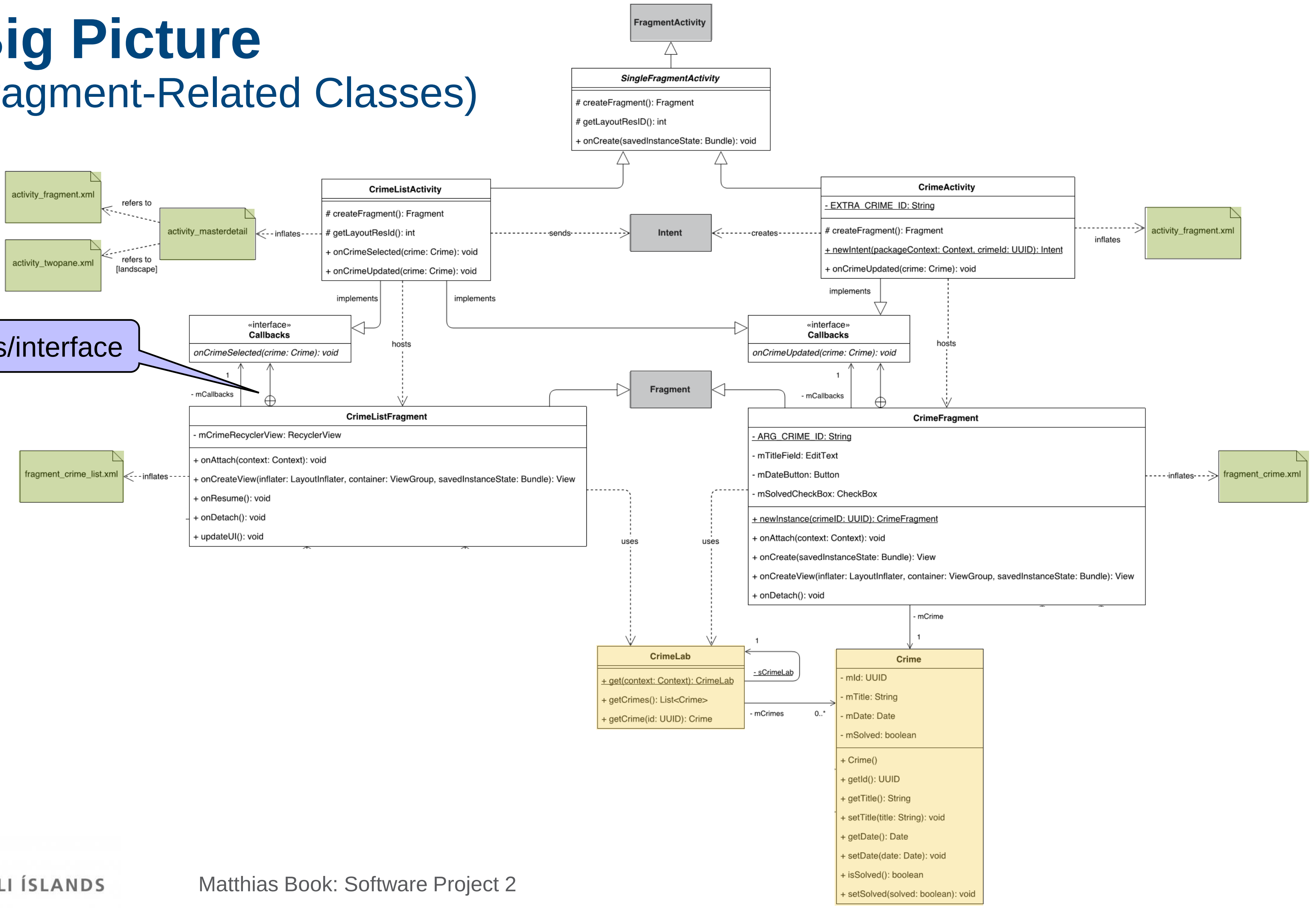
Controller

Android


FragmentActivity
SingleFragmentActivity
# createFragment(): Fragment
# getLayoutResID(): int
+ onCreate(savedInstanceState: Bundle): void
activity_fragment.xml
refers to
activity_masterdetail
activity_twopane.xml
refers to [landscape]
inflates
CrimeListActivity
# createFragment(): Fragment
# getLayoutResId(): int
+ onCrimeSelected(crime: Crime): void
+ onCrimeUpdated(crime: Crime): void
sends
Intent
creates
CrimeActivity
- EXTRA_CRIME_ID: String
# createFragment(): Fragment
+ newIntent(packageContext: Context, crimeId: UUID): Intent
+ onCrimeUpdated(crime: Crime): void
inflates
activity_fragment.xml
implements
implements
implements
«interface» Callbacks
onCrimeSelected(crime: Crime): void
hosts
«interface» Callbacks
onCrimeUpdated(crime: Crime): void
hosts
1
- mCallbacks
Fragment
1
- mCallbacks
CrimeListFragment
- mCrimeRecyclerView: RecyclerView
+ onAttach(context: Context): void
+ onCreateView(inflater: LayoutInflater, container: ViewGroup, savedInstanceState: Bundle): View
+ onResume(): void
+ onDetach(): void
+ updateUI(): void
fragment_crime_list.xml
inflates
- mAdapter
CrimeFragment
- ARG_CRIME_ID: String
- mTitleField: EditText
- mDateButton: Button
- mSolvedCheckBox: CheckBox
+ newInstance(crimeID: UUID): CrimeFragment
+ onAttach(context: Context): void
+ onCreate(savedInstanceState: Bundle): View
+ onCreateView(inflater: LayoutInflater, container: ViewGroup, savedInstanceState: Bundle): View
+ onDetach(): void
inflates
fragment_crime.xml
uses
uses
RecyclerView.Adapter<CrimeHolder>
RecyclerView.ViewHolder
«interface» View.OnClickListener
implements
1
CrimeAdapter
+ CrimeAdapter(crimes: List<Crime>)
+ onCreateView(parent: ViewGroup, viewType: int): CrimeHolder
+ onBindViewHolder(holder: CrimeHolder, position: int): void
+ getItemCount(): int
list_item_crime.xml
inflates
- mCrimes
CrimeHolder
- mTitleTextView: TextView
- mDateTextView: TextView
- mSolvedCheckBox: CheckBox
+ CrimeHolder(itemView: View)
+ bindCrime(crime: Crime): void
+ onClick(v: View): void
- mCrime
1
CrimeLab
1
- sCrimeLab
+ get(context: Context): CrimeLab
+ getCrimes(): List<Crime>
+ getCrime(id: UUID): Crime
- mCrimes
0..*
- mCrime
1
Crime
- mId: UUID
- mTitle: String
- mDate: Date
- mSolved: boolean
+ Crime()
+ getId(): UUID
+ getTitle(): String
+ setTitle(title: String): void
+ getDate(): Date
+ setDate(date: Date): void
+ isSolved(): boolean
+ setSolved(solved: boolean): void
0..*
«anonymous» TextWatcher
+ beforeTextChanged(s: CharSequence, start: int, count: int, after: int): void
+ onTextChanged(s: CharSequence, start: int, before: int, count: int): void
+ afterTextChanged(s: Editable): void
«anonymous» CompoundButton.OnCheckedChangeListener
+ onCheckedChanged(buttonView: CompoundButton, isChecked: boolean): void

The Big Picture
(Only Fragment-Related Classes)

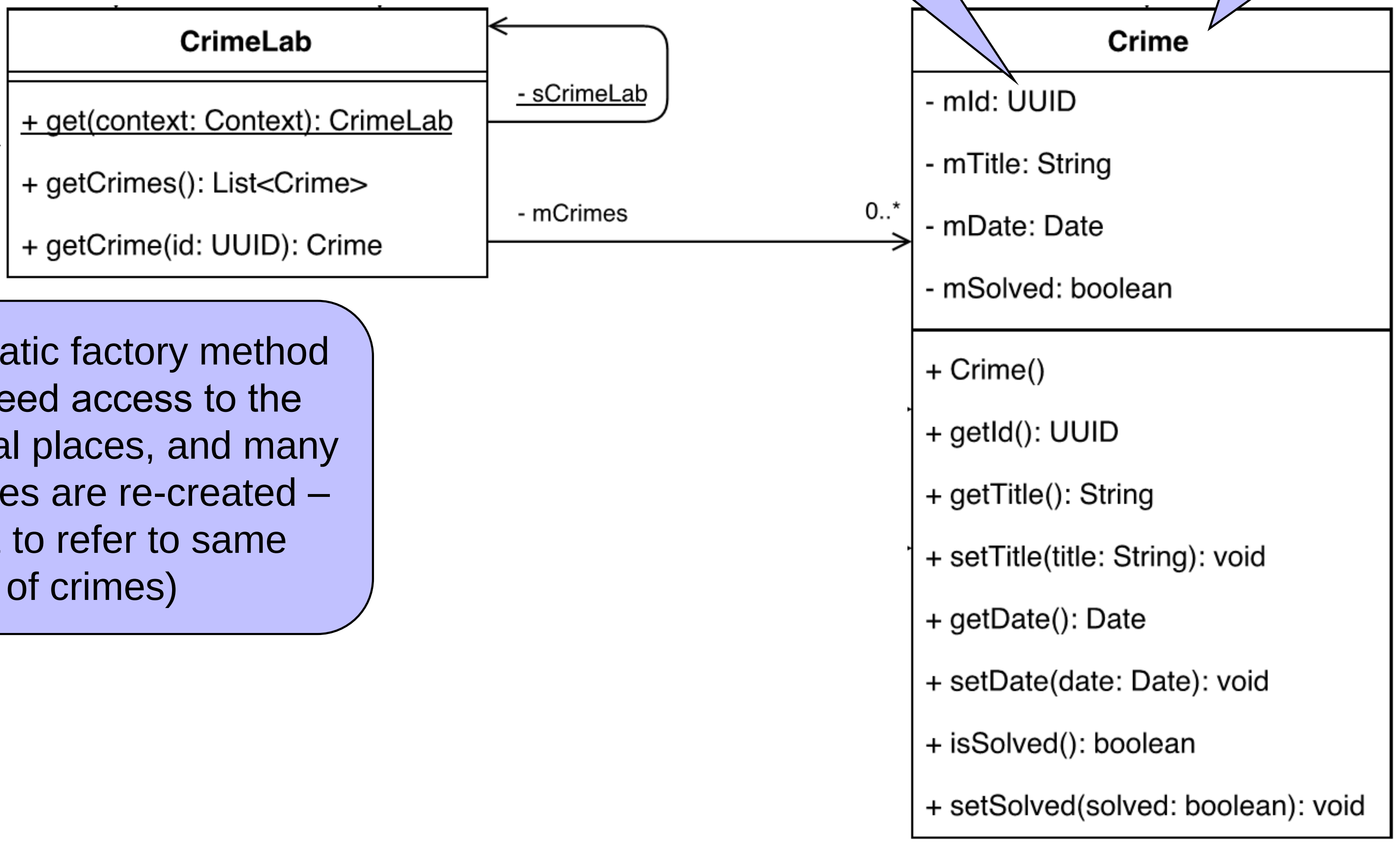
Inner class/interface



FragmentActivity
SingleFragmentActivity
# createFragment(): Fragment
# getLayoutResID(): int
+ onCreate(savedInstanceState: Bundle): void
activity_fragment.xml
refers to
activity_masterdetail
activity_twopane.xml
refers to [landscape]
inflates
CrimeListActivity
# createFragment(): Fragment
# getLayoutResId(): int
+ onCrimeSelected(crime: Crime): void
+ onCrimeUpdated(crime: Crime): void
sends
Intent
creates
CrimeActivity
- EXTRA_CRIME_ID: String
# createFragment(): Fragment
+ newIntent(packageContext: Context, crimeId: UUID): Intent
+ onCrimeUpdated(crime: Crime): void
inflates
activity_fragment.xml
implements
implements
implements
«interface» Callbacks
onCrimeSelected(crime: Crime): void
hosts
«interface» Callbacks
onCrimeUpdated(crime: Crime): void
hosts
1
- mCallbacks
Fragment
1
- mCallbacks
CrimeListFragment
- mCrimeRecyclerView: RecyclerView
+ onAttach(context: Context): void
+ onCreateView(inflater: LayoutInflater, container: ViewGroup, savedInstanceState: Bundle): View
+ onResume(): void
+ onDetach(): void
+ updateUI(): void
fragment_crime_list.xml
inflates
CrimeFragment
- ARG_CRIME_ID: String
- mTitleField: EditText
- mDateButton: Button
- mSolvedCheckBox: CheckBox
+ newInstance(crimeID: UUID): CrimeFragment
+ onAttach(context: Context): void
+ onCreate(savedInstanceState: Bundle): View
+ onCreateView(inflater: LayoutInflater, container: ViewGroup, savedInstanceState: Bundle): View
+ onDetach(): void
inflates
fragment_crime.xml
uses
uses
- mCrime
1
1
CrimeLab
+ get(context: Context): CrimeLab
+ getCrimes(): List<Crime>
+ getCrime(id: UUID): Crime
- sCrimeLab
- mCrimes
0..*
Crime
- mId: UUID
- mTitle: String
- mDate: Date
- mSolved: boolean
+ Crime()
+ getId(): UUID
+ getTitle(): String
+ setTitle(title: String): void
+ getDate(): Date
+ setDate(date: Date): void
+ isSolved(): boolean
+ setSolved(solved: boolean): void

