Androids *do* Daydream

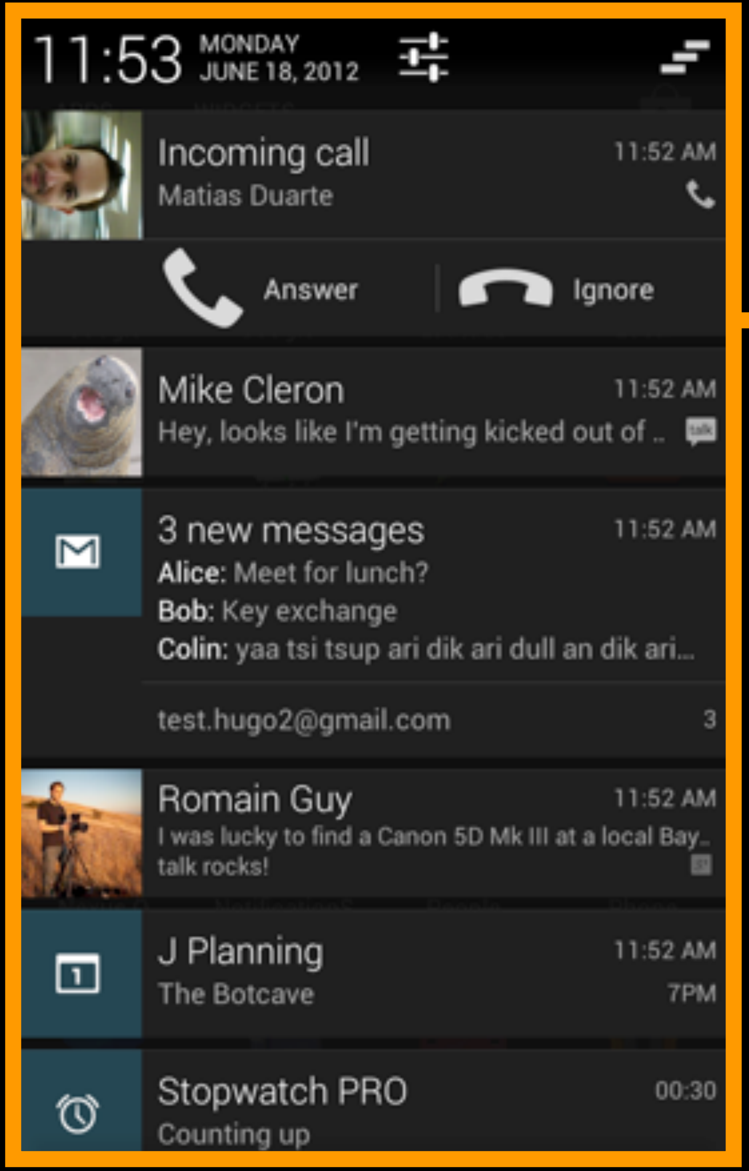
Daniel Sandler
Android System UI Team

hi

system ui

4:20
Google
Maps
Play Store
YouTube
Google

LAUNCHER
4:20
Google
Maps
Play Store
YouTube
Google



11:53 MONDAY JUNE 18, 2012

Incoming call 11:52 AM
Matias Duarte
Answer Ignore

Mike Cleron 11:52 AM
Hey, looks like I'm getting kicked out of ..

