Fragments

Появились в **Android 3.0 (API level 11)**

Fragment - кусочек пользовательского интерфейса или поведения, которое можно поместить в *Activity*. Они были придуманы для того чтобы можно было заполнять большим количеством информации большие экраны планшетов.

Фрагмент существует только в контексте *Activity*. Жизненные циклы *Fragment* и *Activity* очень связаны: если *Activity* остановлена, то *Fragment* внутри не может быть в состоянии started. Когда уничтожается *Activity*, то и все *Fragment* будут уничтожены. Не прописываются в манифесте!

Чтобы создать *Fragment* необходимо отекстендить класс `Fragment`. Далее мы рассмотрим методы добавления фрагмента в *Activity.*

Первый способ добавления фрагмента

Первый способ добавления фрагмента в *Activity* - ***через XML***. Для того, чтобы при изменении конфигурации *Activity* не потеряла свои фрагменты в *xml* в лейауте фрагмента необходимо указать *id*, *tag* или *id* лейаута, в который помещен фрагмент. Такие фрагменты не могут быть удалены, только скрыты. Если вам необходимо менять удалять и или добавлять фрагменты во время работы приложения, тогда необходимо использовать второй способ.

```
<fragment android:name="com.example.news.ArticlesFragment"
        android:id="@+id/articles"
        android:layout_weight="2"
        android:layout_width="0dp"
        android:layout_height="match_parent" />
```

Для того, чтобы найти находящийся в активити фрагмент нужно использовать метод - ***findFragmentById()***.

Этот способ добавления фрагмента используется очень редко.

Второй способ добавления фрагмента

Второй способ добавления фрагмента - программно. Для работы с фрагментами используется класс ***FragmentManager***. В любой момент, пока активити активна можно добавить фрагменты нужно только указать в какой лейаут (его id), содержащийся в разметке активити.

Код для добавления фрагмента:

```
FragmentManager fragmentManager = getFragmentManager()
FragmentTransaction fragmentTransaction = fragmentManager.beginTransaction();
ExampleFragment fragment = new ExampleFragment();
fragmentTransaction.add(R.id.fragment_container, fragment);
fragmentTransaction.commit();
```

Для того, чтобы найти добавленный в активити фрагмент нужно использовать метод ***findFragmentByTag()***.

Удалить фрагменты из стека - ***popBackStack()***. Можно слушать изменения в бек стеке с помощью слушателя ***addOnBackStackChangedListener()***.

Методы для работы с фрагментами

Нельзя применить remove(), replace(), detach(), или attach() к “статическим” фрагментам добавленным в лейаут активити!

Добавление:

add(int containerViewId, Fragment fragment)
add(int containerViewId, Fragment fragment, String tag)

attach(Fragment fragment) - Присоединить фрагмент к ui, который был от соединён с помощью метода detach(Fragment). При этом ui фрагмента будет пересоздан, подсоединен и показан.

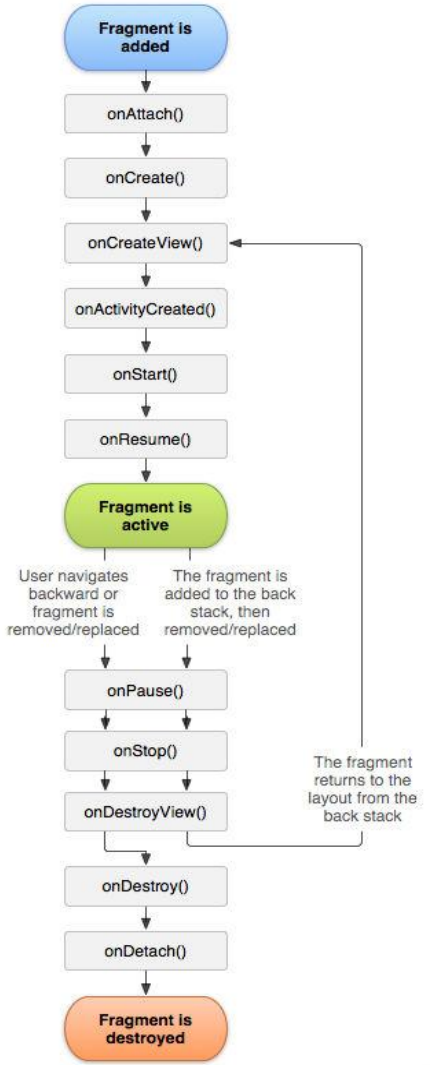
detach(Fragment fragment) - От соединяет выбранный фрагмент от ui. При этом фрагмент находится в таком же состоянии как и фрагмент добавленный в back stack: фрагмент удален из ui, но его состояние управляется fragment manager. Если фрагмент переходит в это состояние, то его ui будет уничтожен.

hide(Fragment fragment) - Прячет фрагмент, добавленный в лейаут активити.

remove(Fragment fragment) - Удаляет существующий фрагмент.

replace(int containerViewId, Fragment fragment, String tag) - Заменяет добавленный ранее фрагмент. Удаление всех существующих фрагментов в этом лейауте и добавление туда нового.

show(Fragment fragment) - Показывает ранее скрытый фрагмент, который добавлен в лейаут.