Model
View
Controller
Android

The Model: Crime and CrimeLab

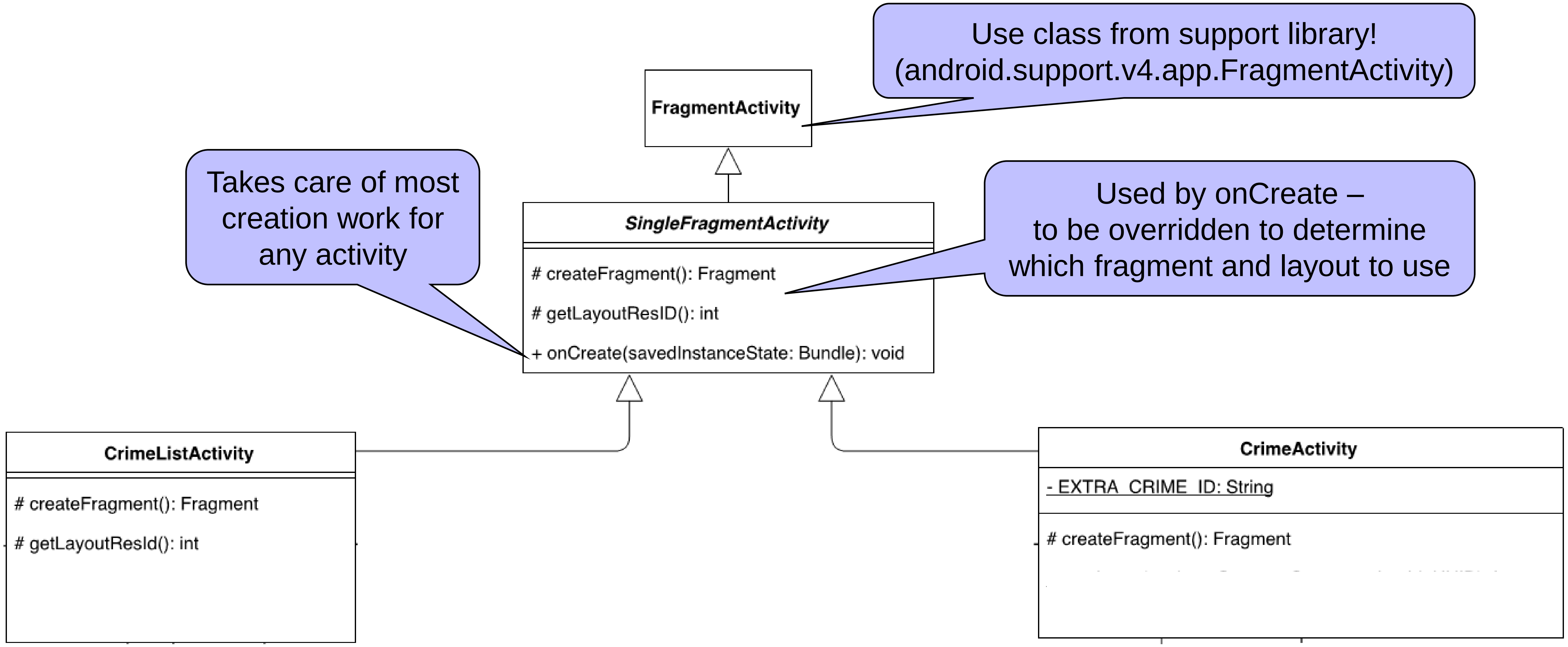

CrimeLab
+ get(context: Context): CrimeLab
+ getCrimes(): List<Crime>
+ getCrime(id: UUID): Crime
- sCrimeLab
- mCrimes
0..*
Crime
- mId: UUID
- mTitle: String
- mDate: Date
- mSolved: boolean
+ Crime()
+ getId(): UUID
+ getTitle(): String
+ setTitle(title: String): void
+ getDate(): Date
+ setDate(date: Date): void
+ isSolved(): boolean
+ setSolved(solved: boolean): void

java.util.UUID, a unique identifier

POJO representing an “office crime”

Singleton with a static factory method (Reason: We’ll need access to the CrimeLab in several places, and many times when activities are re-created – but always want to refer to same collection of crimes)

The Activities: CrimeListActivity and CrimeActivity


Use class from support library! (android.support.v4.app.FragmentActivity)
FragmentActivity
Takes care of most creation work for any activity
SingleFragmentActivity
# createFragment(): Fragment
# getLayoutResID(): int
+ onCreate(savedInstanceState: Bundle): void
Used by onCreate – to be overridden to determine which fragment and layout to use
CrimeListActivity
# createFragment(): Fragment
# getLayoutResId(): int
CrimeActivity
- EXTRA_CRIME_ID: String
# createFragment(): Fragment

The Abstract Superclass SingleFragmentActivity

```
public abstract class SingleFragmentActivity extends FragmentActivity {

    protected abstract Fragment createFragment();

    @LayoutRes
    protected int getLayoutResId() {
        return R.layout.activity_fragment;
    }

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(getLayoutResId());

        FragmentManager fm = getSupportFragmentManager();
        Fragment fragment = fm.findFragmentById(R.id.fragment_container);

        if (fragment == null) {
            fragment = createFragment();
            fm.beginTransaction()
                    .add(R.id.fragment_container, fragment)
                    .commit();
        }
    }
}
```

Used by onCreate – to be overridden to determine which fragment and layout to use

This can be used in any project to incorporate single fragments into their host activities

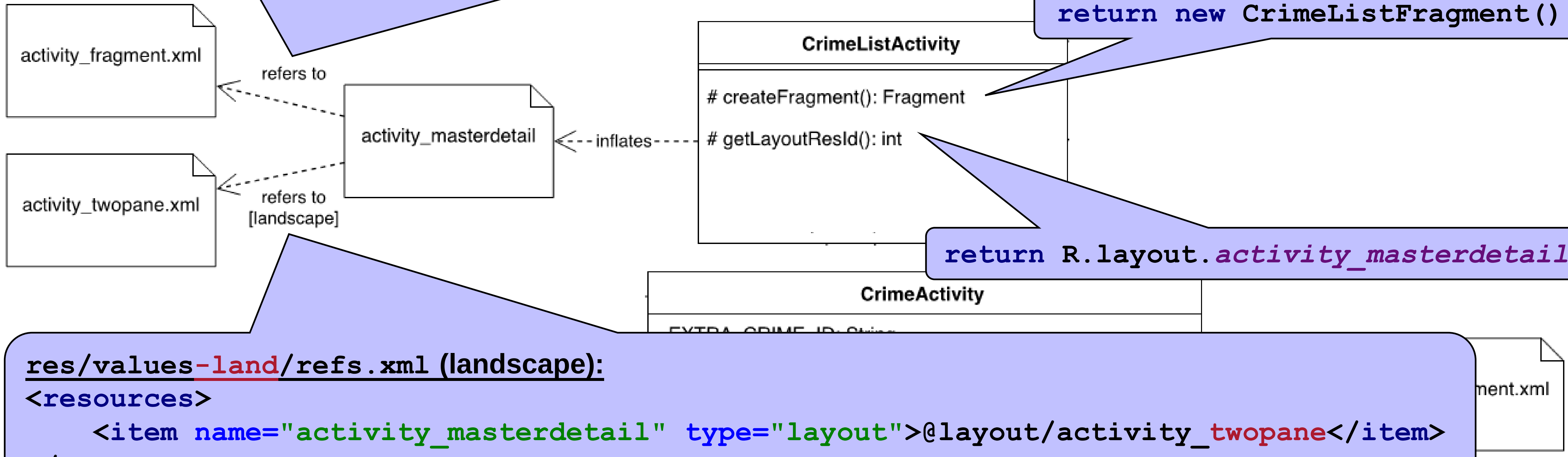
Set the layout

Try to retrieve the fragment from the layout

Fragment may already exist if activity is re-created

Instantiate fragment and insert it into the layout

The Activity Layouts: activity_fragment and activity_twopane



res/values/refs.xml (default: portrait):

```
<resources>
    <item name="activity_masterdetail" type="layout">@layout/activity_fragment</item>
</resources>
```

```
return new CrimeListFragment();
```

```
return R.layout.activity_masterdetail;
```

res/values-land/refs.xml (landscape):

```
<resources>
    <item name="activity_masterdetail" type="layout">@layout/activity_twopane</item>
</resources>
```

The Activity Layouts: activity_fragment and activity_twopane

activity_fragment.xml

```
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/fragment_container"
    android:layout_width="match_parent" android:layout_height="match_parent"/>
```