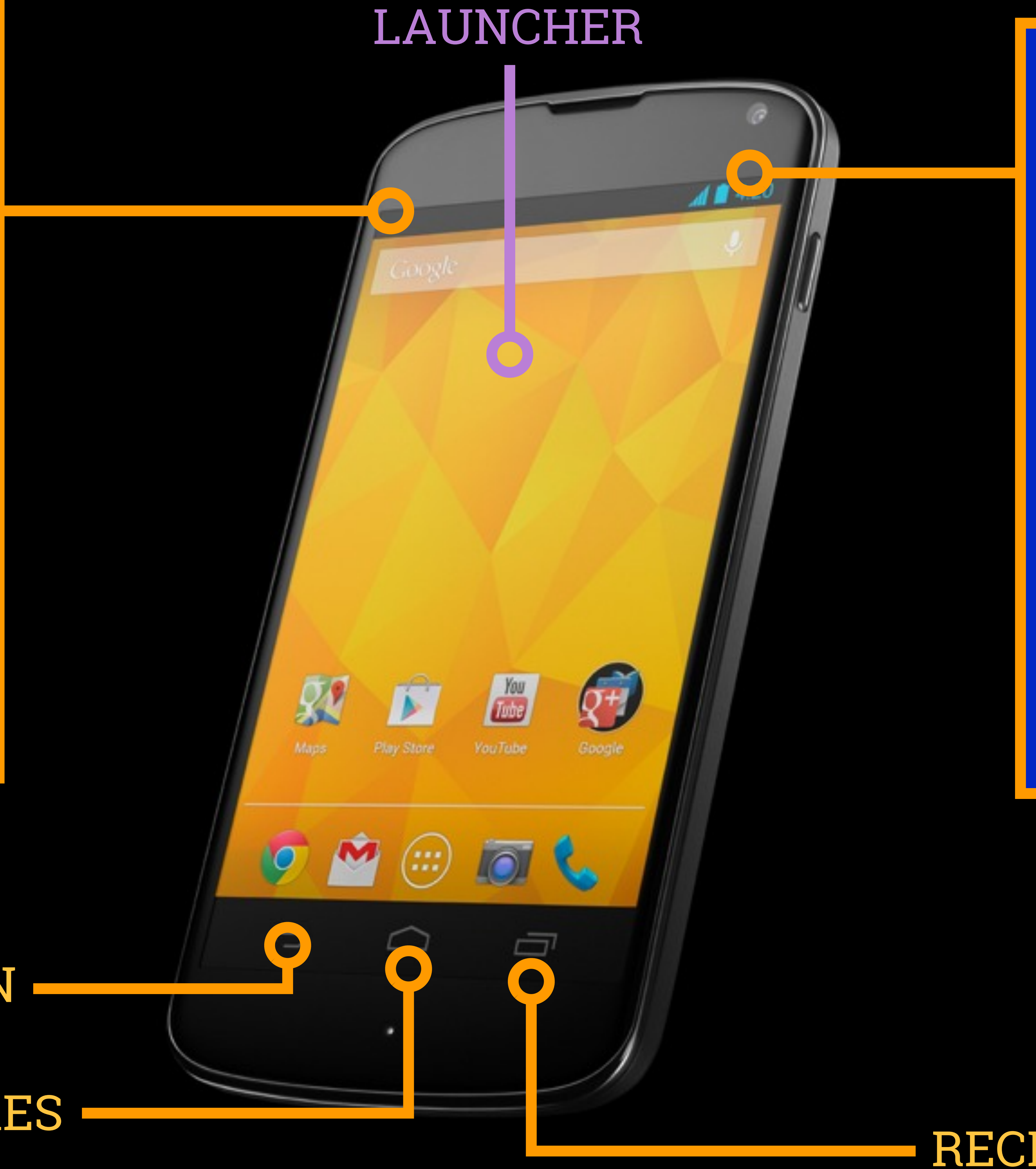
3 new messages 11:52 AM
Alice: Meet for lunch?
Bob: Key exchange
Colin: yaa tsi tsup ari dik ari dull an dik ari...
test.hugo2@gmail.com 3

Romain Guy 11:52 AM
I was lucky to find a Canon 5D Mk III at a local Bay.. talk rocks!

J Planning 11:52 AM
The Botcave 7PM

Stopwatch PRO 00:30
Counting up

NOTIFICATIONS



QUICK SETTINGS

MAIN NAVIGATION

GLOBAL GESTURES

RECENT APPS

It’s a screen saver, essentially.

why?

we must
go back to **2009**

desk dock

car dock

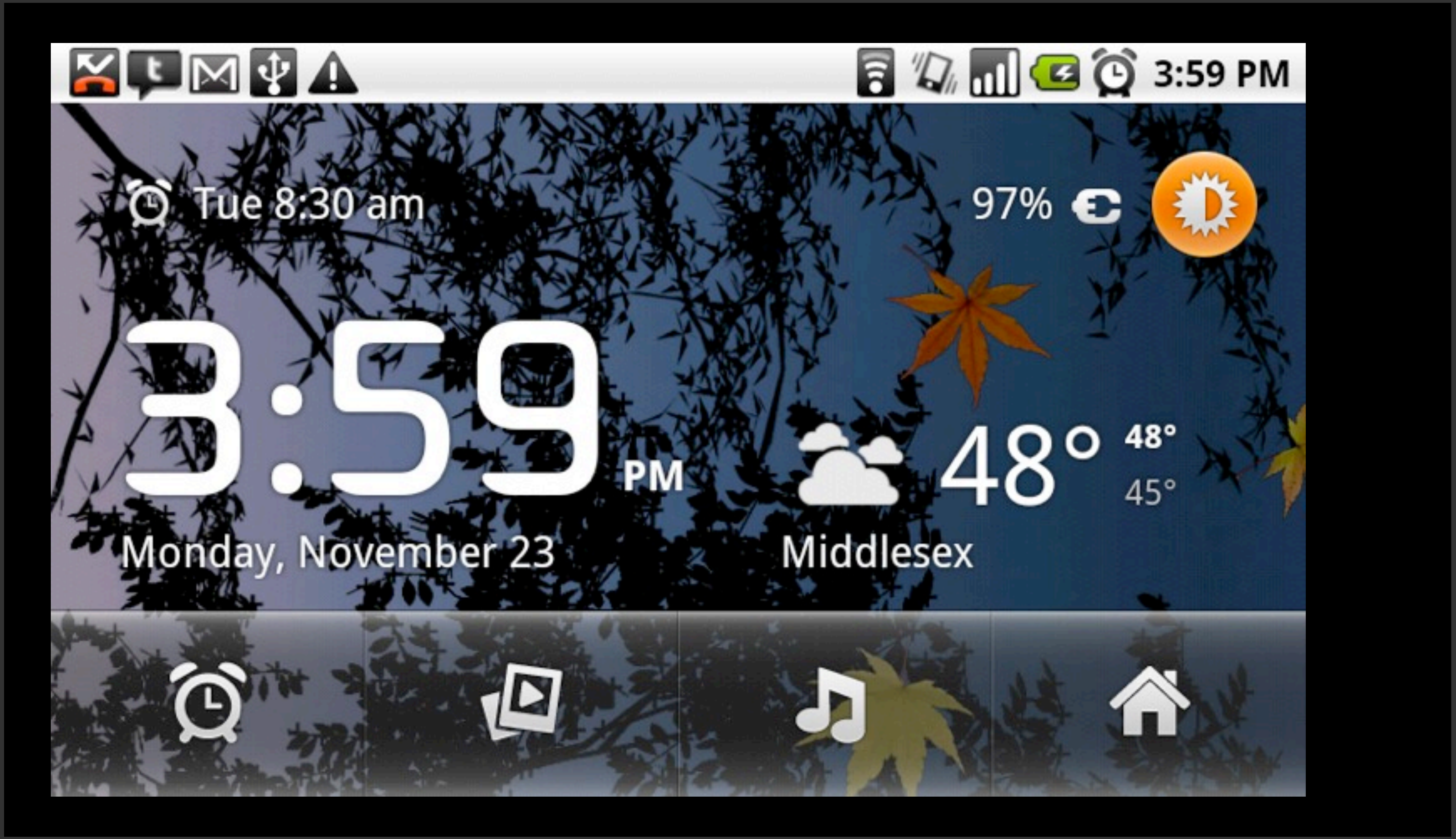
MOTOROLA
Head northwest on W 53rd St toward Ave of the Americas
New York
2:55 PM

