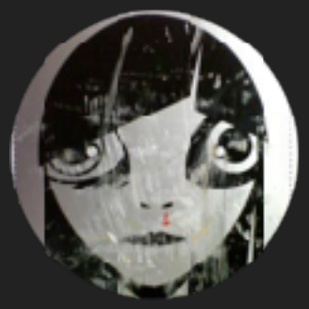
Advanced Shared Element Transition

Ryota Niinomi

AWA



Niinomi　Ryota

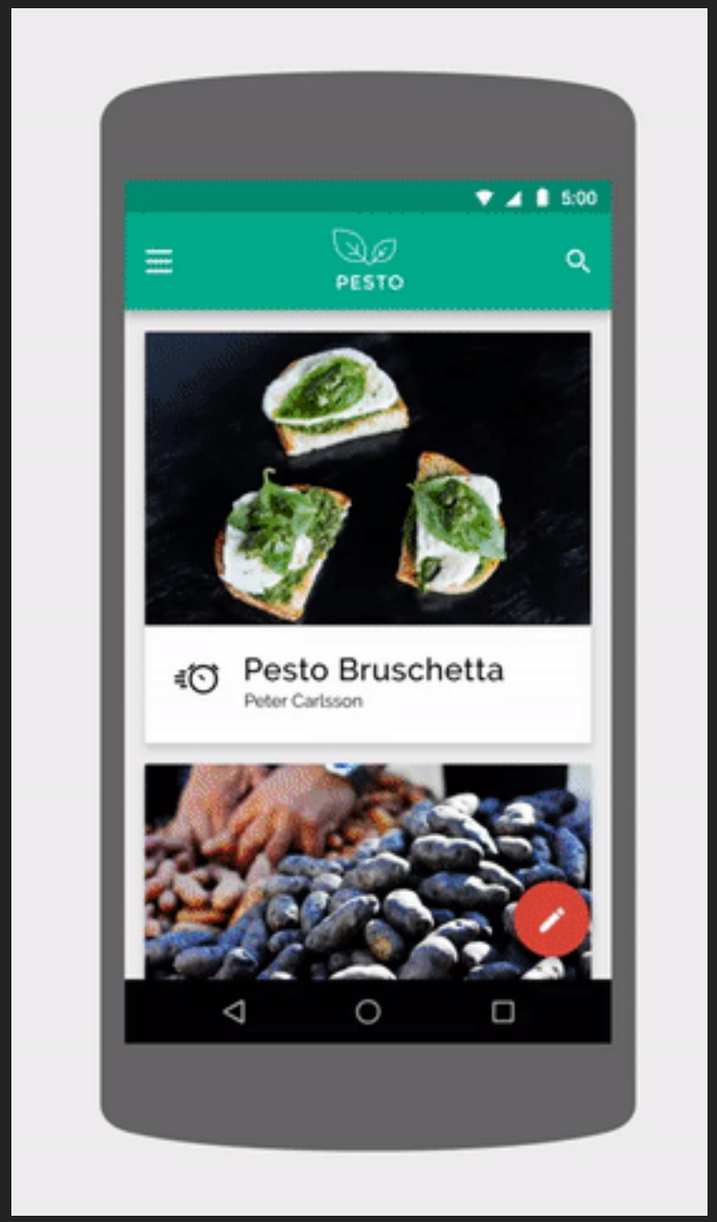
新家　亮太

r21nomi　@r21nomi

2012年　CA入社
コミュニティサービスのフロントエンド開発

2014年〜　AWAでAndroid開発

Shared Element Transition?



・２画面間のギャップをなくす

・視線を誘導する

・読み込み処理から意識をそらす

・心地よさ

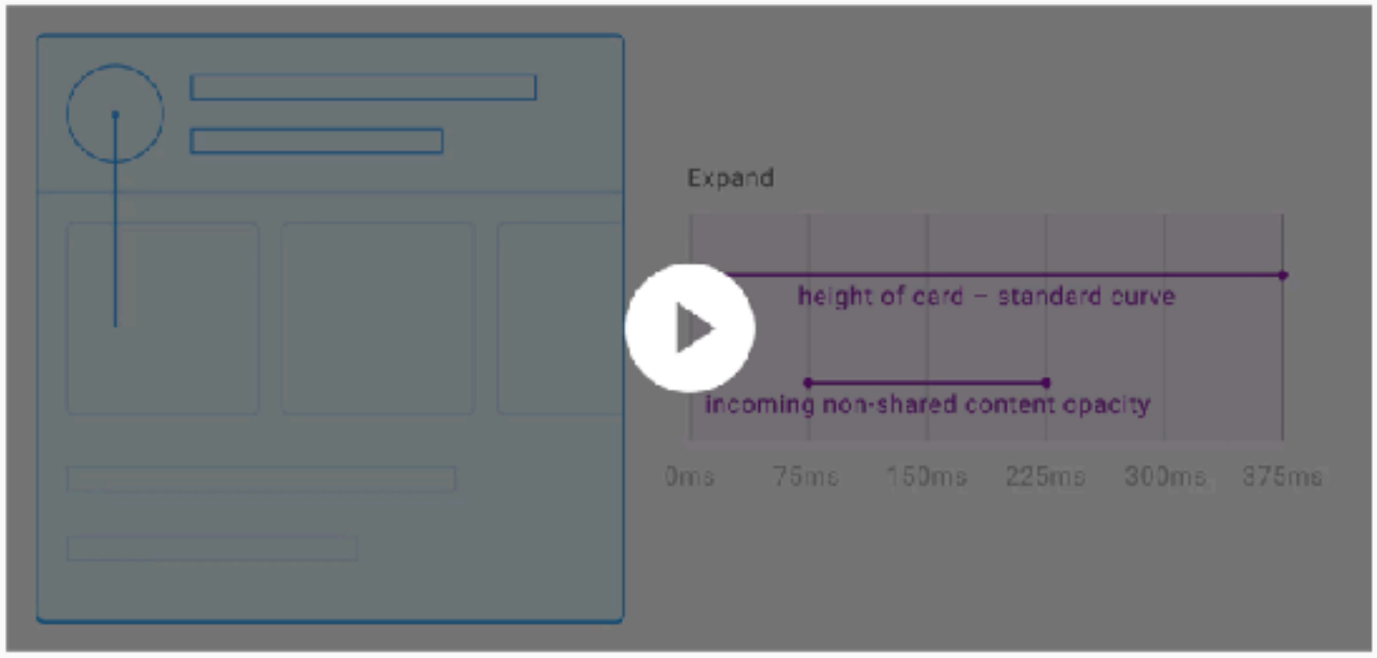
Continuity

Maintain a clear focal point during transitions by carefully selecting the number and type of elements shared across the transitions.

All content elements are shared

While a surface is expanding, a significant number of elements should remain visible during the transition.

Complex transitions should keep a single element visible (see below).