Generic one-pane layout

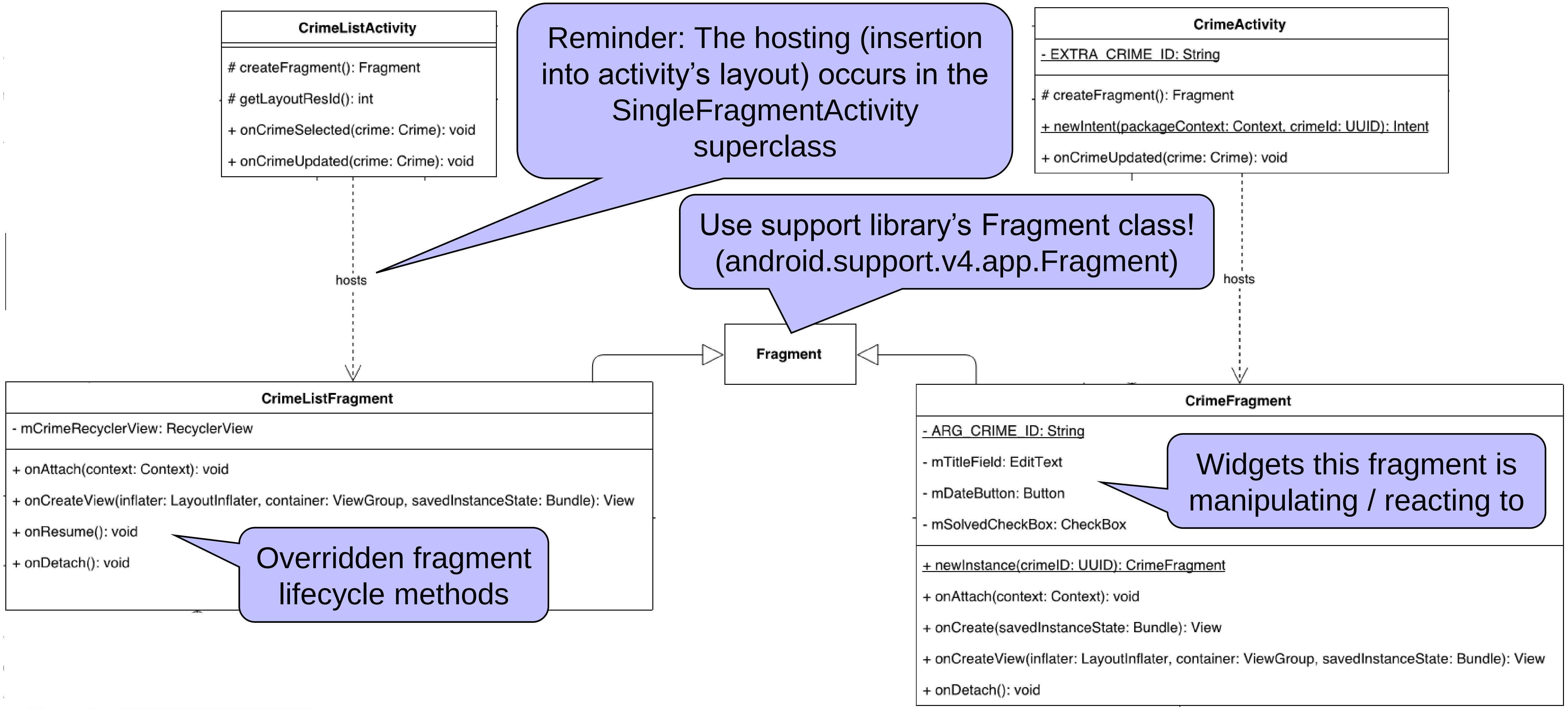
activity_twopane.xml

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent" android:layout_height="match_parent"
    android:divider="?android:attr/dividerHorizontal" android:showDividers="middle"
    android:orientation="horizontal">
  <FrameLayout android:id="@+id/fragment_container"
      android:layout_width="0dp" android:layout_height="match_parent"
      android:layout_weight="1"/>
  <FrameLayout android:id="@+id/detail_fragment_container"
      android:layout_width="0dp" android:layout_height="match_parent"
      android:layout_weight="3"/>
</LinearLayout>
```

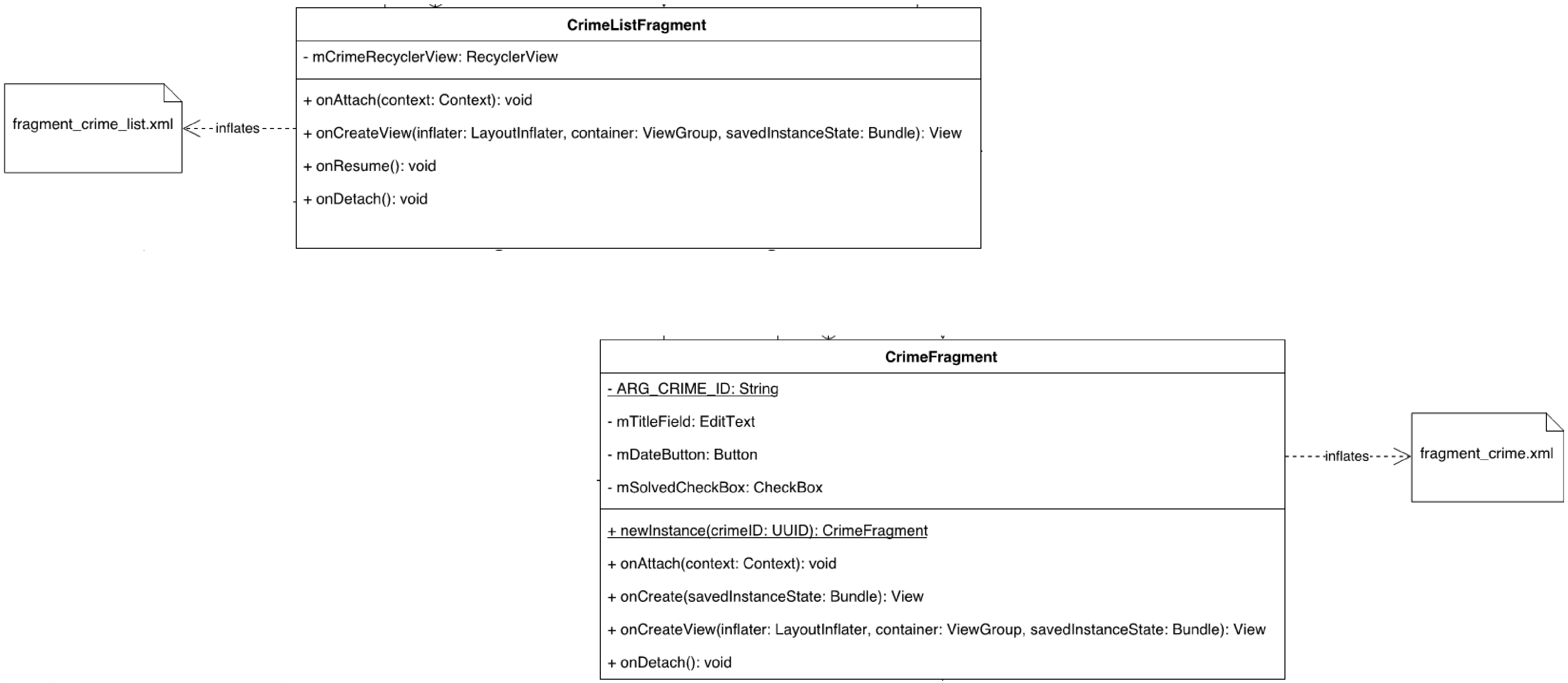
Two panes with 1:3 size relation

Note: SingleFragmentActivity will only populate the fragment_container. The detail_fragment_container initially remains empty.

The Fragments: CrimeListFragment and CrimeFragment


CrimeListActivity
# createFragment(): Fragment
# getLayoutResId(): int
+ onCrimeSelected(crime: Crime): void
+ onCrimeUpdated(crime: Crime): void
Reminder: The hosting (insertion into activity's layout) occurs in the SingleFragmentActivity superclass
CrimeActivity
- EXTRA_CRIME_ID: String
# createFragment(): Fragment
+ newIntent(packageContext: Context, crimeId: UUID): Intent
+ onCrimeUpdated(crime: Crime): void
Use support library's Fragment class! (android.support.v4.app.Fragment)
hosts
hosts
Fragment
CrimeListFragment
- mCrimeRecyclerView: RecyclerView
+ onAttach(context: Context): void
+ onCreateView(inflater: LayoutInflater, container: ViewGroup, savedInstanceState: Bundle): View
+ onResume(): void
+ onDetach(): void
Overridden fragment lifecycle methods
CrimeFragment
- ARG_CRIME_ID: String
- mTitleField: EditText
- mDateButton: Button
- mSolvedCheckBox: CheckBox
Widgets this fragment is manipulating / reacting to
+ newInstance(crimeID: UUID): CrimeFragment
+ onAttach(context: Context): void
+ onCreate(savedInstanceState: Bundle): View
+ onCreateView(inflater: LayoutInflater, container: ViewGroup, savedInstanceState: Bundle): View
+ onDetach(): void

The Fragment Layouts: fragment_crime_list and fragment_crime



fragment_crime_list.xml
inflates
CrimeListFragment
- mCrimeRecyclerView: RecyclerView
+ onAttach(context: Context): void
+ onCreateView(inflater: LayoutInflater, container: ViewGroup, savedInstanceState: Bundle): View
+ onResume(): void
+ onDetach(): void
CrimeFragment
- ARG_CRIME_ID: String
- mTitleField: EditText
- mDateButton: Button
- mSolvedCheckBox: CheckBox
+ newInstance(crimeID: UUID): CrimeFragment
+ onAttach(context: Context): void
+ onCreate(savedInstanceState: Bundle): View
+ onCreateView(inflater: LayoutInflater, container: ViewGroup, savedInstanceState: Bundle): View
+ onDetach(): void
inflates
fragment_crime.xml

The Fragment Layouts: fragment_crime_list and fragment_crime

Layout of the list *items* will be defined in separate file (see Android textbook, Ch. 8)

List layout

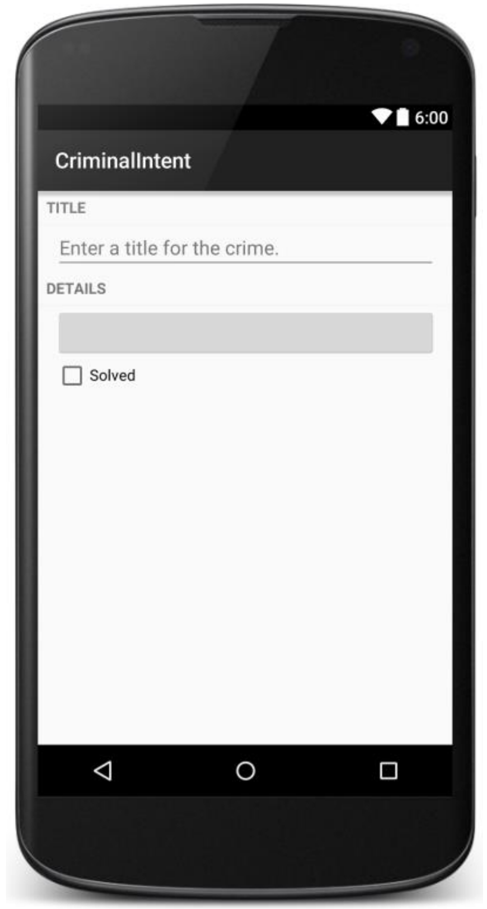
fragment_crime_list.xml

```
<android.support.v7.widget.RecyclerView xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/crime_recycler_view"
    android:layout_width="match_parent" android:layout_height="match_parent"/>
```

Detail layout

fragment_crime.xml

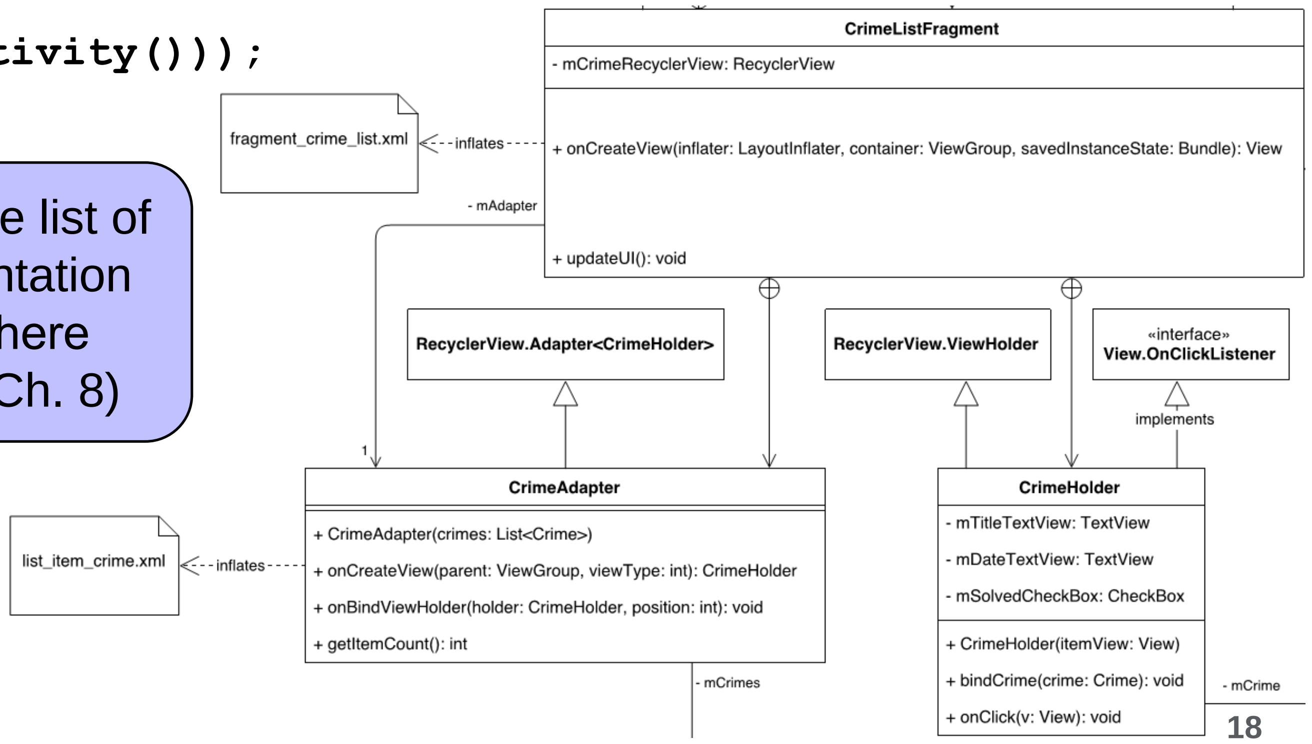
```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="match_parent" android:layout_height="match_parent">