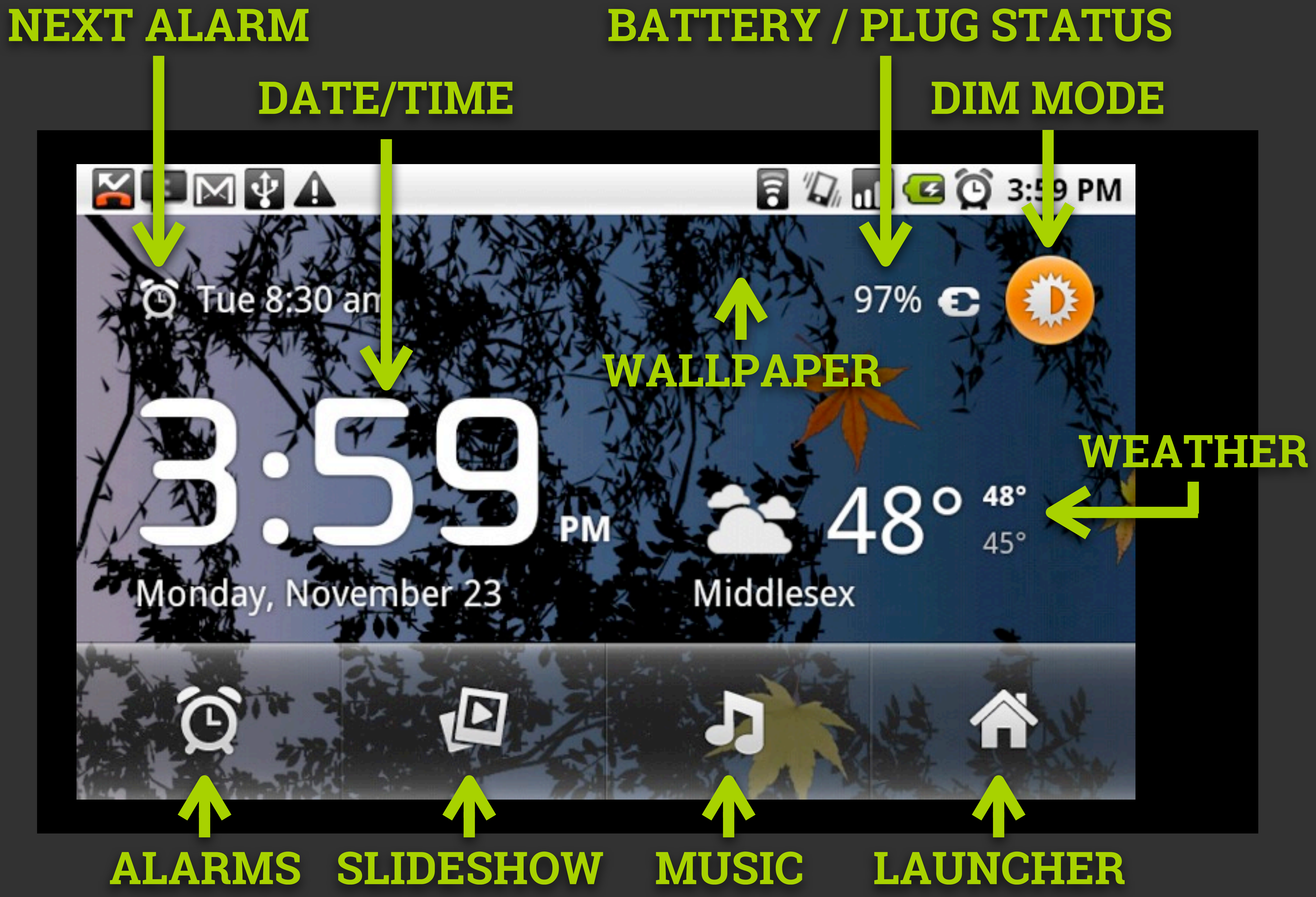
hat dock?