All elements from this collapsed card form the header of the expanded card.

https://material.io/guidelines/motion/choreography.html#choreography-continuity

Activity Transitions

Activity Transitions

- Shared Element Transitionを実現するAPI
- API Level 21〜

Basic Implementation

MainActivity -> DetailActivity

MainActivity.java

```
Intent intent = new Intent(this, DetailActivity.class);
Bundle options = ActivityOptionsCompat.makeSceneTransitionAnimation(
    context,
    sharedElementView,
    “shared_element_view”
).toBundle();
startActivity(intent, options);
```

DetailActivity.java

```
@Override
protected void onCreate(Bundle savedInstanceState){
    ...
    View view = findViewById(R.id.animationView);
    ViewCompat.setTransitionName(view, "shared_element_view");
}
```

Activity Transitions Using Adapter

Shared Elementの対象がAdapter内のView

```
postponeEnterTransition();  // 待機

startPostponedEnterTransition();  // 再開
```

DetailActivity.java

```
private ItemDetailAdapter.Listener mListener = new ItemDetailAdapter.Listener() {
    @Override
    public void onSharedElementViewPrepared() {
        startPostponedEnterTransition();
    }
};
```

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    ...
    postponeEnterTransition();

    adapter.setListner(mListener);
}
```

DetailActivity.java

```
private ItemDetailAdapter.Listener mListener = new ItemDetailAdapter.Listener() {
    @Override
    public void onSharedElementViewPrepared() {
        startPostponedEnterTransition();
    }
};
```

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    ...
    postponeEnterTransition();

    adapter.setListner(mListener);
}
```

ItemAdapter.java

```java
@Override
public RecyclerView.ViewHolder onCreateViewHolder(ViewGroup parent, int viewType) {
    View view = LayoutInflater.from(parent.getContext())
            .inflate(R.layout.item_viewholder, parent, false);
    final ViewHolder vh = new ViewHolder(view);

    ViewCompat.setTransitionName(vh.thumb, "shared_element_view");
    vh.thumb.getViewTreeObserver()
            .addOnPreDrawListener(new ViewTreeObserver.OnPreDrawListener() {
                @Override
                public boolean onPreDraw() {
                    vh.thumb.getViewTreeObserver().removeOnPreDrawListener(this);
                    mListener.onSharedElementViewPrepared();
                    return true;
                }
            });
    return vh;
}
```

ItemAdapter.java

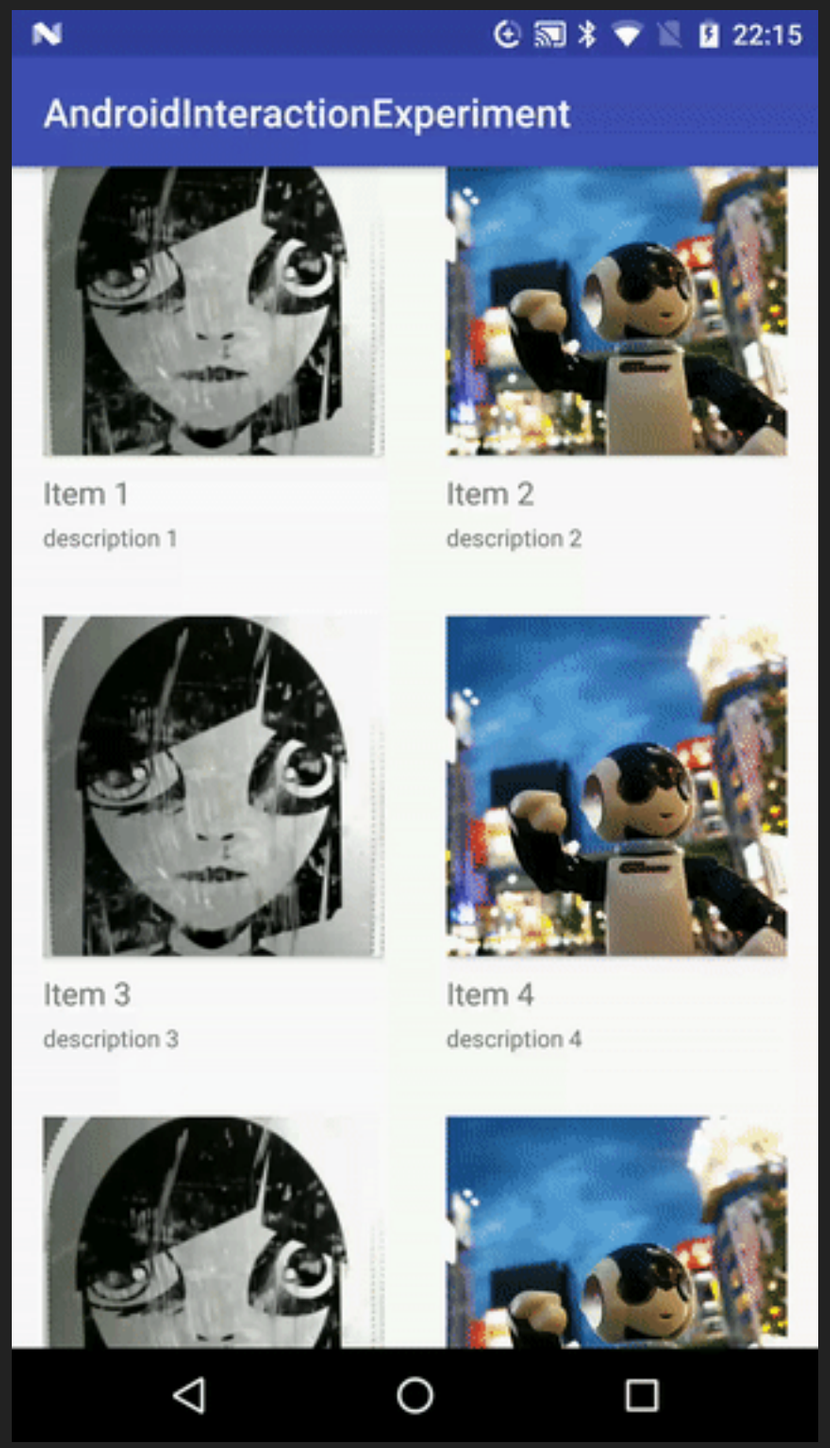
```java
@Override
public RecyclerView.ViewHolder onCreateViewHolder(ViewGroup parent, int viewType) {
    View view = LayoutInflater.from(parent.getContext())
            .inflate(R.layout.item_viewholder, parent, false);
    final ViewHolder vh = new ViewHolder(view);

    ViewCompat.setTransitionName(vh.thumb, "shared_element_view");
    vh.thumb.getViewTreeObserver()
            .addOnPreDrawListener(new ViewTreeObserver.OnPreDrawListener() {
                @Override
                public boolean onPreDraw() {
                    vh.thumb.getViewTreeObserver().removeOnPreDrawListener(this);
                    mListener.onSharedElementViewPrepared();
                    return true;
                }
            });
    return vh;
}
```

ItemAdapter.java

```
@Override
public RecyclerView.ViewHolder onCreateViewHolder(ViewGroup parent, int viewType) {
    View view = LayoutInflater.from(parent.getContext())
            .inflate(R.layout.item_viewholder, parent, false);
    final ViewHolder vh = new ViewHolder(view);

    ViewCompat.setTransitionName(vh.thumb, "shared_element_view");
    vh.thumb.getViewTreeObserver()
            .addOnPreDrawListener(new ViewTreeObserver.OnPreDrawListener() {
                @Override
                public boolean onPreDraw() {
                    vh.thumb.getViewTreeObserver().removeOnPreDrawListener(this);
                    mListener.onSharedElementViewPrepared();
                    return true;
                }
            });
    return vh;
}
```