  <TextView android:text="@string/crime_title_label"
      android:layout_width="match_parent" android:layout_height="wrap_content"
      style="?android:listSeparatorTextViewStyle"/>

  <EditText android:id="@+id/crime_title"
      android:layout_width="match_parent" android:layout_height="wrap_content"
      android:layout_marginLeft="16dp" android:layout_marginRight="16dp"
      android:hint="@string/crime_title_hint"/>

  <TextView android:text="@string/crime_details_label"
      android:layout_width="match_parent" android:layout_height="wrap_content"
      style="?android:listSeparatorTextViewStyle"/>

  <Button android:id="@+id/crime_date"
      android:layout_width="match_parent" android:layout_height="wrap_content"
      android:layout_marginLeft="16dp" android:layout_marginRight="16dp"/>

  <CheckBox android:id="@+id/crime_solved"
      android:layout_width="match_parent" android:layout_height="wrap_content"
      android:layout_marginLeft="16dp" android:layout_marginRight="16dp"
      android:text="@string/crime_solved_label"/>

</LinearLayout>
```



Creating the List Fragment: CrimeListFragment.onCreateView

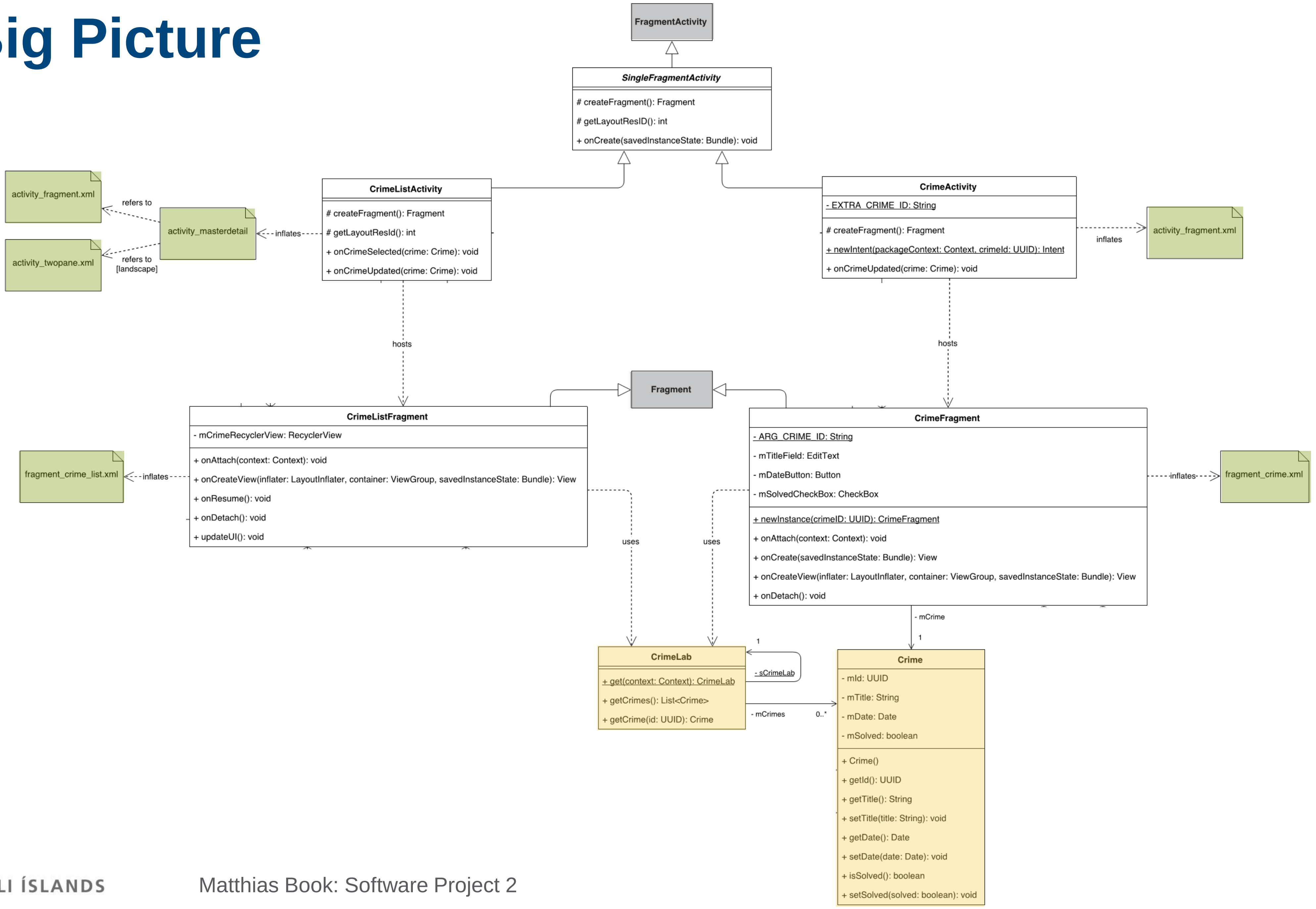
```
@Override
public View onCreateView(LayoutInflater inflater, ViewGroup container,
                         Bundle savedInstanceState) {
    View view = inflater.inflate(R.layout.fragment_crime_list, container, false);

    mCrimeRecyclerView = (RecyclerView) view
            .findViewById(R.id.crime_recycler_view);
    mCrimeRecyclerView.setLayoutManager(
            new LinearLayoutManager(getActivity()));

    updateUI();

