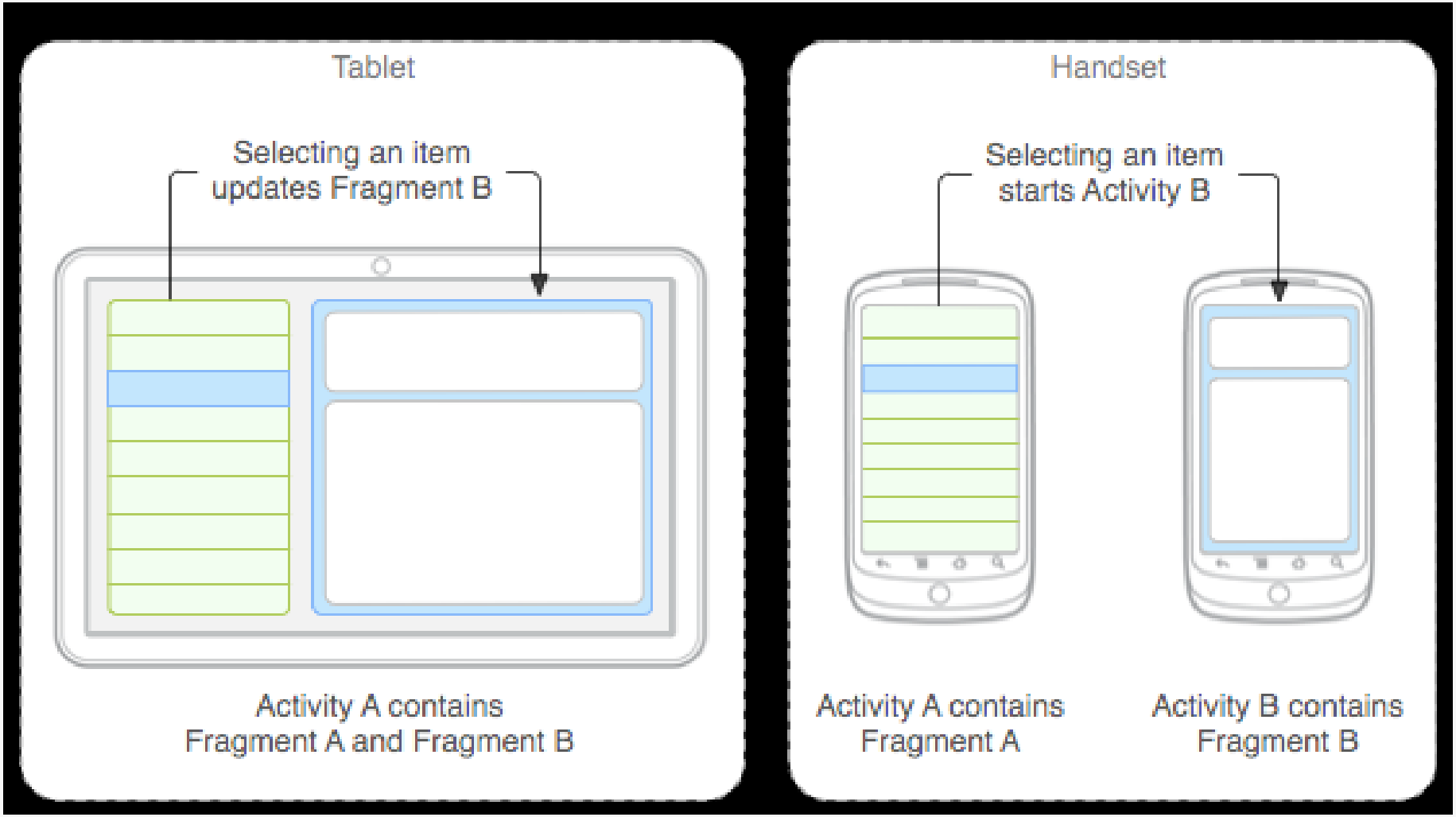
Breminale2012 unter der Oberfläche

Björn Ahlfeld

- Live Demo!
- Fragments und Activity Callbacks
- Loaders

Live Demo!

ListFragments & Activity Callbacks



ListFragments & Activity Callbacks

- Reagieren auf Item Klicks
  - Interface im Fragment anlegen
  - public void onAttach (Activity activity)
    - Wird aufgerufen sobald dass Fragment an die Activity angefügt wurde
    - Prüfen ob die Activity das Interface des Fragments implmentiert

Beispielcode des ListFragments

```
public interface OnArticleSelectedListener {
        public void onArticleSelected(Uri articleUri);
}

@Override
public void onAttach(Activity activity) {
     super.onAttach(activity);
        try {
            mListener = (OnArticleSelectedListener) activity;
        } catch (ClassCastException e) {
            throw new ClassCastException(activity.toString() + "
must implement
OnArticleSelectedListener");
        }
}
```

Beispielcode DetailsFragment

```
public finall class DetailsFragment extends Fragment{
    public static final String URL = „url“;

    […]

    public DetailsFragment newInstance(Url url){
        DetailsFragment f = new DetailsFragment();
        Bundle args = new Bundle();
        args.putString(URL, url.toString());
        f.setArguments(args);
        return f;
    }  [...]
}
```

Beispielcode der Activity

```
public final class MyActivity extends FragmentActivity implements
OnArticleSelectedListener {
    [...]

    @Override
    public void onArticleSelected(Uri articleUri){
        //Check if DetailsFragment is in the layout
                DetailFragment details =
getSupportFragmentManager().findFragmentById(R.id.detailsFr
agment);
            if(details == null){
                    //Start new activity via Intent
            } else { //Create new instance of detailfagment }
    }  [...]
}
```

Beispielcode der DetailsActivity

```
public final class DetailsActivity extends FragmentActivity{
    […]
    @Override onCreate(Bundle savedInstanceState){
        if(sa,vedInstanceState == null){
            DetailsFragment details = new DetailFragment();
            Details.setArguments(getIntent().getExtras());

getSupportFragmentManager().beginTransaction().add(R.id.conte
nt,
details).commit();
        }
    }
}
```

Loaders

- Vorteile von Loadern
  - Verrichten ihre Aufgabe im Hintergrund
    - → Große Anfragen blockieren nicht den UI-Thread
    - managedQuery(Uri uri, String[] projection, String selection, String[] selectionArgs, String sortOrder) vs. Loader
    - → manageQuery([...]) ist deprecated!!!
  - CursorLoader updatet sich automatisch
    - Registriert einen ContentObserver
    - Ruft forceLoad() auf, wenn sich Daten verändern

Loaders

- Werden durch den LoaderManager verwaltet
- Sind in Activities und Fragments verfügbar
  - Ausnahme Android-Version < 3 dann nur in der FragmentActivity und Fragments verfügbar
- Interface LoaderManager.CallBack muss implementiert werden

Beispielcode

```
public final class MyFragment extends ListFragment implements
LoaderCallbacks<Cursor> {

    [...]
    @Override
    public Loader<Cursor> onCreateLoader(int arg0, Bundle arg1){
        Final Uri baseUri  = Events.Content_URI;
        String selection = „“;
        return new CursorLoader(getActivity().getApplicationContext(),
baseUri,                PROJECTION, selection, null, null );
    }
    @Override
    public void onLoadFinished(Loader<Cursor> loader, Cursor c){
        mListView.swapCursor(c);
    }

    @Override
    public void onLoaderReset(Loader<Cursor> loader) { }
}
```