Layout Problem

StatusBar, NavigationBarの前面に表示



Shared Element Targets

Shared Element View

StatusBar

NavigationBar

MainActivity.java

```
View statusBar = activity.findViewById(android.R.id.statusBarBackground);
View navigationBar = activity.findViewById(android.R.id.navigationBarBackground);

List<Pair<View, String>> pairs = new ArrayList<>();
pairs.add(Pair.create(statusBar, statusBar.getTransitionName()));
pairs.add(Pair.create(navigationBar, navigationBar.getTransitionName()));
pairs.add(Pair.create(sharedElementView, "shared_element_view"));

Bundle options = ActivityOptionsCompat.makeSceneTransitionAnimation(
    this,
    pairs.toArray(new Pair[pairs.size()])
).toBundle();

startActivity(intent, options);
```

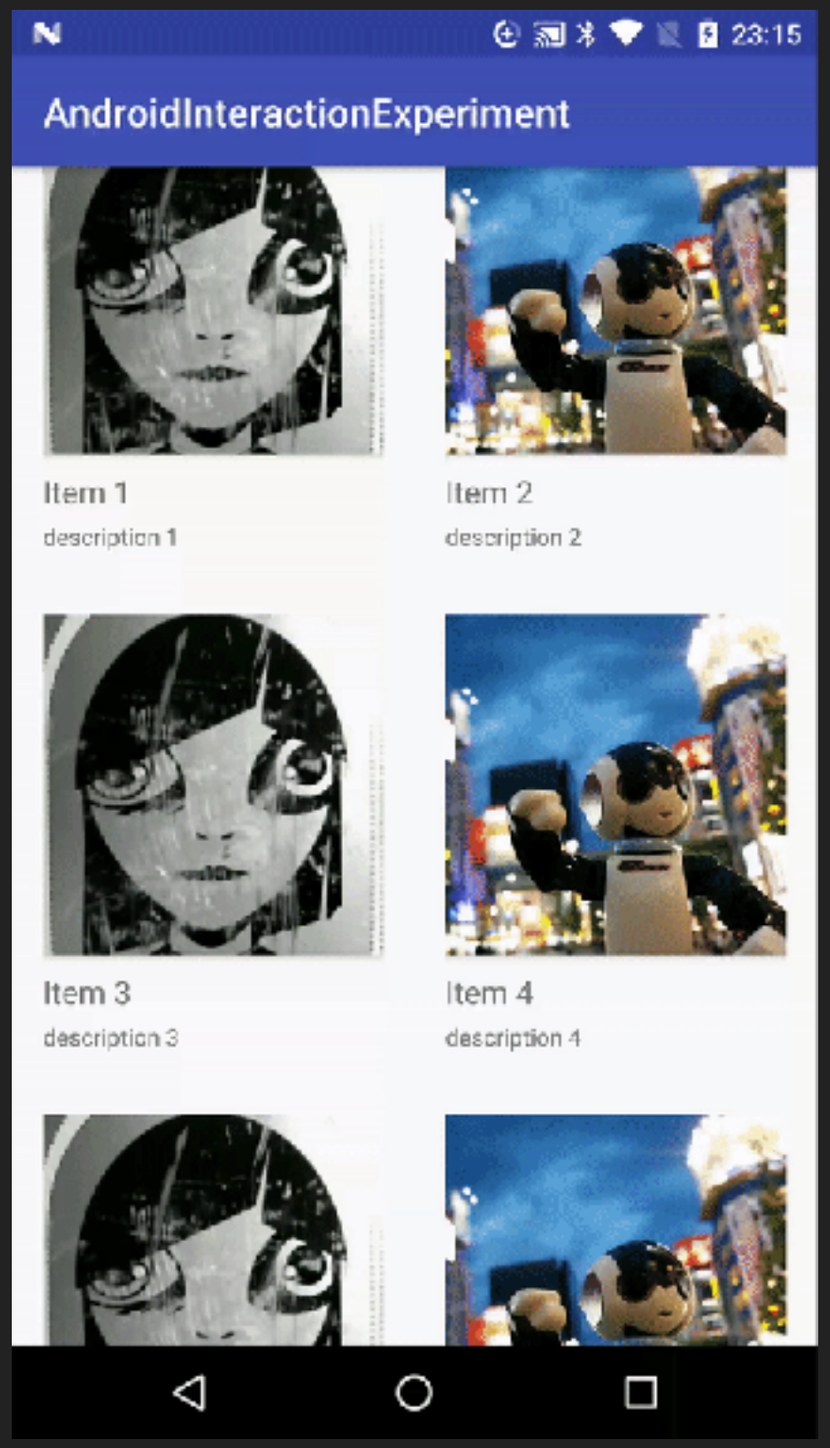
MainActivity.java

```
View statusBar = activity.findViewById(android.R.id.statusBarBackground);
View navigationBar = activity.findViewById(android.R.id.navigationBarBackground);

List<Pair<View, String>> pairs = new ArrayList<>();
pairs.add(Pair.create(statusBar, statusBar.getTransitionName()));
pairs.add(Pair.create(navigationBar, navigationBar.getTransitionName()));
pairs.add(Pair.create(sharedElementView, "shared_element_view"));

Bundle options = ActivityOptionsCompat.makeSceneTransitionAnimation(
    this,
    pairs.toArray(new Pair[pairs.size()])
).toBundle();

startActivity(intent, options);
```

MainActivity.java

```
View statusBar = activity.findViewById(android.R.id.statusBarBackground);
View navigationBar = activity.findViewById(android.R.id.navigationBarBackground);

List<Pair<View, String>> pairs = new ArrayList<>();
pairs.add(Pair.create(statusBar, statusBar.getTransitionName()));
pairs.add(Pair.create(navigationBar, navigationBar.getTransitionName()));
pairs.add(Pair.create(sharedElementView, "shared_element_view"));

Bundle options = ActivityOptionsCompat.makeSceneTransitionAnimation(
    this,
    pairs.toArray(new Pair[pairs.size()])
).toBundle();

startActivity(intent, options);
```