    return view;
}
```

This creates the scrollable list of crimes, whose implementation details we'll skip over here (see Android textbook, Ch. 8)

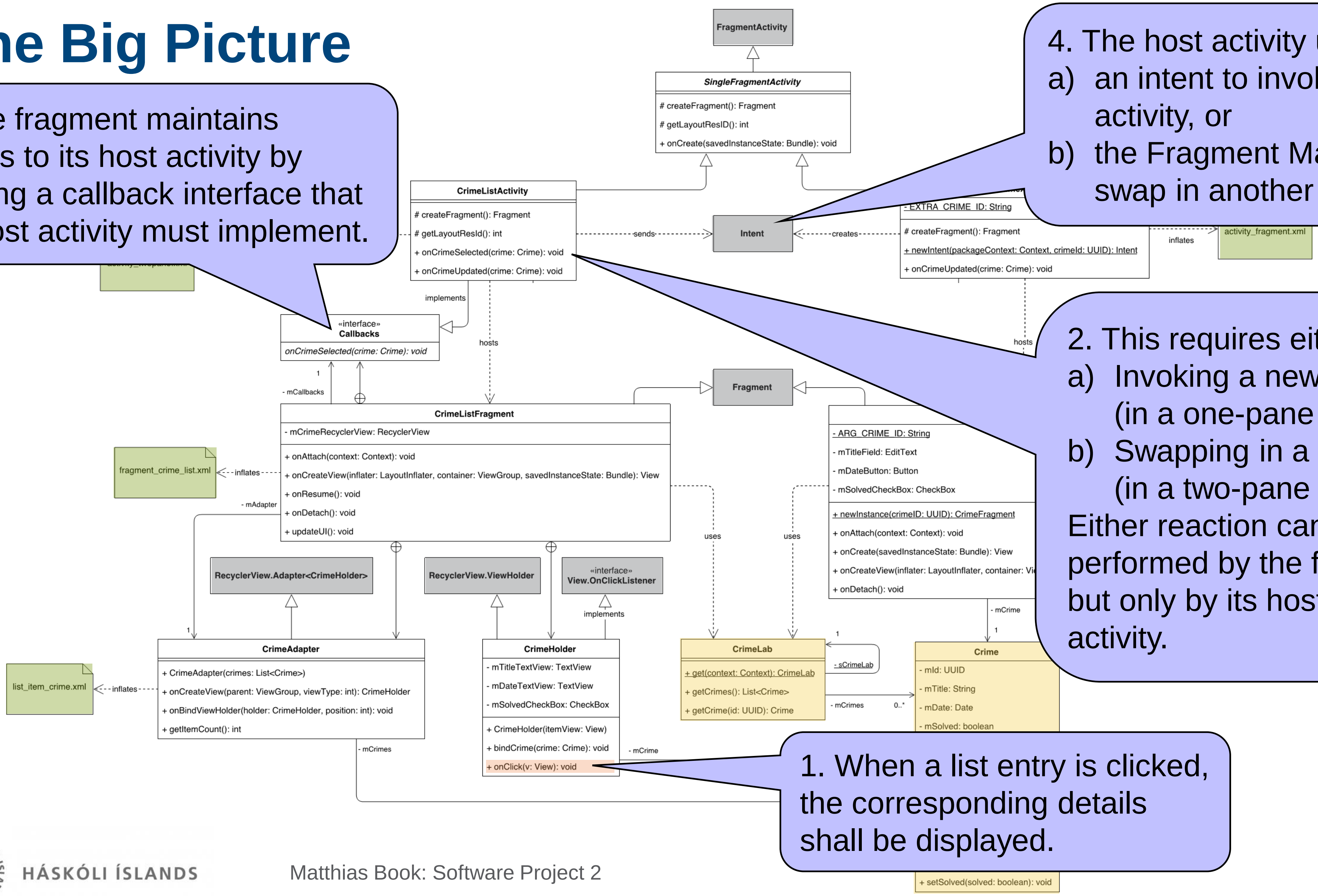


CrimeListFragment
- mCrimeRecyclerView: RecyclerView
+ onCreateView(inflater: LayoutInflater, container: ViewGroup, savedInstanceState: Bundle): View
+ updateUI(): void
fragment_crime_list.xml
inflates
- mAdapter
1
RecyclerView.Adapter<CrimeHolder>
RecyclerView.ViewHolder
«interface» View.OnClickListener
implements
CrimeAdapter
+ CrimeAdapter(crimes: List<Crime>)
+ onCreateView(parent: ViewGroup, viewType: int): CrimeHolder
+ onBindViewHolder(holder: CrimeHolder, position: int): void
+ getItemCount(): int
list_item_crime.xml
inflates
- mCrimes
CrimeHolder
- mTitleTextView: TextView
- mDateTextView: TextView
- mSolvedCheckBox: CheckBox
+ CrimeHolder(itemView: View)
+ bindCrime(crime: Crime): void
+ onClick(v: View): void
- mCrime

The Big Picture


FragmentActivity
SingleFragmentActivity
# createFragment(): Fragment
# getLayoutResID(): int
+ onCreate(savedInstanceState: Bundle): void
CrimeListActivity
# createFragment(): Fragment
# getLayoutResId(): int
+ onCrimeSelected(crime: Crime): void
+ onCrimeUpdated(crime: Crime): void
CrimeActivity
- EXTRA_CRIME_ID: String
# createFragment(): Fragment
+ newIntent(packageContext: Context, crimeId: UUID): Intent
+ onCrimeUpdated(crime: Crime): void
activity_fragment.xml
activity_twopane.xml
activity_masterdetail
refers to
refers to [landscape]
inflates
activity_fragment.xml
inflates
hosts
hosts
Fragment
CrimeListFragment
- mCrimeRecyclerView: RecyclerView
+ onAttach(context: Context): void
+ onCreateView(inflater: LayoutInflater, container: ViewGroup, savedInstanceState: Bundle): View
+ onResume(): void
+ onDetach(): void
+ updateUI(): void
fragment_crime_list.xml
inflates
CrimeFragment
- ARG_CRIME_ID: String
- mTitleField: EditText
- mDateButton: Button
- mSolvedCheckBox: CheckBox
+ newInstance(crimeID: UUID): CrimeFragment
+ onAttach(context: Context): void
+ onCreate(savedInstanceState: Bundle): View
+ onCreateView(inflater: LayoutInflater, container: ViewGroup, savedInstanceState: Bundle): View
+ onDetach(): void
inflates
fragment_crime.xml
uses
uses
- mCrime
1
CrimeLab
+ get(context: Context): CrimeLab
+ getCrimes(): List<Crime>
+ getCrime(id: UUID): Crime
1
- sCrimeLab
- mCrimes
0..*
Crime
- mId: UUID
- mTitle: String
- mDate: Date
- mSolved: boolean
+ Crime()
+ getId(): UUID
+ getTitle(): String
+ setTitle(title: String): void
+ getDate(): Date
+ setDate(date: Date): void
+ isSolved(): boolean
+ setSolved(solved: boolean): void

The Big Picture


FragmentActivity
SingleFragmentActivity
# createFragment(): Fragment
# getLayoutResID(): int
+ onCreate(savedInstanceState: Bundle): void
CrimeListActivity
# createFragment(): Fragment
# getLayoutResId(): int
+ onCrimeSelected(crime: Crime): void
+ onCrimeUpdated(crime: Crime): void
sends
Intent
creates
- EXTRA_CRIME_ID: String
# createFragment(): Fragment
+ newIntent(packageContext: Context, crimeId: UUID): Intent
+ onCrimeUpdated(crime: Crime): void
inflates
activity_fragment.xml
implements
«interface»
Callbacks
onCrimeSelected(crime: Crime): void
hosts
hosts
1
- mCallbacks
Fragment
CrimeListFragment
- mCrimeRecyclerView: RecyclerView
+ onAttach(context: Context): void
+ onCreateView(inflater: LayoutInflater, container: ViewGroup, savedInstanceState: Bundle): View
+ onResume(): void
+ onDetach(): void
+ updateUI(): void
fragment_crime_list.xml
inflates
- mAdapter
- ARG_CRIME_ID: String
- mTitleField: EditText
- mDateButton: Button
- mSolvedCheckBox: CheckBox
+ newInstance(crimeID: UUID): CrimeFragment
+ onAttach(context: Context): void
+ onCreate(savedInstanceState: Bundle): View
+ onCreateView(inflater: LayoutInflater, container: Vi
+ onDetach(): void
uses
uses
RecyclerView.Adapter<CrimeHolder>
RecyclerView.ViewHolder
«interface»
View.OnClickListener
implements
1
CrimeAdapter
+ CrimeAdapter(crimes: List<Crime>)
+ onCreateView(parent: ViewGroup, viewType: int): CrimeHolder
+ onBindViewHolder(holder: CrimeHolder, position: int): void
+ getItemCount(): int
list_item_crime.xml
inflates
- mCrimes
CrimeHolder
- mTitleTextView: TextView
- mDateTextView: TextView
- mSolvedCheckBox: CheckBox
+ CrimeHolder(itemView: View)
+ bindCrime(crime: Crime): void
+ onClick(v: View): void
- mCrime
CrimeLab
+ get(context: Context): CrimeLab
+ getCrimes(): List<Crime>
+ getCrime(id: UUID): Crime
1
- sCrimeLab
- mCrimes
0..*
- mCrime
1
Crime
- mId: UUID
- mTitle: String
- mDate: Date
- mSolved: boolean
+ setSolved(solved: boolean): void

1. When a list entry is clicked, the corresponding details shall be displayed.

2. This requires either:
   a) Invoking a new activity (in a one-pane layout), or
   b) Swapping in a fragment (in a two-pane layout)

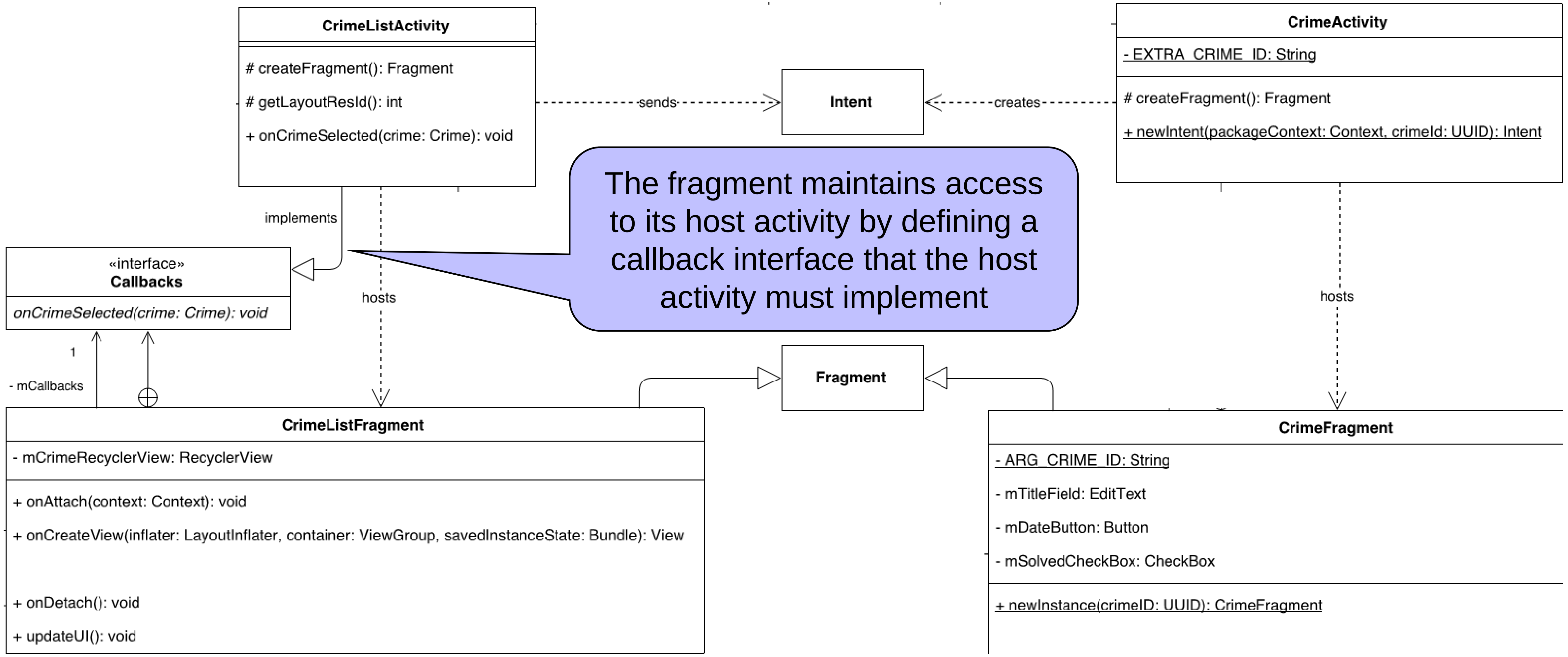
   Either reaction cannot be performed by the fragment, but only by its hosting activity.

3. The fragment maintains access to its host activity by defining a callback interface that the host activity must implement.

4. The host activity uses either:
   a) an intent to invoke another activity, or
   b) the Fragment Manager to swap in another fragment

The List’s Callback Interface: CrimeListFragment.Callbacks

This is how a fragment sends messages to its host activity


CrimeListActivity
# createFragment(): Fragment
# getLayoutResId(): int
+ onCrimeSelected(crime: Crime): void
sends
Intent
creates
CrimeActivity
- EXTRA_CRIME_ID: String
# createFragment(): Fragment
+ newIntent(packageContext: Context, crimeId: UUID): Intent
implements
hosts
«interface» Callbacks
onCrimeSelected(crime: Crime): void
1
- mCallbacks
hosts
Fragment
CrimeListFragment
- mCrimeRecyclerView: RecyclerView
+ onAttach(context: Context): void
+ onCreateView(inflater: LayoutInflater, container: ViewGroup, savedInstanceState: Bundle): View
+ onDetach(): void
+ updateUI(): void
CrimeFragment
- ARG_CRIME_ID: String
- mTitleField: EditText
- mDateButton: Button
- mSolvedCheckBox: CheckBox
+ newInstance(crimeID: UUID): CrimeFragment

The fragment maintains access to its host activity by defining a callback interface that the host activity must implement

Declaring the List’s Callback Interface: CrimeListFragment.Callbacks

```
public class CrimeListFragment extends Fragment {

    private Callbacks mCallbacks;

    public interface Callbacks {
        void onCrimeSelected(Crime crime);
    }

    @Override
    public void onAttach(Context context) {
        super.onAttach(context);
        mCallbacks = (Callbacks) context;
    }

    @Override
    public void onDetach() {
        super.onDetach();
        mCallbacks = null;
    }

    // ...
}
```

Reference to host activity

Declaration of interface that the host activity must implement to react to a selection of a list entry

Automatically stores a reference to the host activity (here: CrimeListActivity) when the fragment is associated with it

Automatically removes the reference to the host activity when the fragment is dissociated from it

Invoking the List's Callback Interface: CrimeListFragment.CrimeHolder.onClick

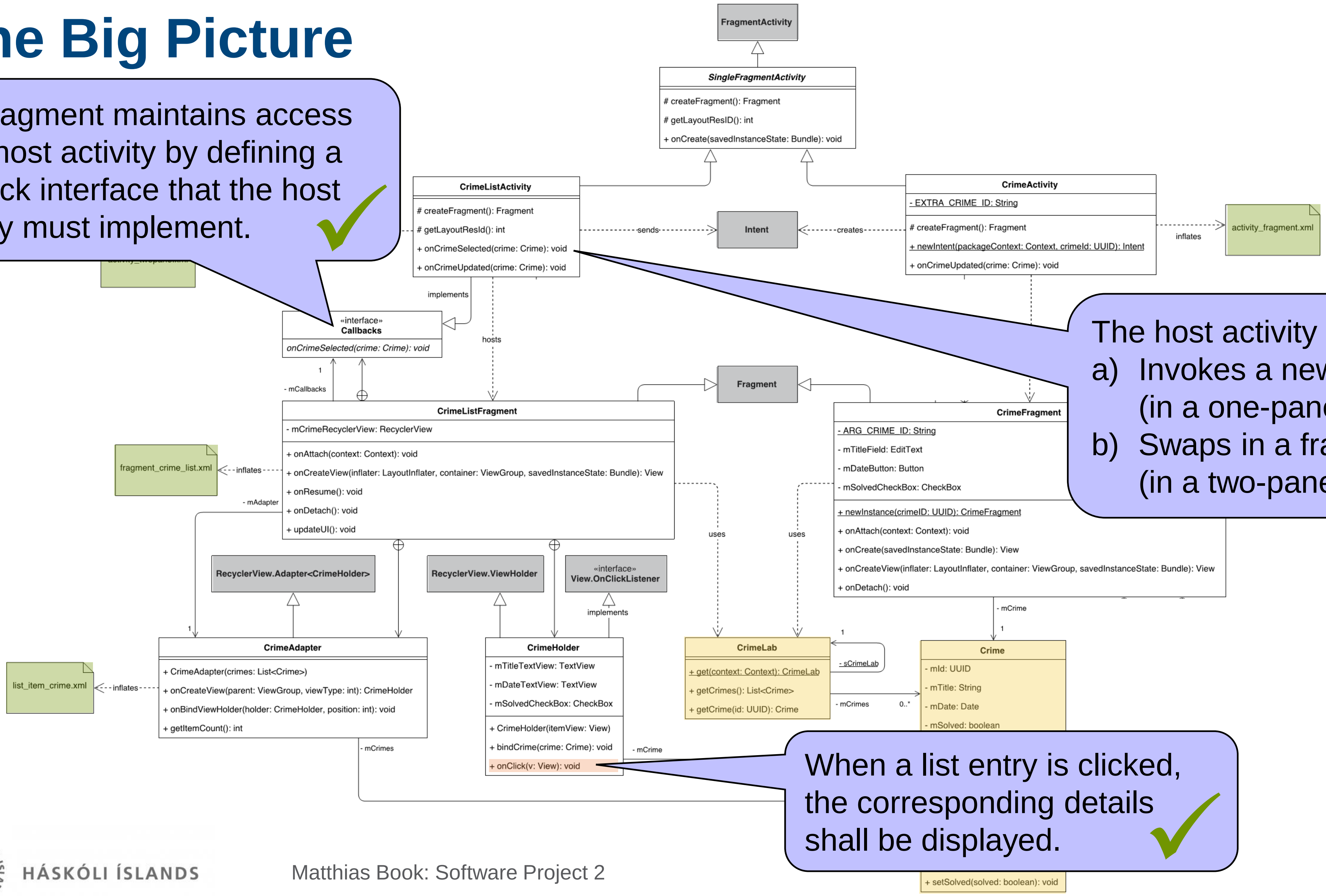
See Android textbook, Ch. 8 for scrolling list implementation details

The scrolling list's event handler for clicks on an item

Notify host activity (via callback reference) of selection of an item