#notglass

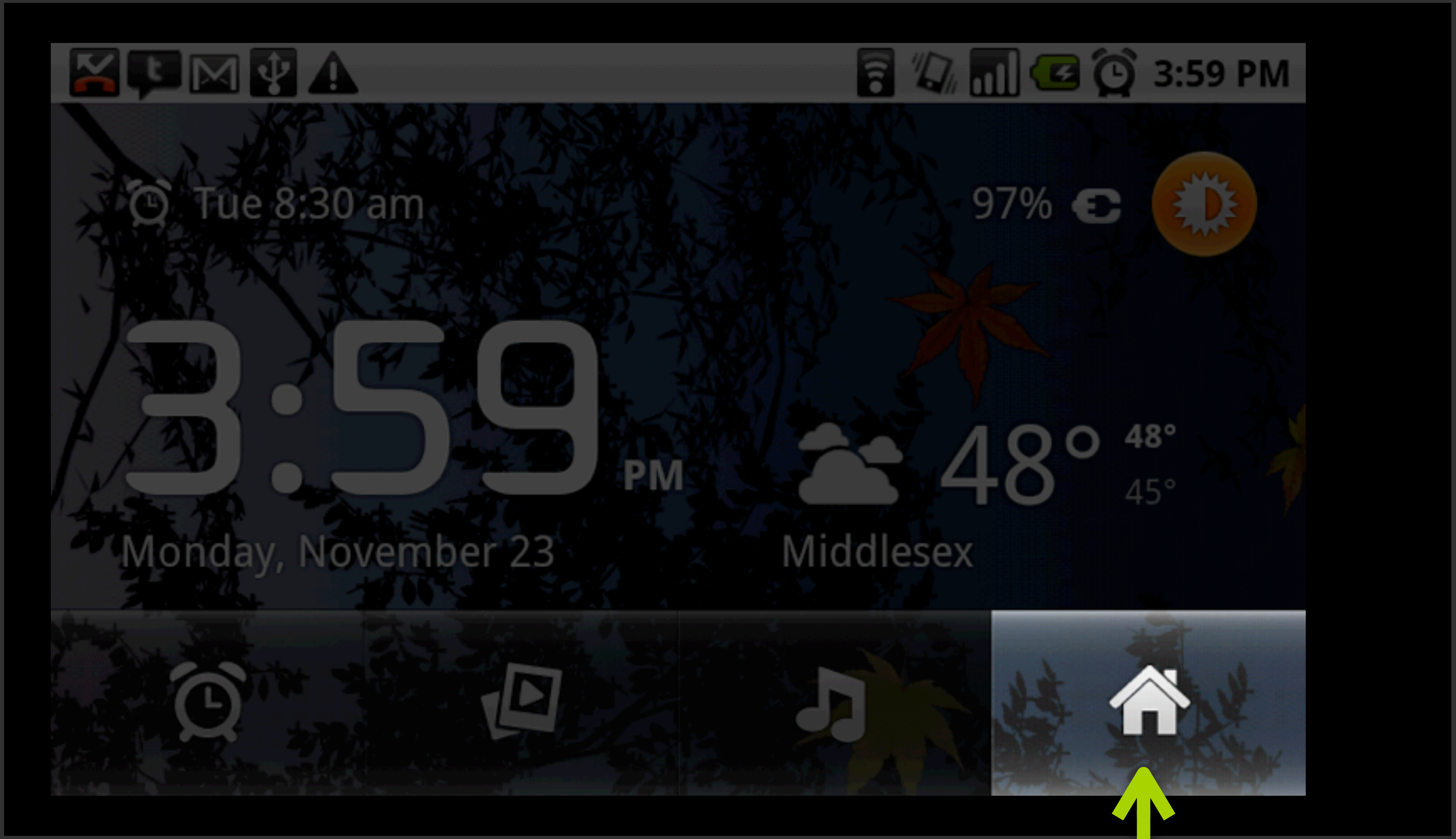
“dock mode”

“dock **apps**”