StatusBar, NavigationBarの背面に表示



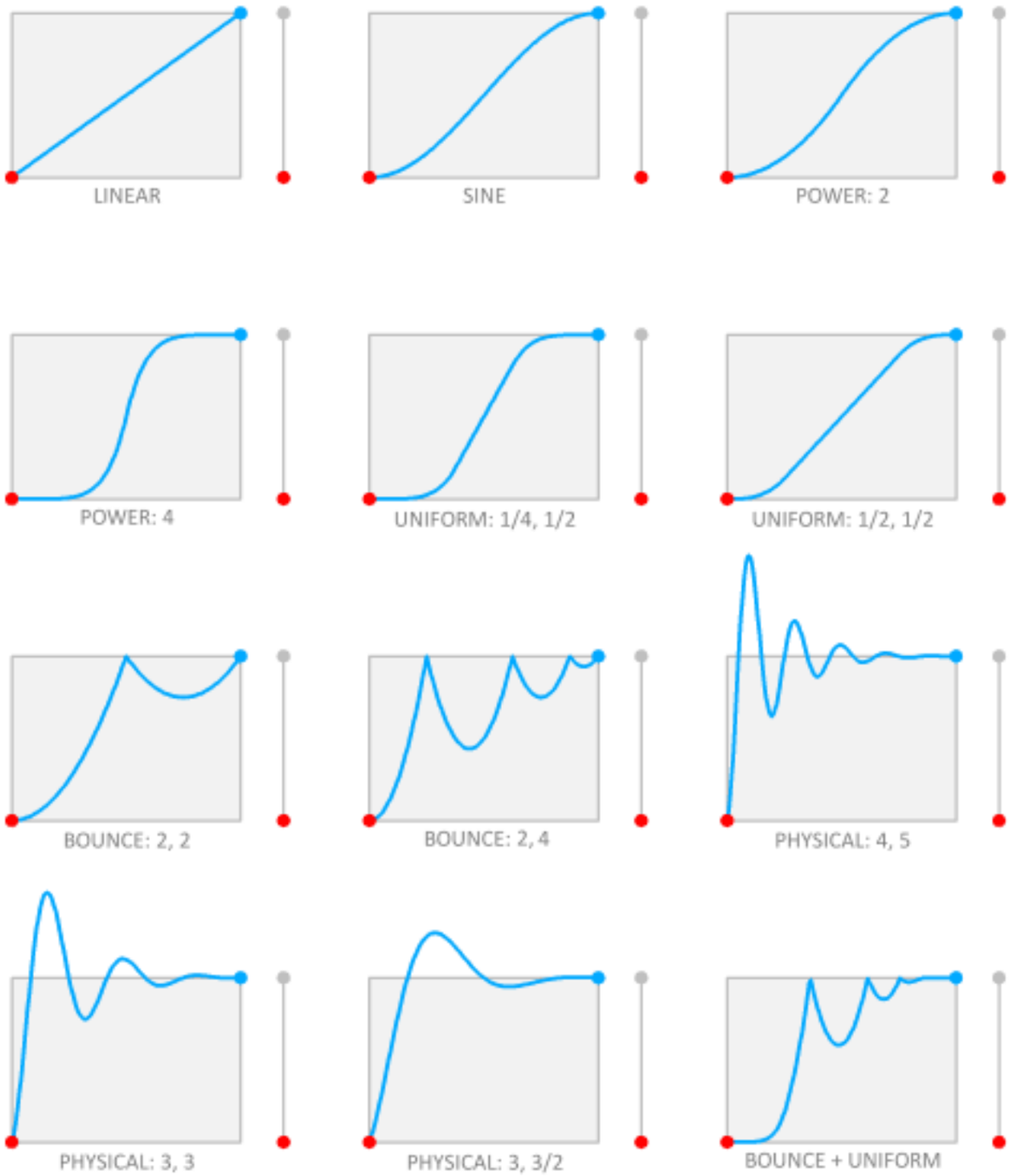
Set callback of Transition

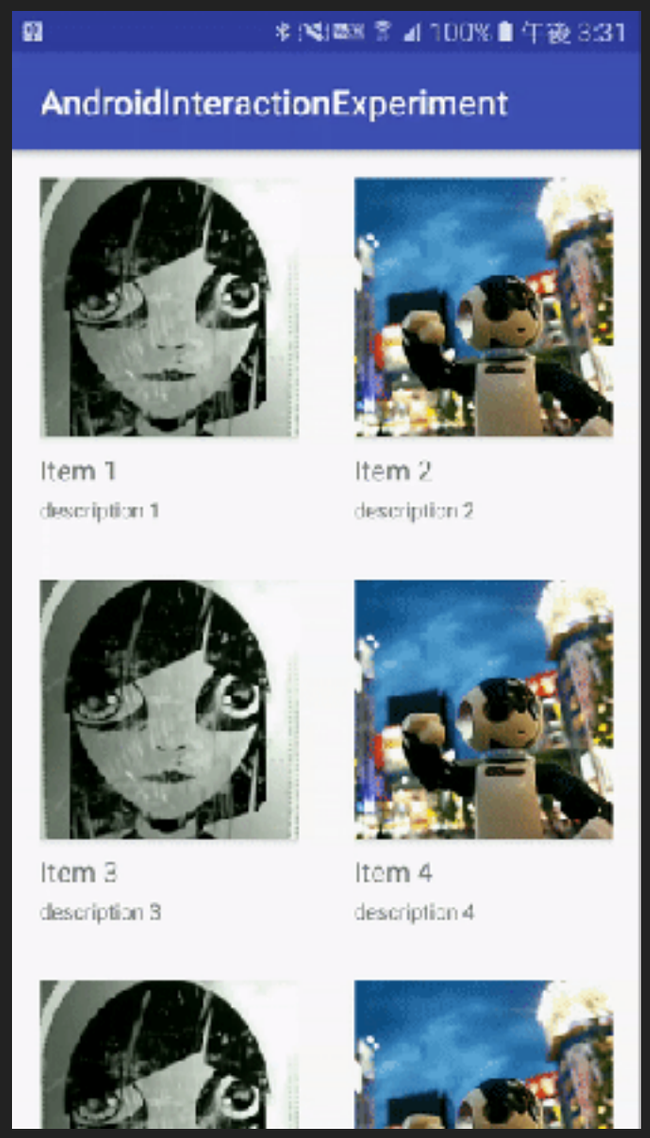
Detect animationEnd

- ✖ Activity.onEnterAnimationComplete
- ✖ Activity.setEnterSharedElementCallback
- ◎ Transition.TransitionListener

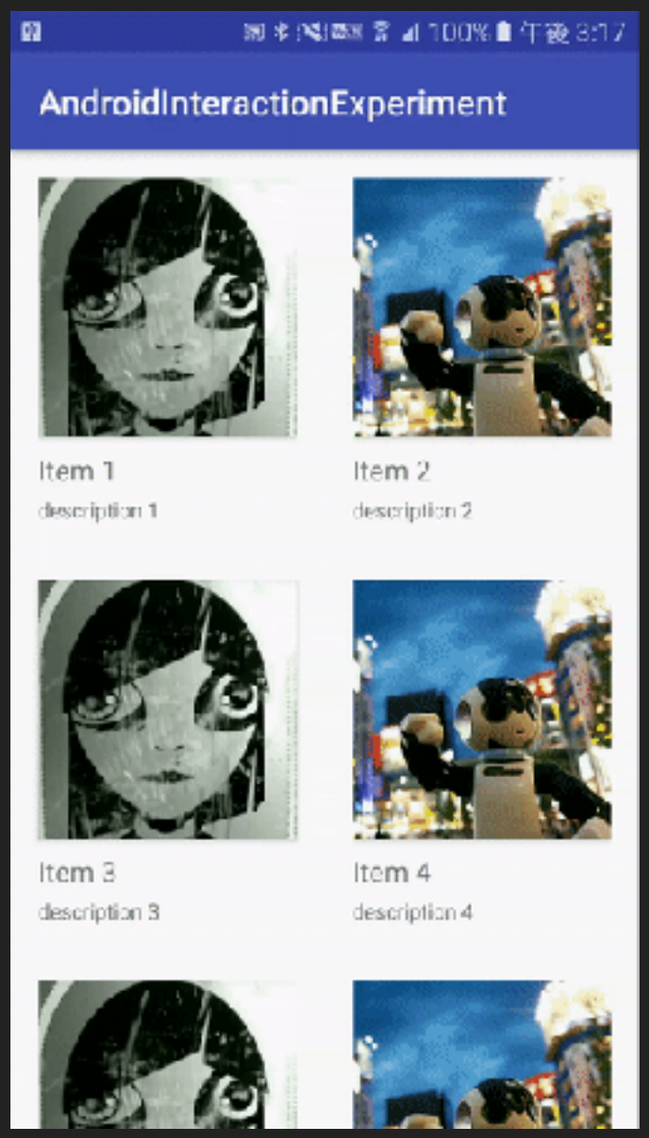
```
getWindow()
        .getSharedElementEnterTransition()
        .addListener(new Transition.TransitionListener() {
            @Override
            public void onTransitionEnd(Transition transition) {
                // Do something after shared element transition end.
            }
            ...
        });
```