```
private class CrimeHolder
        extends RecyclerView.ViewHolder
        implements View.OnClickListener {

    private Crime mCrime;

    @Override
    public void onClick(View v) {
        mCallbacks.onCrimeSelected(mCrime);
    }

    // ...
}
```

The Big Picture



FragmentActivity
SingleFragmentActivity
# createFragment(): Fragment
# getLayoutResID(): int
+ onCreate(savedInstanceState: Bundle): void
CrimeListActivity
# createFragment(): Fragment
# getLayoutResId(): int
+ onCrimeSelected(crime: Crime): void
+ onCrimeUpdated(crime: Crime): void
sends
Intent
creates
CrimeActivity
- EXTRA_CRIME_ID: String
# createFragment(): Fragment
+ newIntent(packageContext: Context, crimeId: UUID): Intent
+ onCrimeUpdated(crime: Crime): void
inflates
activity_fragment.xml
implements
«interface» Callbacks
onCrimeSelected(crime: Crime): void
hosts
1
- mCallbacks
Fragment
CrimeListFragment
- mCrimeRecyclerView: RecyclerView
+ onAttach(context: Context): void
+ onCreateView(inflater: LayoutInflater, container: ViewGroup, savedInstanceState: Bundle): View
+ onResume(): void
+ onDetach(): void
+ updateUI(): void
fragment_crime_list.xml
inflates
- mAdapter
CrimeFragment
- ARG_CRIME_ID: String
- mTitleField: EditText
- mDateButton: Button
- mSolvedCheckBox: CheckBox
+ newInstance(crimeID: UUID): CrimeFragment
+ onAttach(context: Context): void
+ onCreate(savedInstanceState: Bundle): View
+ onCreateView(inflater: LayoutInflater, container: ViewGroup, savedInstanceState: Bundle): View
+ onDetach(): void
uses
uses
RecyclerView.Adapter<CrimeHolder>
RecyclerView.ViewHolder
«interface» View.OnClickListener
implements
1
CrimeAdapter
+ CrimeAdapter(crimes: List<Crime>)
+ onCreateView(parent: ViewGroup, viewType: int): CrimeHolder
+ onBindViewHolder(holder: CrimeHolder, position: int): void
+ getItemCount(): int
list_item_crime.xml
inflates
- mCrimes
CrimeHolder
- mTitleTextView: TextView
- mDateTextView: TextView
- mSolvedCheckBox: CheckBox
+ CrimeHolder(itemView: View)
+ bindCrime(crime: Crime): void
+ onClick(v: View): void
- mCrime
CrimeLab
+ get(context: Context): CrimeLab
+ getCrimes(): List<Crime>
+ getCrime(id: UUID): Crime
1
- sCrimeLab
- mCrimes
0..*
- mCrime
1
Crime
- mId: UUID
- mTitle: String
- mDate: Date
- mSolved: boolean
+ setSolved(solved: boolean): void

The fragment maintains access to its host activity by defining a callback interface that the host activity must implement. ✓

The host activity either:

a) Invokes a new activity (in a one-pane layout), or
b) Swaps in a fragment (in a two-pane layout)

When a list entry is clicked, the corresponding details shall be displayed. ✓

Implementing the List’s Callback Interface: CrimeListActivity.onCrimeSelected

```
public class CrimeListActivity extends SingleFragmentActivity
        implements CrimeListFragment.Callbacks {

    @Override
    public void onCrimeSelected(Crime crime) {
        if (findViewById(R.id.detail_fragment_container) == null) {
            Intent intent = CrimeActivity.newIntent(this, crime.getId());
            startActivity(intent);
        } else {
            Fragment newDetail = CrimeFragment.newInstance(crime.getId());
            getSupportFragmentManager().beginTransaction()
                    .replace(R.id.detail_fragment_container, newDetail)
                    .commit();
        }
    }

    // ...
}
```

Check if the fragment container for the detail view exists (this is only the case if the two-pane layout was inflated earlier)

One-pane interface:
Use intent to start new activity and pass ID of selected item along

Two-pane interface:
Use fragment manager to replace fragment in detail container, and pass ID of selected item along

This is how an activity invokes another activity or fragment

Providing the Intent and Reacting to It (To Show Details in One-Pane Layout)

This is how data is passed to and received by an activity: through intent extras

```java
public class CrimeActivity extends SingleFragmentActivity {

    private static final String EXTRA_CRIME_ID =
            "com.bignerdranch.android.criminalintent.crime_id";

    public static Intent newIntent(Context packageContext, UUID crimeId) {
        Intent intent = new Intent(packageContext, CrimeActivity.class);
        intent.putExtra(EXTRA_CRIME_ID, crimeId);
        return intent;
    }

    @Override
    protected Fragment createFragment() {
        UUID crimeId = (UUID) getIntent().getSerializableExtra(EXTRA_CRIME_ID);
        return CrimeFragment.newInstance(crimeId);
    }
```

Typo-prevention constant

Static method to give CrimeActivity control of creating the intents that other activities shall send to it

Pass desired item ID along as parameter

Reminder: Called by SingleFragmentActivity.onCreate to determine the fragment to be associated with the activity

Create and return the fragment to be associated with the activity

Retrieve ID of item to display from intent

Creating the Detail Fragment for an Item (For Display in One- or Two Pane Layout)

This is how data is passed to and received by a fragment: through fragment arguments

```
public class CrimeFragment extends Fragment {

    private static final String ARG_CRIME_ID = "crime_id";
    private Crime mCrime;

    public static CrimeFragment newInstance(UUID crimeId) {
        Bundle args = new Bundle();
        args.putSerializable(ARG_CRIME_ID, crimeId);
        CrimeFragment fragment = new CrimeFragment();
        fragment.setArguments(args);
        return fragment;
    }

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        UUID crimeId = (UUID) getArguments().getSerializable(ARG_CRIME_ID);
        mCrime = CrimeLab.get(getActivity()).getCrime(crimeId);
    }