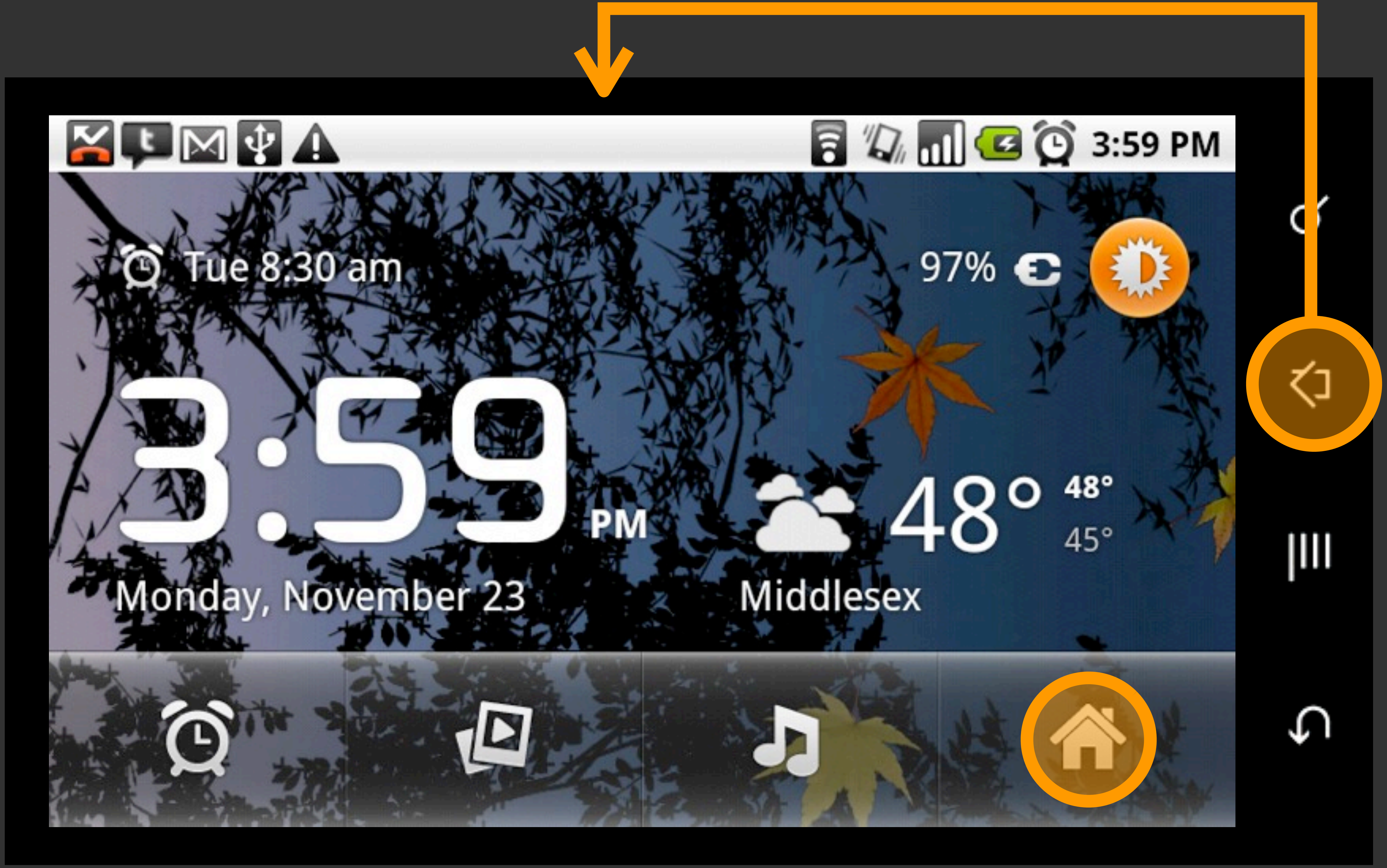

3:59 PM
Tue 8:30 am
97%
3:59 PM
Monday, November 23
48°
48°
45°
Middlesex



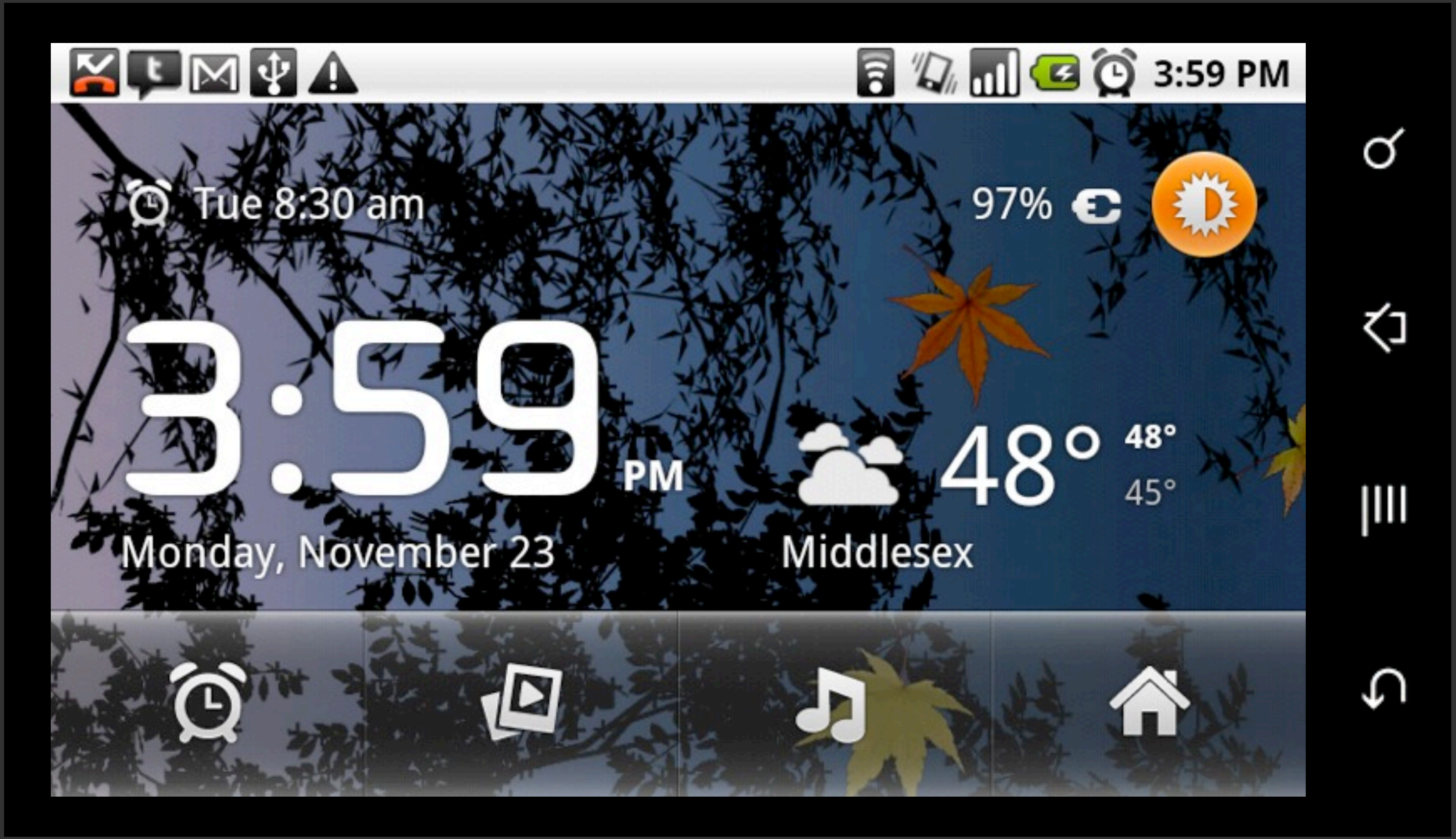
NEXT ALARM
DATE/TIME
BATTERY / PLUG STATUS
DIM MODE
3:59 PM
Tue 8:30 am
97%
WALLPAPER
3:59 PM
Monday, November 23
WEATHER
48° 48° 45°
Middlesex
ALARMS
SLIDESHOW
MUSIC
LAUNCHER