Interpolator

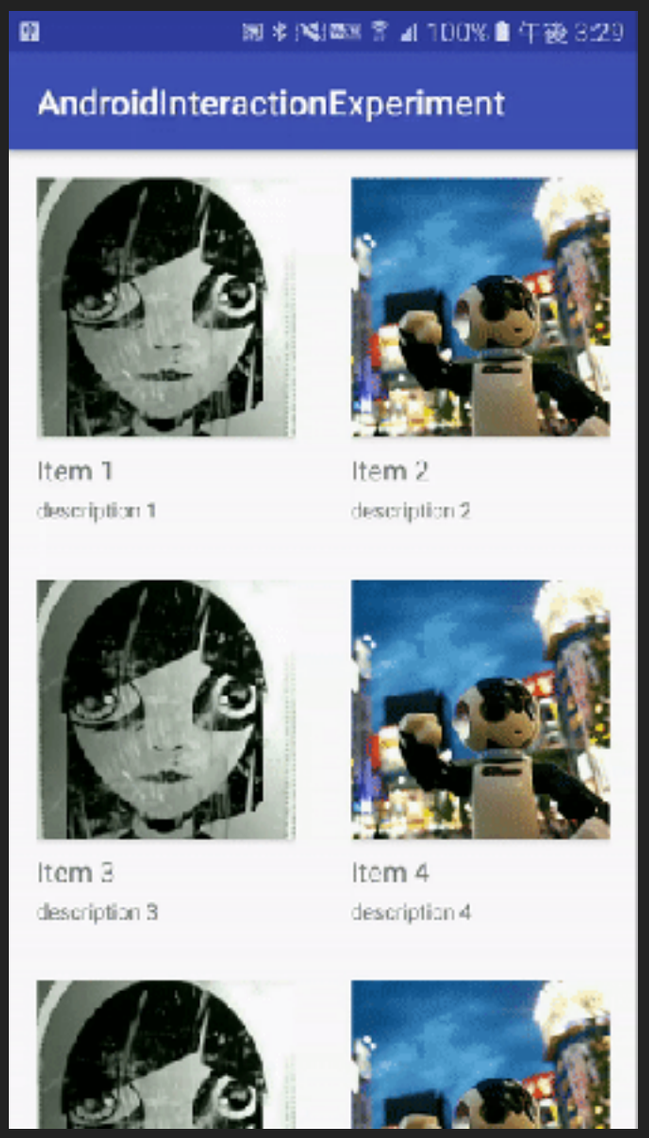

LINEAR
SINE
POWER: 2
POWER: 4
UNIFORM: 1/4, 1/2
UNIFORM: 1/2, 1/2
BOUNCE: 2, 2
BOUNCE: 2, 4
PHYSICAL: 4, 5
PHYSICAL: 3, 3
PHYSICAL: 3, 3/2
BOUNCE + UNIFORM
1ucasvb.tumblr.com



EXPO_IN_OUT



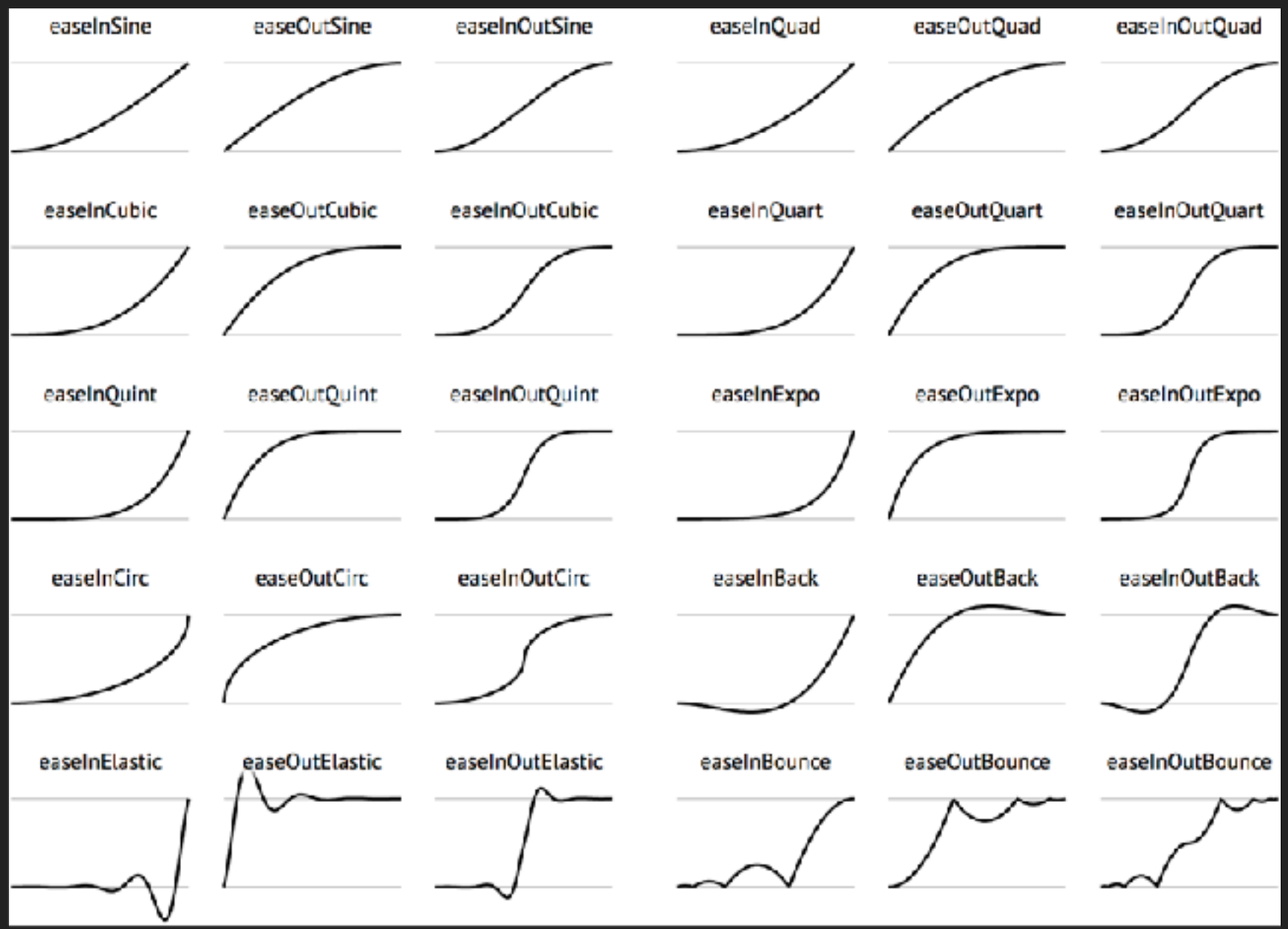
BACK_OUT



ELASTIC_OUT

```
@Override
protected void onCreate(Bundle savedInstanceState){
    ...
    Interpolator interpolator = new FastOutSlowInInterpolator();
    Transition transition = new ChangeBounds().setInterpolator(interpolator);
    getWindow().setSharedElementEnterTransition(transition);
}
```