Жизненный цикл фрагмента

onAttach() - Вызывается когда фрагмент связывается (асоциируется, присоединяется на этапе создания) с *Activity*. В этот метод передаётся *Activity,* но её UI может быть ещё не создан.

onCreateView() - Вызывается для создания UI фрагмента.

onActivityCreated() - Вызывается после завершения метода ***onCreate()*** у *Activity*.

onDestroyView() - Вызывается при удалении UI фрагмента.

onDetach() - Вызывается, когда фрагмент от соединяется от *Activity*, которой принадлежал.

Жизненный цикл фрагмента относительно независим только когда *Activity* находится в состоянии resumed.

ВЫЗЫВАЕМ МЕТОДЫ СУПЕРКЛАССА В ЭТИХ МЕТОДАХ КРОМЕ ***onCreateView()***

Настройка вида фрагмента

В методе onCreateView система спрашивает у нас, что ей отображать внутри фрагмента. Мы сообщаем системе, что хотим видеть во фрагменте содержимое соответствующего layout-файла. Для этого мы сами создаем View с помощью inflater и отдаем его системе. Т.е. по смыслу это аналог метода setContentView, который мы вызываем в Activity. Только здесь нам приходится самим создавать View, а не просто передавать идентификатор layout-файла.

```
@Override
public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) {
    mView = inflater.inflate(getLayoutId(), container, false);
    return mView;
}
```

Нахождение вью из лейаута выглядит так:

```
mEmptyView = mView.findViewById(R.id.empty);
```

Связь активити и фрагмента

Получить ссылку на активити из фрагмента можно с помощью getActivity(). Также можно получить виджеты из разметки активити, но только после того как активити пройдет свой метод onCreate()

```
View listView = getActivity().findViewById(R.id.list);
```

из активити ссылка на фрагмент:

```
ExampleFragment fragment = (ExampleFragment)
getFragmentManager().findFragmentById(R.id.example_fragment);
((TextView) frag2.getView().findViewById(R.id.textView))
```

Опять же, обращаться к вью фрагмента необходимо после того как фрагмент пройдет свой метод onCreateView()

Передача данных во фрагмент

Передача данных из активити во фрагмент происходит с помощью Bundle. Нельзя передавать данные в конструктор фрагмента! newInstance(args) - хороший паттерн для запуска фрагмента.

Код в активити:

```
changeFragment(ExampleFragment.newInstance(arg));
```

Код во фрагменте ExampleFragment:

```
public static ExampleFragment newInstance(int arg1) {
        ExampleFragment fragment = new ExampleFragment();
        Bundle args = new Bundle();
        args.putInt(“key”, arg1);
        fragment.setArguments(args);
  return fragment;
}
```

получить можно в методе onCreate(), onCreateView(), onActivityCreated():
getArguments().getInt(“key”);

Совместимость

Если Activity наследует ***android.app.Activity***, то фрагменты должны наследовать ***android.app.Fragment***. В версиях Андроид 3.0 и выше можно использовать эти классы.

Если в проекте используются библиотеки совместимости, то следует использовать классы

android.support.v4.app.Fragment а активити должна экстендить класс ***android.support.v4.app.FragmentActivity***

Так как мы всегда используем библиотеку AppCompat, класс AppCompatActivity (также как и предыдущая версия ActionBarActivity) уже экстендит ***FragmentActivity*** и может работать с фрагментами.

Работа с меню во фрагменте

При создании фрагмента в методе onCreateView() следует указать, что фрагмент будет добавлять айтемы меню с помощью метода setHasOptionsMenu()

Для создания и настройки меню во фрагменте есть подобные активити методы:

```
@Override
public void onCreateOptionsMenu Menu menu, MenuInflater inflater) {
      inflater.inflate(R.menu.menu_sample, menu);
}
```

и

```
@Override
public void onPrepareOptionsMenu Menu menu) {
      super.onPrepareOptionsMenu(menu);
      getActivity().invalidateOptionsMenu();
      MenuItem filter = menu.findItem(R.id.section);
      filter.setVisible false);
}
```

при переопределении и написании метода onOptionsItemSelected() нужно учитывать что сначала вызовется соответствующий метод у активити, и если пункт меню не будет найден, то вызовется метод фрагмента.