3:59 PM
Tue 8:30 am
97%
3:59 PM
Monday, November 23
48°
48°
45°
Middlesex

WHY?


3:59 PM
Tue 8:30 am
97%
3:59 PM
Monday, November 23
48°
48°
45°
Middlesex

? ? ? ?


3:59 PM
Tue 8:30 am
97%
3:59
PM
48°
48°
45°
Monday, November 23
Middlesex

3:59

3:59

3:59

7:52P
Monday, March 21
Set alarm

11:46
Monday, January 9
Set alarm
11:46

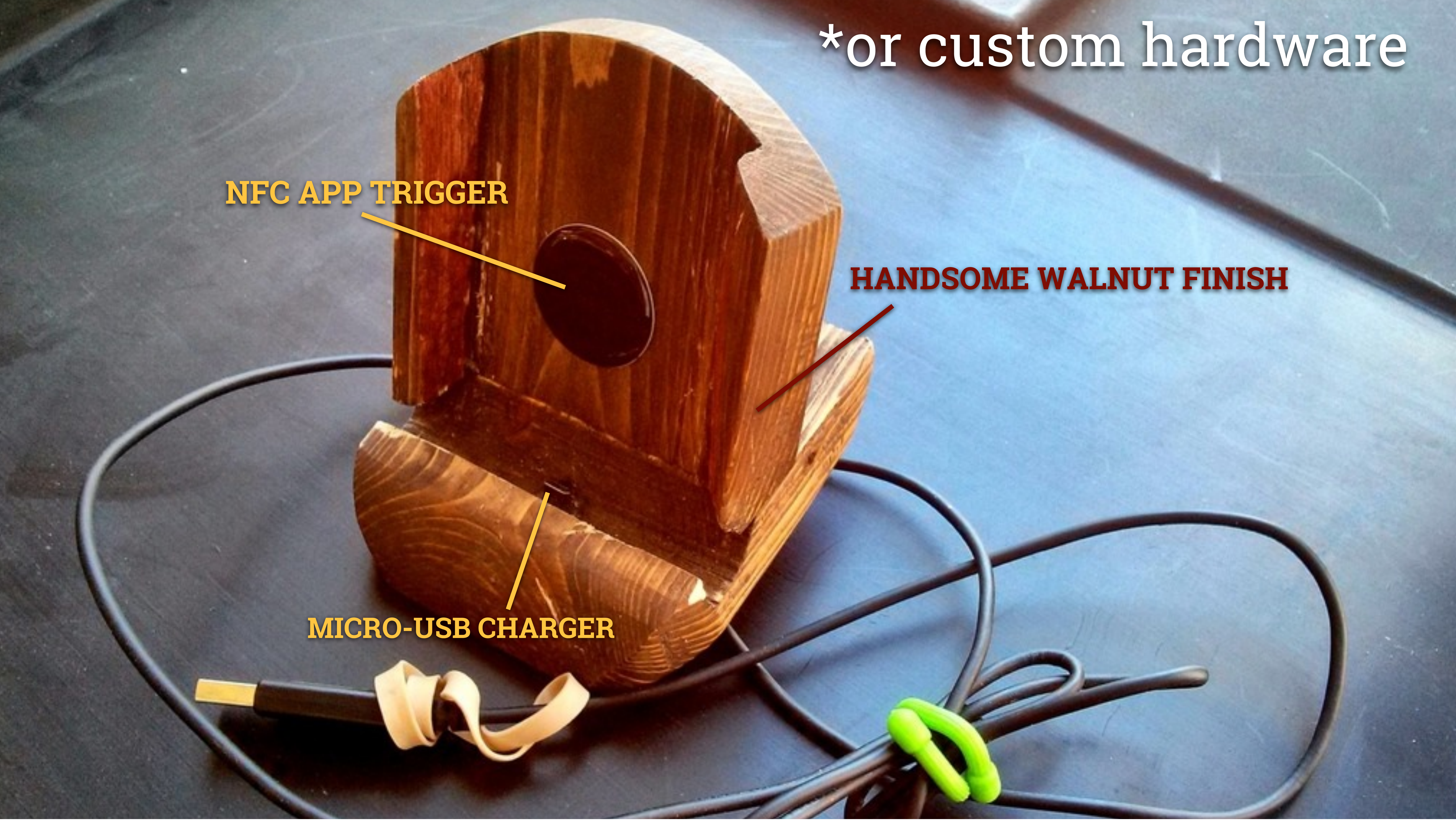
dock apps are cool

+ customized UX
+ easy to access
+ glanceable, ambient info

dock apps are annoying

- home-key capture is confusing
- requires special hardware

*or custom hardware



writing dock apps is tricky

should you capture the home key?

no → no easy way to get back in

yes → great, now you need a “real launcher” button

wake locks

great, now you need to watch the plug state

display burn-in