Robert Penner's Easing Functions



http://robertpenner.com/easing/

AndroidRPInterpolator

https://github.com/r21nomi/AndroidRPInterpolator

Activity Transitions Using ViewPager

MainActivity.java

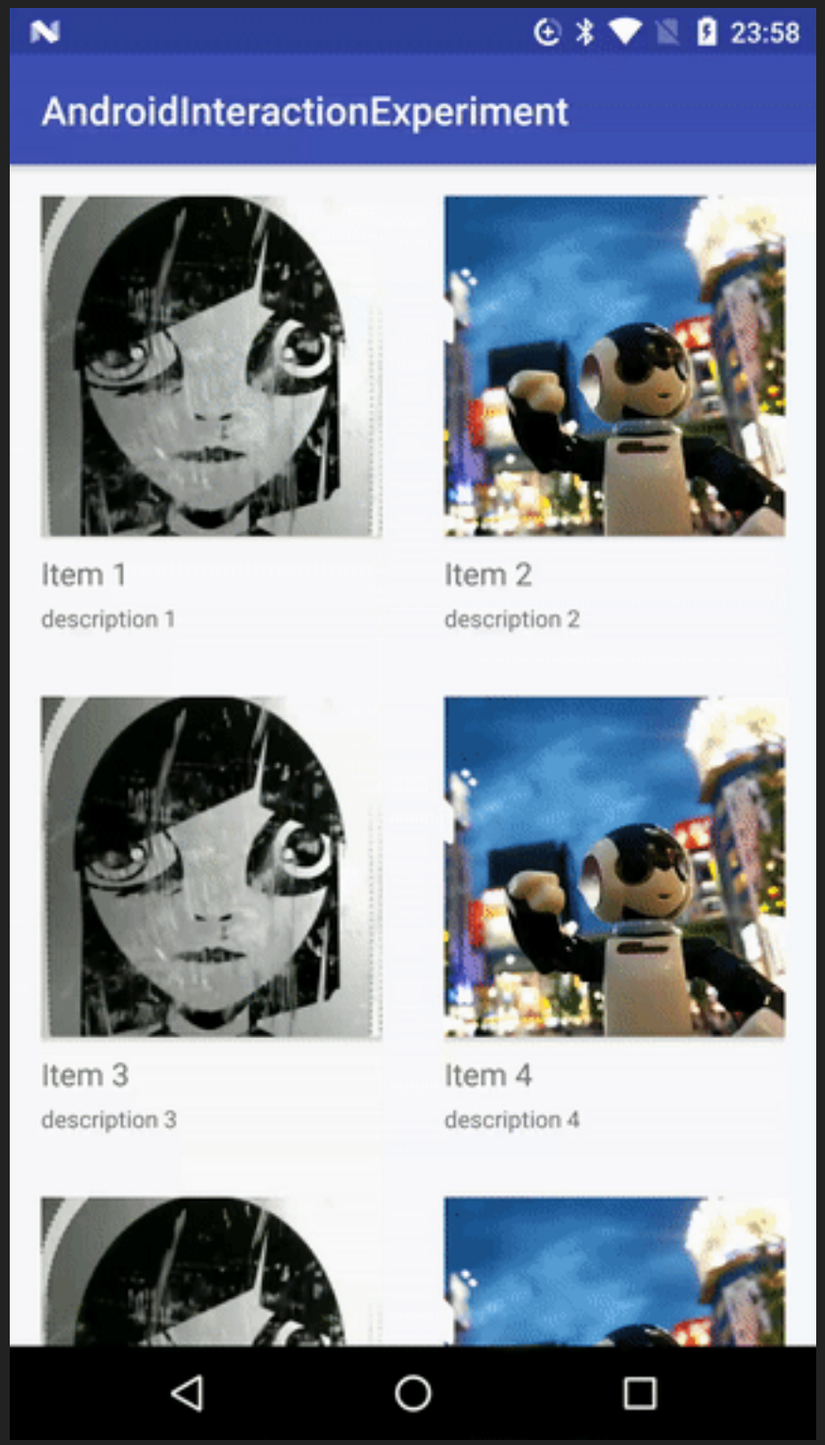
RecyclerView

DetailActivity.java

ViewPager

flick * n

?


23:58
AndroidInteractionExperiment
Item 1
description 1
Item 2
description 2
Item 3
description 3
Item 4
description 4

Shared Element Viewを入れ替え

`SharedElementCallback.onMapSharedElements()`

DetailActivity.java

```
setEnterSharedElementCallback(new SharedElementCallback() {
    @Override
    public void onMapSharedElements(List<String> names, Map<String, View>
sharedElements) {
        int position = mViewPager.getCurrentItem();
        View view = mViewPager.findViewWithTag(ItemPagerAdapter.getTag(position));
        sharedElements.clear();
        sharedElements.put("shared_element_view", view);
    }
});
```

DetailActivity.java

```
setEnterSharedElementCallback(new SharedElementCallback() {
    @Override
    public void onMapSharedElements(List<String> names, Map<String, View>
sharedElements) {
        int position = mViewPager.getCurrentItem();
        View view = mViewPager.findViewWithTag(ItemPagerAdapter.getTag(position));
        sharedElements.clear();
        sharedElements.put("shared_element_view", view);
    }
});
```

ViewPagerの描画準備完了を待つ

DetailActivity.java

```
postponeEnterTransition();
mViewPager.getViewTreeObserver().addOnPreDrawListener(new
ViewTreeObserver.OnPreDrawListener() {
    @Override
    public boolean onPreDraw() {
        mViewPager.getViewTreeObserver().removeOnPreDrawListener(this);
        startPostponedEnterTransition();
        return true;
    }
});
```

ViewPagerの描画準備完了を待つ

DetailActivity.java

```
postponeEnterTransition();
mViewPager.getViewTreeObserver().addOnPreDrawListener(new
ViewTreeObserver.OnPreDrawListener() {
    @Override
    public boolean onPreDraw() {
        mViewPager.getViewTreeObserver().removeOnPreDrawListener(this);
        startPostponedEnterTransition();
        return true;
    }
});
```

flickでMainActivityに通知するpositionを更新

DetailActivity.java

```
mViewPager.addOnPageChangeListener(new ViewPager.OnPageChangeListener() {
    @Override
    public void onPageSelected(int position) {
        Intent intent = new Intent();
        Bundle bundle = new Bundle();
        bundle.putInt("position", position);
        intent.putExtras(bundle);
        setResult(Activity.RESULT_OK, intent);
    }
    …
});
```

flickでMainActivityに通知するpositionを更新

DetailActivity.java

```
mViewPager.addOnPageChangeListener(new ViewPager.OnPageChangeListener() {
    @Override
    public void onPageSelected(int position) {
        Intent intent = new Intent();
        Bundle bundle = new Bundle();
        bundle.putInt("position", position);
        intent.putExtras(bundle);
        setResult(Activity.RESULT_OK, intent);
    }
    ...
});
```