    // ...
}
```

Static factory method

Place ID of item to show in an arguments bundle for the new fragment

Retrieve ID of item to show from fragment arguments and store in member variable

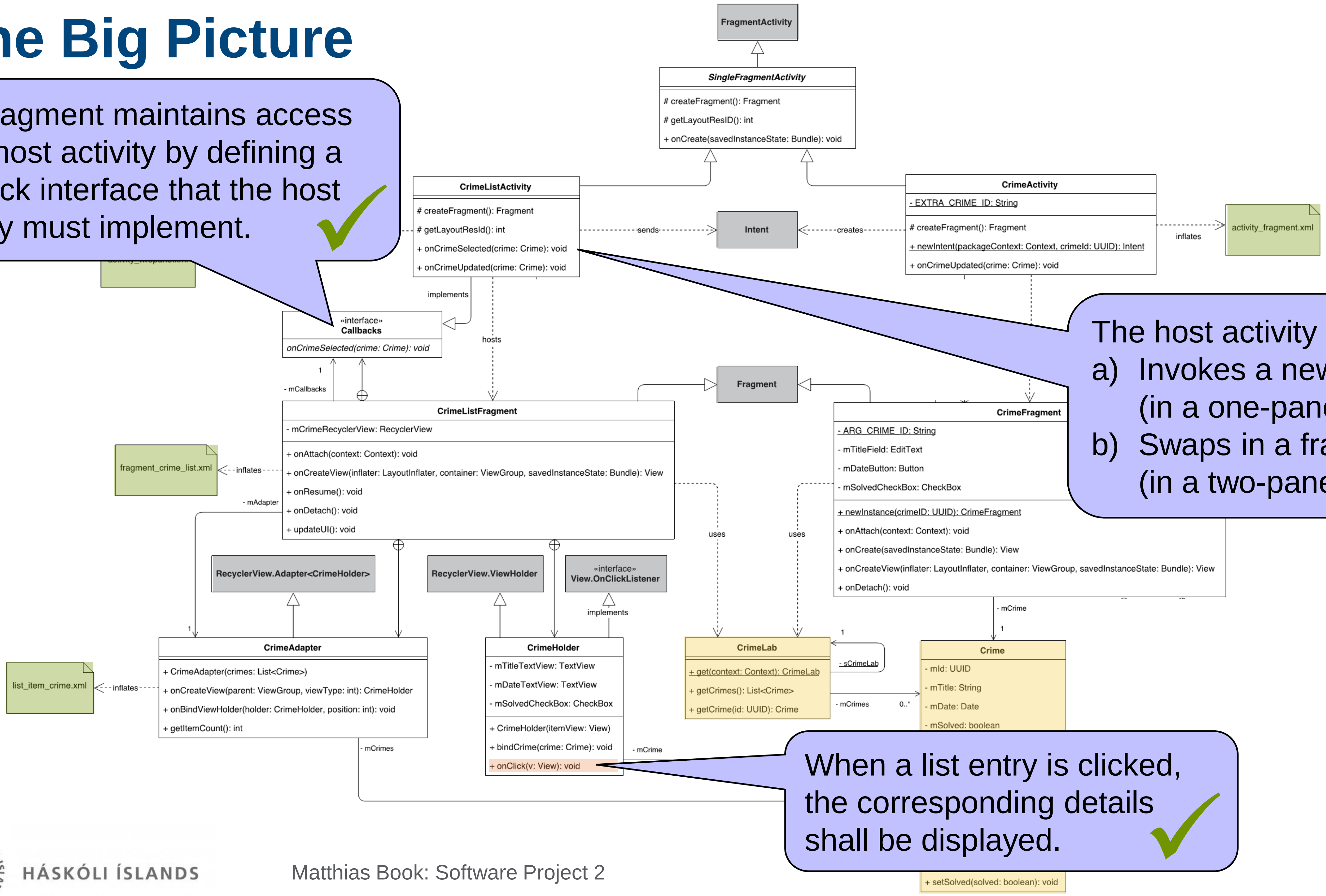
The Big Picture

The fragment maintains access to its host activity by defining a callback interface that the host activity must implement. ✓

The host activity either:

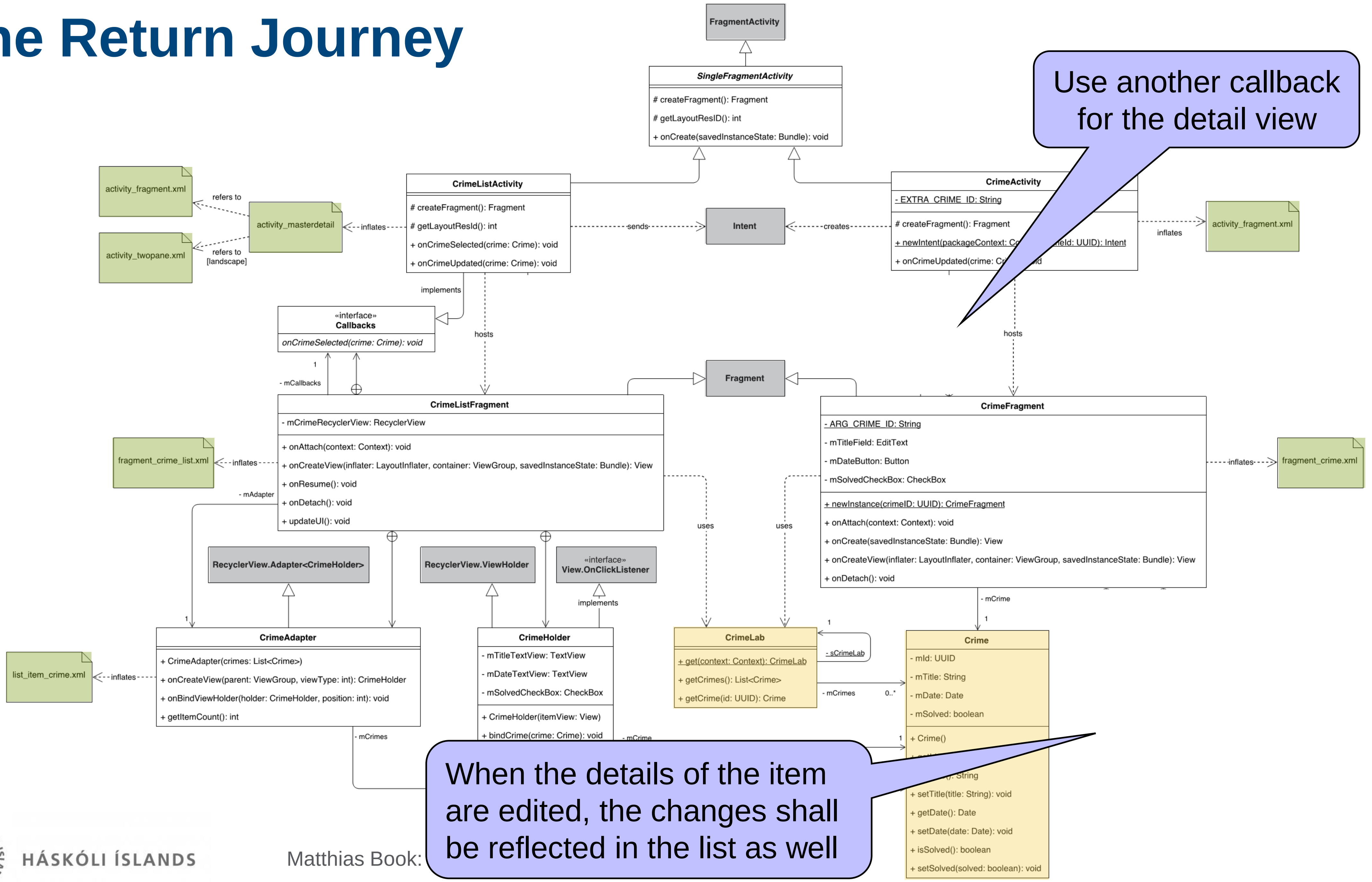
a) Invokes a new activity (in a one-pane layout), or
b) Swaps in a fragment (in a two-pane layout) ✓

When a list entry is clicked, the corresponding details shall be displayed. ✓



FragmentActivity
SingleFragmentActivity
# createFragment(): Fragment
# getLayoutResID(): int
+ onCreate(savedInstanceState: Bundle): void
CrimeListActivity
# createFragment(): Fragment
# getLayoutResId(): int
+ onCrimeSelected(crime: Crime): void
+ onCrimeUpdated(crime: Crime): void
sends
Intent
creates
CrimeActivity
- EXTRA_CRIME_ID: String
# createFragment(): Fragment
+ newIntent(packageContext: Context, crimeId: UUID): Intent
+ onCrimeUpdated(crime: Crime): void
inflates
activity_fragment.xml
implements
«interface» Callbacks
onCrimeSelected(crime: Crime): void
hosts
1
- mCallbacks
Fragment
CrimeListFragment
- mCrimeRecyclerView: RecyclerView
+ onAttach(context: Context): void
+ onCreateView(inflater: LayoutInflater, container: ViewGroup, savedInstanceState: Bundle): View
+ onResume(): void
+ onDetach(): void
+ updateUI(): void
fragment_crime_list.xml
inflates
- mAdapter
CrimeFragment
- ARG_CRIME_ID: String
- mTitleField: EditText
- mDateButton: Button
- mSolvedCheckBox: CheckBox
+ newInstance(crimeID: UUID): CrimeFragment
+ onAttach(context: Context): void
+ onCreate(savedInstanceState: Bundle): View
+ onCreateView(inflater: LayoutInflater, container: ViewGroup, savedInstanceState: Bundle): View
+ onDetach(): void
uses
uses
RecyclerView.Adapter<CrimeHolder>
RecyclerView.ViewHolder
«interface» View.OnClickListener
implements
1
CrimeAdapter
+ CrimeAdapter(crimes: List<Crime>)
+ onCreateView(parent: ViewGroup, viewType: int): CrimeHolder
+ onBindViewHolder(holder: CrimeHolder, position: int): void
+ getItemCount(): int
list_item_crime.xml
inflates
- mCrimes
CrimeHolder
- mTitleTextView: TextView
- mDateTextView: TextView
- mSolvedCheckBox: CheckBox
+ CrimeHolder(itemView: View)
+ bindCrime(crime: Crime): void
+ onClick(v: View): void
- mCrime
CrimeLab
+ get(context: Context): CrimeLab
+ getCrimes(): List<Crime>
+ getCrime(id: UUID): Crime
1
- sCrimeLab
- mCrimes
0..*
- mCrime
1
Crime
- mId: UUID
- mTitle: String
- mDate: Date
- mSolved: boolean
+ setSolved(solved: boolean): void

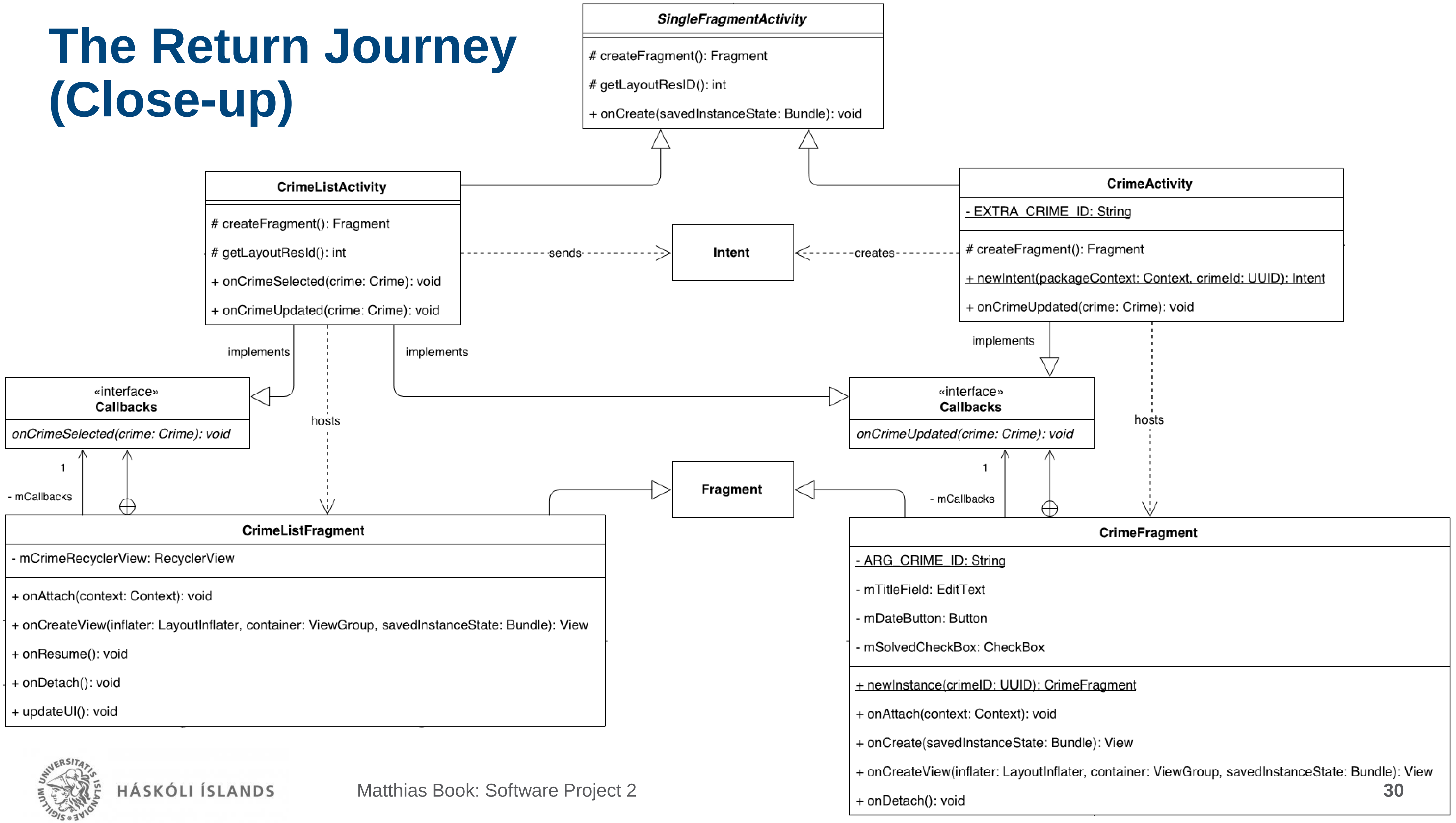
The Return Journey


FragmentActivity
SingleFragmentActivity
# createFragment(): Fragment
# getLayoutResID(): int
+ onCreate(savedInstanceState: Bundle): void
activity_fragment.xml
refers to
activity_masterdetail
activity_twopane.xml
refers to [landscape]
inflates
CrimeListActivity
# createFragment(): Fragment
# getLayoutResId(): int
+ onCrimeSelected(crime: Crime): void
+ onCrimeUpdated(crime: Crime): void
sends
Intent
creates
CrimeActivity
- EXTRA_CRIME_ID: String
# createFragment(): Fragment
+ newIntent(packageContext: C... ...Id: UUID): Intent
+ onCrimeUpdated(crime: Cr... ...d
inflates
activity_fragment.xml
implements
«interface» Callbacks
onCrimeSelected(crime: Crime): void
hosts
hosts
1
- mCallbacks
Fragment
CrimeListFragment
- mCrimeRecyclerView: RecyclerView
+ onAttach(context: Context): void
+ onCreateView(inflater: LayoutInflater, container: ViewGroup, savedInstanceState: Bundle): View
+ onResume(): void
+ onDetach(): void
+ updateUI(): void
fragment_crime_list.xml
inflates
- mAdapter
CrimeFragment
- ARG_CRIME_ID: String
- mTitleField: EditText
- mDateButton: Button
- mSolvedCheckBox: CheckBox
+ newInstance(crimeID: UUID): CrimeFragment
+ onAttach(context: Context): void
+ onCreate(savedInstanceState: Bundle): View
+ onCreateView(inflater: LayoutInflater, container: ViewGroup, savedInstanceState: Bundle): View
+ onDetach(): void
inflates
fragment_crime.xml
uses
uses
RecyclerView.Adapter<CrimeHolder>
RecyclerView.ViewHolder
«interface» View.OnClickListener
implements
1
CrimeAdapter
+ CrimeAdapter(crimes: List<Crime>)
+ onCreateView(parent: ViewGroup, viewType: int): CrimeHolder
+ onBindViewHolder(holder: CrimeHolder, position: int): void
+ getItemCount(): int
list_item_crime.xml
inflates
- mCrimes
CrimeHolder
- mTitleTextView: TextView
- mDateTextView: TextView
- mSolvedCheckBox: CheckBox
+ CrimeHolder(itemView: View)
+ bindCrime(crime: Crime): void
- mCrime
CrimeLab
1
- sCrimeLab
+ get(context: Context): CrimeLab
+ getCrimes(): List<Crime>
+ getCrime(id: UUID): Crime
- mCrimes
0..*
- mCrime
1
Crime
- mId: UUID
- mTitle: String
- mDate: Date
- mSolved: boolean
+ Crime()
1
...(): String
+ setTitle(title: String): void
+ getDate(): Date
+ setDate(date: Date): void
+ isSolved(): boolean
+ setSolved(solved: boolean): void

Use another callback for the detail view

When the details of the item are edited, the changes shall be reflected in the list as well

The Return Journey (Close-up)


SingleFragmentActivity
# createFragment(): Fragment
# getLayoutResID(): int
+ onCreate(savedInstanceState: Bundle): void
CrimeListActivity
# createFragment(): Fragment
# getLayoutResId(): int
+ onCrimeSelected(crime: Crime): void
+ onCrimeUpdated(crime: Crime): void
sends
Intent
creates
CrimeActivity
- EXTRA_CRIME_ID: String
# createFragment(): Fragment
+ newIntent(packageContext: Context, crimeId: UUID): Intent
+ onCrimeUpdated(crime: Crime): void
implements
implements
implements
hosts
hosts
«interface»
Callbacks
onCrimeSelected(crime: Crime): void
«interface»
Callbacks
onCrimeUpdated(crime: Crime): void
1
- mCallbacks
1
- mCallbacks
Fragment
CrimeListFragment
- mCrimeRecyclerView: RecyclerView
+ onAttach(context: Context): void
+ onCreateView(inflater: LayoutInflater, container: ViewGroup, savedInstanceState: Bundle): View
+ onResume(): void
+ onDetach(): void
+ updateUI(): void
CrimeFragment
- ARG_CRIME_ID: String
- mTitleField: EditText
- mDateButton: Button
- mSolvedCheckBox: CheckBox
+ newInstance(crimeID: UUID): CrimeFragment
+ onAttach(context: Context): void
+ onCreate(savedInstanceState: Bundle): View
+ onCreateView(inflater: LayoutInflater, container: ViewGroup, savedInstanceState: Bundle): View
+ onDetach(): void

Declaring the Detail's Callback Interface: CrimeFragment.Callbacks

Reference to host activity

Declaration of interface that the host activity must implement to react to updates of detail data

Automatically stores a reference to the host activity when the fragment is associated with it.

- In one-pane layout, this is CrimeActivity
- In two-pane layout, this is CrimeListActivity

```
public class CrimeFragment extends Fragment {

    private Callbacks mCallbacks;

    public interface Callbacks {
        void onCrimeUpdated(Crime crime);
    }

    @Override
    public void onAttach(Context context) {
        super.onAttach(context);
        mCallbacks = (Callbacks) context;
    }

    @Override
    public void onDetach() {
        super.onDetach();
        mCallbacks = null;
    }

    // ...
}
```

Invoking Callback in CrimeFragment's Event Listeners

```
public class CrimeFragment extends Fragment {
    private Crime mCrime;
    private CheckBox mSolvedCheckBox;
    private Callbacks mCallbacks;
    // ...