great, now you need a screen saver timer

something better

daydream

API 17 / ANDROID 4.2

features

easy to write

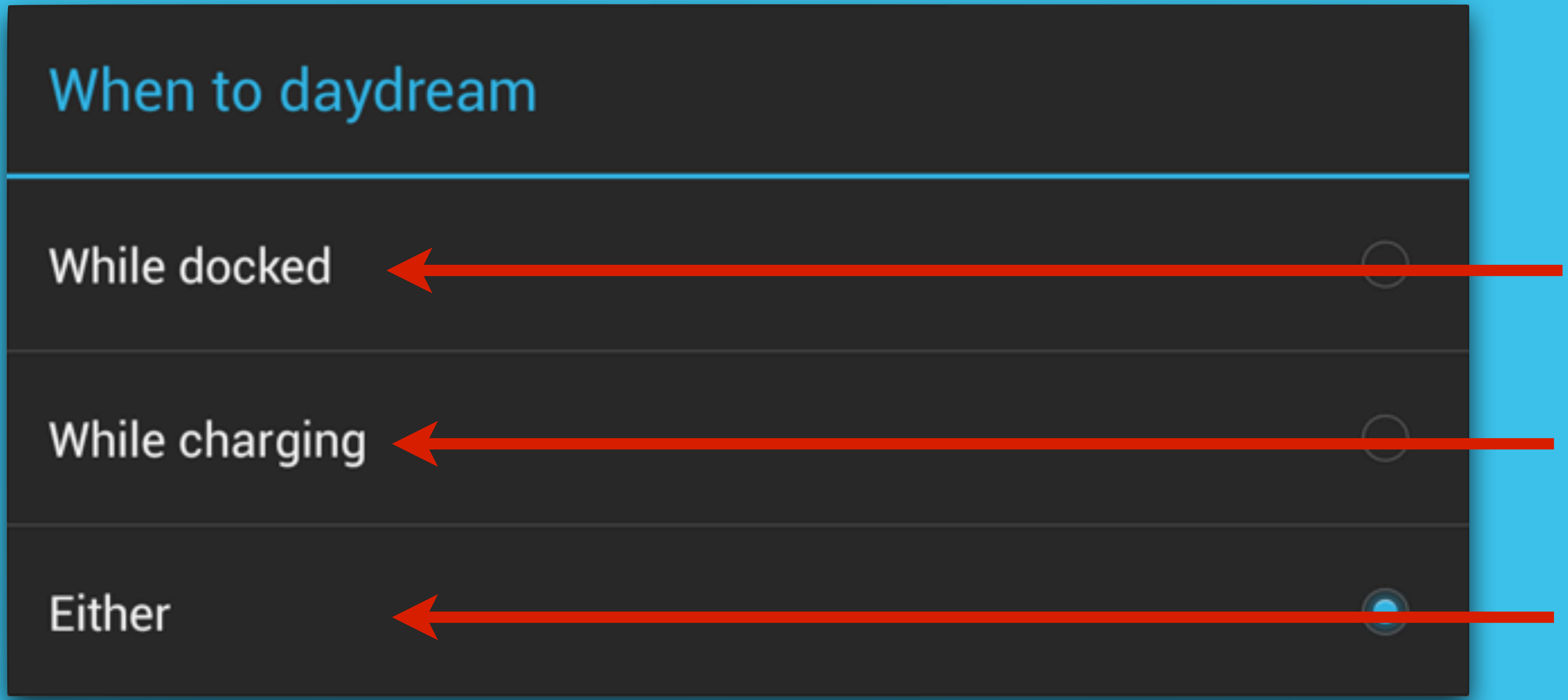
launched automatically

full drawing API

full touch/interaction

other free stuff

daydream will start



When to daydream

While docked

While charging

[✓ Either]

like a dock app

like a screen saver

like a BOSS

unless something is holding a wake lock

keep screen on → “do not interrupt”

how to write one

1. extend **DreamService**

2. do whatever you want!

```
public class MyDaydream extends DreamService {
    public void onDreamingStarted() {
        super.onDreamingStarted();
        setContentView( some view or layout );
        // whatever, as promised
    }
}
```

why a service?

early prototype based on Activity

perhaps you saw it...