SharedElementTransitionのバック時にdataと共に呼ばれる

```
Activity.onActivityReenter(…, Intent data)
```

MainActivity.java

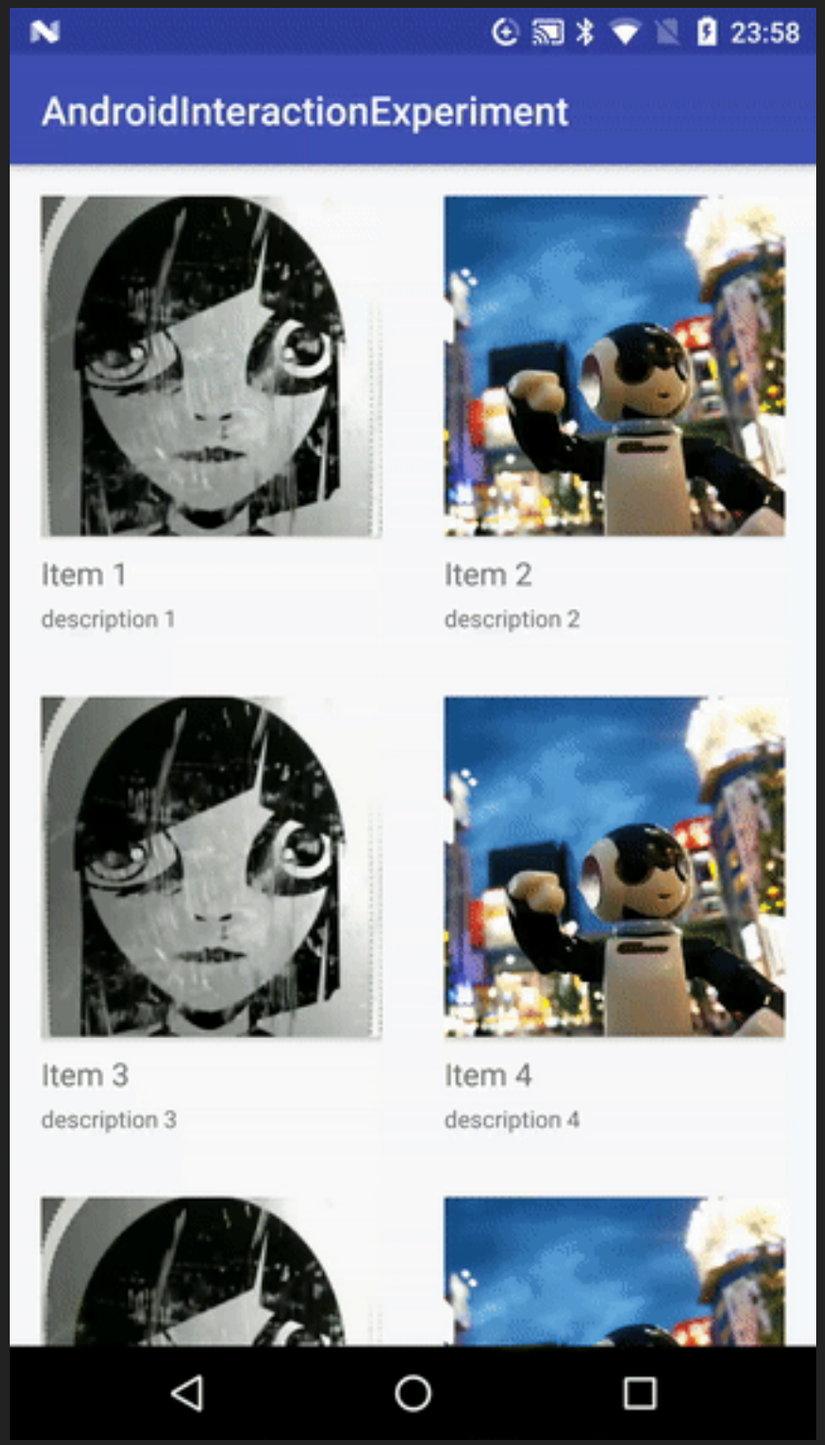
```
@Override
public void onActivityReenter(int resultCode, Intent data) {
    super.onActivityReenter(resultCode, data);

    int position = data.getIntExtra("position", 0);
    ItemAdapter.ViewHolder viewHolder = (ItemAdapter.ViewHolder)
mRecyclerView.findViewHolderForAdapterPosition(position);
    final View sharedElementView = (viewHolder == null) ? null : viewHolder.thumb;

    setExitSharedElementCallback(new SharedElementCallback() {
        @Override
        public void onMapSharedElements(List<String> names, Map<String, View>
sharedElements) {
            sharedElements.clear();
            sharedElements.put("shared_element_view", sharedElementView);
            setExitSharedElementCallback((SharedElementCallback) null);
        }
    });
}
```

MainActivity.java

```
@Override
public void onActivityReenter(int resultCode, Intent data) {
    super.onActivityReenter(resultCode, data);

    int position = data.getIntExtra("position", 0);
    ItemAdapter.ViewHolder viewHolder = (ItemAdapter.ViewHolder)
mRecyclerView.findViewHolderForAdapterPosition(position);
    final View sharedElementView = (viewHolder == null) ? null : viewHolder.thumb;

    setExitSharedElementCallback(new SharedElementCallback() {
        @Override
        public void onMapSharedElements(List<String> names, Map<String, View>
sharedElements) {
            sharedElements.clear();
            sharedElements.put("shared_element_view", sharedElementView);
            setExitSharedElementCallback((SharedElementCallback) null);
        }
    });
}
```

MainActivity.java

```
@Override
public void onActivityReenter(int resultCode, Intent data) {
    super.onActivityReenter(resultCode, data);

    int position = data.getIntExtra("position", 0);
    ItemAdapter.ViewHolder viewHolder = (ItemAdapter.ViewHolder)
mRecyclerView.findViewHolderForAdapterPosition(position);
    final View sharedElementView = (viewHolder == null) ? null : viewHolder.thumb;

    setExitSharedElementCallback(new SharedElementCallback() {
        @Override
        public void onMapSharedElements(List<String> names, Map<String, View>
sharedElements) {
            sharedElements.clear();
            sharedElements.put("shared_element_view", sharedElementView);
            setExitSharedElementCallback((SharedElementCallback) null);
        }
    });
}
```


23:58
AndroidInteractionExperiment
Item 1
description 1
Item 2
description 2
Item 3
description 3
Item 4
description 4

Summary

・Shared Elementの対象にはStatusBar、NavigationBarを含める

・適宜、postponeEnterTransitionでアニメーションを遅延させる

・OnPreDrawListener.onPreDrawを使う

・適宜、onMapSharedElementsでShared Element対象を入れ替え

Self Implementation for Shared Element Transition

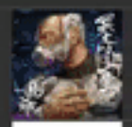
0:24

初音

完了

"初音" in TRACKS

1,000+ TRACKS

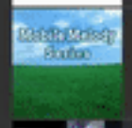


初音ミクの消失

島爺

初音ミクの暴走(-Full ver.-)

cosMo@暴走P



初音ミクの消失 (COSMO@暴走P : オリジナル歌...