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
            Bundle savedInstanceState) {
        View v = inflater.inflate(R.layout.fragment_crime, container, false);
        // ...
        mSolvedCheckBox = (CheckBox) v.findViewById(R.id.crime_solved);
        mSolvedCheckBox.setChecked(mCrime.isSolved());
        mSolvedCheckBox.setOnCheckedChangeListener(
                new CompoundButton.OnCheckedChangeListener() {
            @Override
            public void onCheckedChanged(CompoundButton buttonView, boolean isChecked) {
                mCrime.setSolved(isChecked);
                mCallbacks.onCrimeUpdated(mCrime);
            }
        });
        // ...
    }
    // ...
}
```

This is how a fragment sends messages to another activity…

Example: The checkbox' event handler

Update the model

Notify list activity (via callback reference) of update

Implementing the Callback Interface: Crime[List]Activity.onCrimeUpdated

…which can then trigger logic in the fragment it hosts

```
public class CrimeListActivity extends SingleFragmentActivity
        implements CrimeListFragment.Callbacks, CrimeFragment.Callbacks {

    @Override
    public void onCrimeUpdated(Crime crime) {
        CrimeListFragment listFragment = (CrimeListFragment)
            getSupportFragmentManager().findFragmentById(R.id.fragment_container);
        listFragment.updateUI();
    }

    // ...
}
```

…and update its UI

In two-pane layout, retrieve the list fragment this activity is hosting…

```
public class CrimeActivity extends SingleFragmentActivity
        implements CrimeFragment.Callbacks {
    // ...
    @Override
    public void onCrimeUpdated(Crime crime) { }
}
```

In one-pane layout, the hosting fragment is CrimeActivity, which does not have a list to update – so this method implementation remains empty.

Ensuring the List is Updated in One-Pane Layout

```
public class CrimeListFragment extends Fragment {

    private CrimeAdapter mAdapter;

    @Override
    public void onResume() {
        super.onResume();
        updateUI();
    }

    public void updateUI() {
        CrimeLab crimeLab = CrimeLab.get(getActivity());
        List<Crime> crimes = crimeLab.getCrimes();

        if (mAdapter == null) {
            mAdapter = new CrimeAdapter(crimes);
            mCrimeRecyclerView.setAdapter(mAdapter);
        } else {
            mAdapter.notifyDataSetChanged();
        }
    }

    // ...
}
```

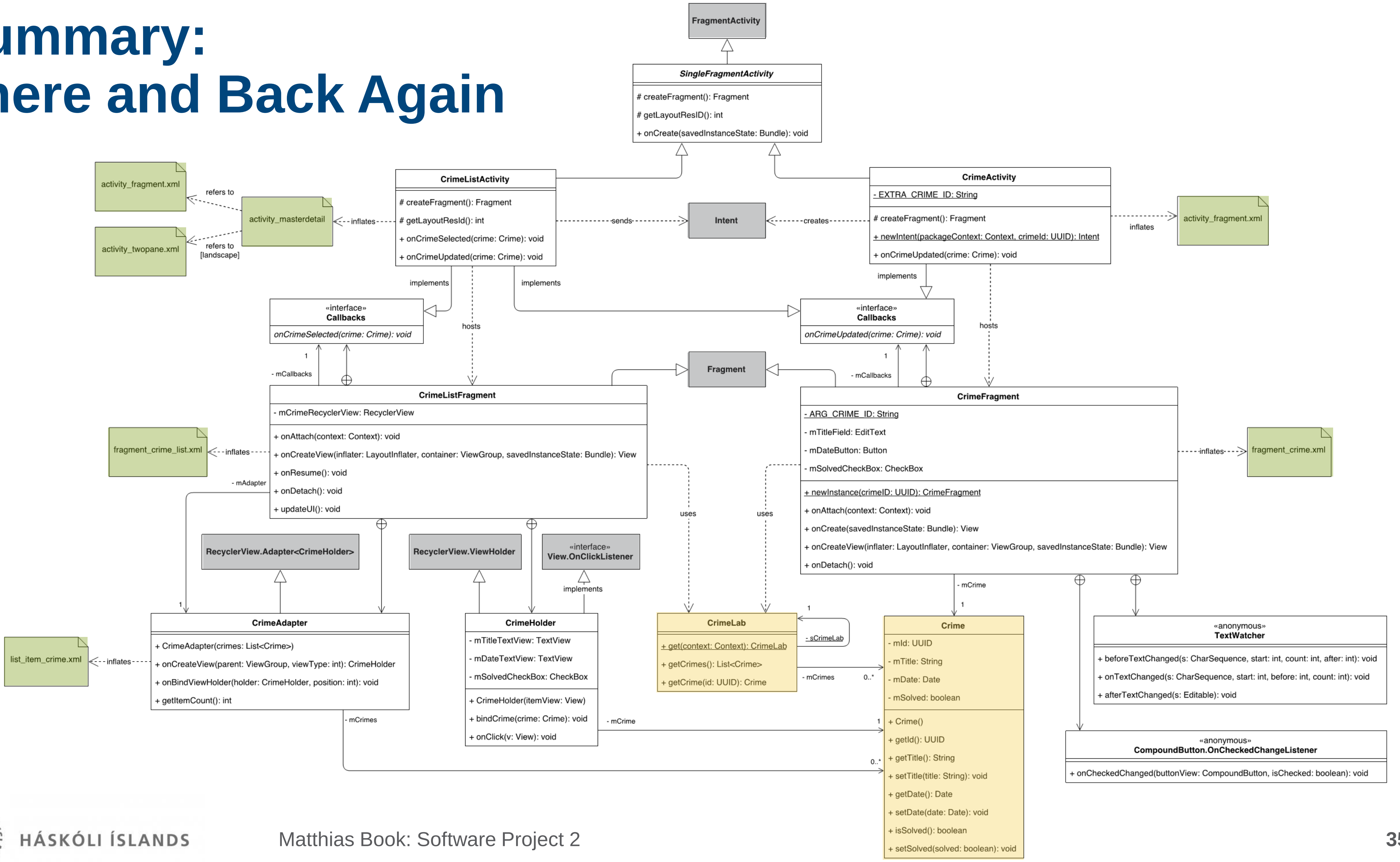
In one-pane layout, update the list UI when the list activity (and thus the list fragment) comes to the front again (i.e. resumes) after the user backed out of the detail view

To update the UI in one- and two-pane layout, retrieve current list of crimes from the model…

…and update the scrollable list with it (see Android textbook, Ch. 8 for details)

Caution – inefficient solution (chosen here for clarity):
It would not be necessary to retrieve all list items again since just one item has been changed!

Summary: There and Back Again


FragmentActivity
SingleFragmentActivity
# createFragment(): Fragment
# getLayoutResID(): int
+ onCreate(savedInstanceState: Bundle): void
CrimeListActivity
# createFragment(): Fragment
# getLayoutResId(): int
+ onCrimeSelected(crime: Crime): void
+ onCrimeUpdated(crime: Crime): void
CrimeActivity
- EXTRA_CRIME_ID: String
# createFragment(): Fragment
+ newIntent(packageContext: Context, crimeId: UUID): Intent
+ onCrimeUpdated(crime: Crime): void
activity_fragment.xml
refers to
activity_masterdetail
inflates
activity_twopane.xml
refers to [landscape]
sends
Intent
creates
activity_fragment.xml
inflates
implements
implements
implements
«interface» Callbacks
onCrimeSelected(crime: Crime): void
hosts
«interface» Callbacks
onCrimeUpdated(crime: Crime): void
hosts
1
- mCallbacks
Fragment
1
- mCallbacks
CrimeListFragment
- mCrimeRecyclerView: RecyclerView
+ onAttach(context: Context): void
+ onCreateView(inflater: LayoutInflater, container: ViewGroup, savedInstanceState: Bundle): View
+ onResume(): void
+ onDetach(): void
+ updateUI(): void
fragment_crime_list.xml
inflates
- mAdapter
CrimeFragment
- ARG_CRIME_ID: String
- mTitleField: EditText
- mDateButton: Button
- mSolvedCheckBox: CheckBox
+ newInstance(crimeID: UUID): CrimeFragment
+ onAttach(context: Context): void
+ onCreate(savedInstanceState: Bundle): View
+ onCreateView(inflater: LayoutInflater, container: ViewGroup, savedInstanceState: Bundle): View
+ onDetach(): void
inflates
fragment_crime.xml
uses
uses
RecyclerView.Adapter<CrimeHolder>
RecyclerView.ViewHolder
«interface» View.OnClickListener
implements
- mCrime
1
1
CrimeAdapter
+ CrimeAdapter(crimes: List<Crime>)
+ onCreateView(parent: ViewGroup, viewType: int): CrimeHolder
+ onBindViewHolder(holder: CrimeHolder, position: int): void
+ getItemCount(): int
list_item_crime.xml
inflates
CrimeHolder
- mTitleTextView: TextView
- mDateTextView: TextView
- mSolvedCheckBox: CheckBox
+ CrimeHolder(itemView: View)
+ bindCrime(crime: Crime): void
+ onClick(v: View): void
CrimeLab
+ get(context: Context): CrimeLab
+ getCrimes(): List<Crime>
+ getCrime(id: UUID): Crime
1
- sCrimeLab
- mCrimes
0..*
Crime
- mId: UUID
- mTitle: String
- mDate: Date
- mSolved: boolean
+ Crime()
+ getId(): UUID
+ getTitle(): String
+ setTitle(title: String): void
+ getDate(): Date
+ setDate(date: Date): void
+ isSolved(): boolean
+ setSolved(solved: boolean): void
- mCrimes
- mCrime
1
0..*
«anonymous» TextWatcher
+ beforeTextChanged(s: CharSequence, start: int, count: int, after: int): void
+ onTextChanged(s: CharSequence, start: int, before: int, count: int): void
+ afterTextChanged(s: Editable): void
«anonymous» CompoundButton.OnCheckedChangeListener
+ onCheckedChanged(buttonView: CompoundButton, isChecked: boolean): void

HÁSKÓLI ÍSLANDS

In-Class Quiz 7: Activity & Fragment Communication

Note the numbers of the matching sentence parts:

a) An activity can start another activity…

b) An activity can start a fragment…

c) A fragment can start an activity…

d) An activity can send a message to a fragment it hosts…

e) A fragment can send a message to the activity hosting it…

f) A fragment can send a message to another fragment…

1. …by calling a method of a callback interface defined by the fragment and implemented by the activity.
2. …by calling a method of the fragment via the reference the activity holds to it.
3. …by creating an intent and passing it to the ActivityManager, who can then start the desired activity.
4. …by sending a message to the fragment's host activity, who can then start the desired activity.
5. …by sending a message to the fragment's host activity, who can then send a message to the other fragment.
6. …by telling the FragmentManager to start the desired fragment in a container in the activity's layout.