Mobile Melody Series

初音ミク for DTM (feat. 初音ミク)

AVTechNO!

初音ミクの消失 -MIKUNOPOLIS in LOS ANGELES...

cosMo@暴走P

初音ミクの消失(feat.石川綾子)

cosMo@暴走P

初音奔放曲

ラマーズP

初音ミクの暴走

cosMo@暴走P feat.vip店長

1

No Selection

2

No Selection

3

No Selection

4

No Selection

MainActivity

1. viewのsize, positionをDetailActivityに渡す

DetailActivity

1. Shared element viewにsize, positionを設定
2. アニメーション

MainActivity

1. viewのsize, positionをDetailActivityに渡す

```
int[] location = new int[2];
view.getLocationOnScreen(location);

location[0];  // left
location[1];  // top
```

MainActivity

1. viewのsize, positionをDetailActivityに渡す

```
public class Position implements Parcelable {
    int left;
    int top;
    int width;
    int height;

    public Position(View view) {
        int[] location = new int[2];
        view.getLocationOnScreen(location);

        left = location[0];
        top = location[1];
        width = view.getWidth();
        height = view.getHeight();
    }
    ...
}
```

DetailActivity

1. Shared Element Viewにsize, positionを設定

```
// Set size
ViewGroup.LayoutParams param = view.getLayoutParams();
param.width = position.getWidth();
param.height = position.getHeight();
view.setLayoutParams(param);
```

DetailActivity

1. Shared Element Viewにsize, positionを設定

```
// Wait for view preparing
view.getViewTreeObserver().addOnPreDrawListener(new
ViewTreeObserver.OnPreDrawListener() {
    @Override
    public boolean onPreDraw() {
    view.getViewTreeObserver().removeOnPreDrawListener(this);

    float top = position.getTop() - getStatusBarheight();
    // Use setX / setY to set absolute position.
    view.setX(position.getLeft());
    view.setY(top);

    startEnterAnimation();
    return true;
    }
});
```

DetailActivity

1. Shared Element Viewにsize, positionを設定

```
// Wait for view preparing
view.getViewTreeObserver().addOnPreDrawListener(new
ViewTreeObserver.OnPreDrawListener() {
    @Override
    public boolean onPreDraw() {
    view.getViewTreeObserver().removeOnPreDrawListener(this);

    float top = position.getTop() - getStatusBarheight();
    // Use setX / setY to set absolute position.
    view.setX(position.getLeft());
    view.setY(top);

    startEnterAnimation();
    return true;
    }
});
```

Shared Element Viewのtop − statusBarの高さ

Point

position.getTop() つまり location[1]
はstatus barの高さを含む

DetailActivity

1. Shared Element Viewにsize, positionを設定

```
// Wait for view preparing
view.getViewTreeObserver().addOnPreDrawListener(new
ViewTreeObserver.OnPreDrawListener() {
    @Override
    public boolean onPreDraw() {
    view.getViewTreeObserver().removeOnPreDrawListener(this);

    float top = position.getTop() - getStatusBarheight();
    // Use setX / setY to set absolute position.
    view.setX(position.getLeft());
    view.setY(top);

    startEnterAnimation();
    return true;
    }
});
```

✖ `view.setTranslationY(top);`

◎ `view.setY(top);`

```
// 元の位置から計算
✖  view.setTranslationY(top);

// 基準点から計算
◎  view.setY(top);
```

DetailActivity

2. アニメーション

```
private void startEnterAnimation() {
    int targetWidth = getTargetWidth();
    int targetHeight = targetWidth * position.getWidth() / position.getHeight();

    AnimatorSet animSet = new AnimatorSet();
    animSet.playTogether(
    view.getToRectAnimator(),
        // Use translationX / translationY to translate to relative position.
        ObjectAnimator.ofFloat(view, "translationX", 0),
        ObjectAnimator.ofFloat(view, "translationY", 0),
        ValueAnimator.ofObject(new WidthEvaluator(view), position.getWidth(),
targetWidth),
        ValueAnimator.ofObject(new HeightEvaluator(view), position.getHeight(),
targetHeight)
    );
    animSet.start();
}
```

DetailActivity

2. アニメーション

```
private void startEnterAnimation() {
    int targetWidth = getTargetWidth();
    int targetHeight = targetWidth * position.getWidth() / position.getHeight();

    AnimatorSet animSet = new AnimatorSet();
    animSet.playTogether(
    view.getToRectAnimator(),
        // Use translationX / translationY to translate to relative position.
        ObjectAnimator.ofFloat(view, "translationX", 0),
        ObjectAnimator.ofFloat(view, "translationY", 0),
        ValueAnimator.ofObject(new WidthEvaluator(view), position.getWidth(),
targetWidth),
        ValueAnimator.ofObject(new HeightEvaluator(view), position.getHeight(),
targetHeight)
    );
    animSet.start();
}
```

DetailActivity

2. アニメーション

“translationY” == setTranslationY()

DetailActivity

2. アニメーション

✖ “y”

◎ “translationY”

DetailActivity

2. アニメーション

```
private void startEnterAnimation() {
    int targetWidth = getTargetWidth();
    int targetHeight = targetWidth * position.getWidth() / position.getHeight();

    AnimatorSet animSet = new AnimatorSet();
    animSet.playTogether(
    view.getToRectAnimator(),
        // Use translationX / translationY to translate to relative position.
        ObjectAnimator.ofFloat(view, "translationX", 0),
        ObjectAnimator.ofFloat(view, "translationY", 0),
        ValueAnimator.ofObject(new WidthEvaluator(view), position.getWidth(),
targetWidth),
        ValueAnimator.ofObject(new HeightEvaluator(view), position.getHeight(),
targetHeight)
    );
    animSet.start();
}
```

DetailActivity

2. アニメーション

```
private void startEnterAnimation() {
    int targetWidth = getTargetWidth();
    int targetHeight = targetWidth * position.getWidth() / position.getHeight();

    AnimatorSet animSet = new AnimatorSet();
    animSet.playTogether(
    view.getToRectAnimator(),
        // Use translationX / translationY to translate to relative position.
        ObjectAnimator.ofFloat(view, "translationX", 0),
        ObjectAnimator.ofFloat(view, "translationY", 0),
        ValueAnimator.ofObject(new WidthEvaluator(view), position.getWidth(),
targetWidth),
        ValueAnimator.ofObject(new HeightEvaluator(view), position.getHeight(),
targetHeight)
    );
    animSet.start();
}
```

WidthEvaluator

アニメーション値をwidthに設定

```
public class WidthEvaluator extends IntEvaluator {
    private View mView;

    public WidthEvaluator(View view) {
        mView = view;
    }

    @Override
    public Integer evaluate(float fraction, Integer startValue, Integer endValue) {
        Integer num = super.evaluate(fraction, startValue, endValue);
        ViewGroup.LayoutParams params = mView.getLayoutParams();
        params.width = num;
        mView.setLayoutParams(params);
        return num;
    }
}
```

Summary

・View.getLocationOnScreen(int[] outLocation)で位置を取得

・OnPreDrawListener.onPreDrawでViewの計算完了を待つ

・初期位置の設定はsetY()で

・Y座標の設定にはstatus barを考慮する

Thank you

r21nomi

@r21nomi