[Updated: Not An Easter Egg But A Screensaver!] Weird, Star Wars-Like Light Speed Launcher Found Hidden In The Depths Of Ice Cream Sandwich

Posted by Artem Russakovskii in Galaxy Nexus, Ice Cream Sandwich 4.0, News, Off-Topic, Videos

androidpolice.com

it didn't work so well

lots of bookkeeping: plug & dock state, manage wake locks

power manager had a hard time keeping track

wrong window layer

do whatever

standard layouts & widgets

user interaction

custom views

view animation

canvas 2D drawing

OpenGL 3D drawing

video

do whatever

standard layouts & widgets*

user interaction*

} launcher widgets

custom views

view animation

canvas 2D drawing

OpenGL 3D drawing

video

do whatever

standard layouts & widgets

user interaction

custom views

view animation

canvas 2D drawing

OpenGL 3D drawing

video

$\}$ live wallpapers

do whatever

standard layouts & widgets

user interaction

custom views

view animation

canvas 2D drawing

OpenGL 3D drawing

video

$\Bigg\}$ **daydream**

other Daydream API freebies

automatically started & stopped
by the power manager

choose interactive/noninteractive

a special window on top

but...whyyyy

a chance for
your app to be
free

free
of all its
features

free
of all its
use cases

free
of all its
menus

free
of all its
settings

to be

experimental

to be

weird

to be

delightful.

can has code pls

1

periodic animation

Jumper.java

```
public class Jumper extends FrameLayout {
    Runnable mRunnable = new Runnable() {
        public void run() {
            final View parent = (View) getParent();
            if (parent == null) return;

            // reposition in parent using setX() and setY()
            final float width = getMeasuredWidth();
            final float height = getMeasuredHeight();
            final float parentw = parent.getMeasuredWidth();
            final float parenth = parent.getMeasuredHeight();
            setX((float) Math.random() * (parentw - width));
            setY((float) Math.random() * (parenth - height));

            postDelayed(this, 2000); // let's do this again, soon
        }
    };
```

Jumper.java

```
    public void onAttachedToWindow() {
        getHandler().post(mRunnable);
    }

    // usual View constructors
}
```

layout/main.xml

```
<FrameLayout
      android:layout_width="match_parent"
      android:layout_height="match_parent" >

      <com.example.daydream.jumper.Jumper
          android:layout_width="wrap_content"
          android:layout_height="wrap_content">

          <!-- whatever you want; ImageView, etc. -->

      </com.example.daydream.jumper.Jumper>

</FrameLayout>
```

nothing
daydream-specific
so far!

JumperDaydream.java

```
public class JumperDaydream extends DreamService {
    public void onDreamingStarted() {
        setContentView(R.layout.main);
    }
}
```

JumperDaydream.java

```
public class JumperDaydream extends DreamService {
    public void onDreamingStarted() {
        setContentView(R.layout.main);
    }
}
```

Simple Activity-like lifecycle:

onAttachedToWindow()
... onDreamingStarted()
... onDreamingStopped()
... onDetachedFromWindow().

AndroidManifest.xml

```
<manifest package="com.example.daydream.jumper"
    [...] >
    <uses-sdk android:targetSdkVersion="17" android:minSdkVersion="17"/>

    <application android:label="@string/app_name">
        <service android:name=".JumperDaydream"
                 android:exported="true">
            <intent-filter>
                <category android:name="android.intent.category.DEFAULT" />
                <action android:name="android.service.dreams.DreamService" />
            </intent-filter>
        </service>
    </application>
</manifest>
```

a little fancier?

Jumper.java

```java
public class Jumper extends FrameLayout {
    Runnable mRunnable = new Runnable() {
        public void run() {
            final View parent = (View) getParent();
            if (parent == null) return;

            // reposition in parent using setX() and setY()
            final float width = getMeasuredWidth();
            final float height = getMeasuredHeight();
            final float parentw = parent.getMeasuredWidth();
            final float parenth = parent.getMeasuredHeight();
            setX((float) Math.random() * (parentw - width));
            setY((float) Math.random() * (parenth - height));

            postDelayed(this, 2000); // let's do this again, soon
        }
    };
```

Jumper.java

```
public class Jumper extends FrameLayout {
    Runnable mRunnable = new Runnable() {
        public void run() {
            final View parent = (View) getParent();
            if (parent == null) return;

            // reposition in parent using animation
            final float width = getMeasuredWidth();
            final float height = getMeasuredHeight();
            final float parentw = parent.getMeasuredWidth();
            final float parenth = parent.getMeasuredHeight();
            animate().x((float) Math.random() * (parentw - textw))
                     .y((float) Math.random() * (parenth - texth))
                     .rotateBy(360) // whee!
                     .setDuration(500);

            postDelayed(this, 2000); // let’s do this again, soon
        }};
```

keep going?

main.xml

```
<FrameLayout>
    <com.android.daydream.jumper.Jumper
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" >
        <ImageView
            android:layout_width="180dp"
            android:layout_height="180dp"
            android:src="@drawable/matiasduarte" />
    </com.android.daydream.jumper.Jumper>
</FrameLayout>
```

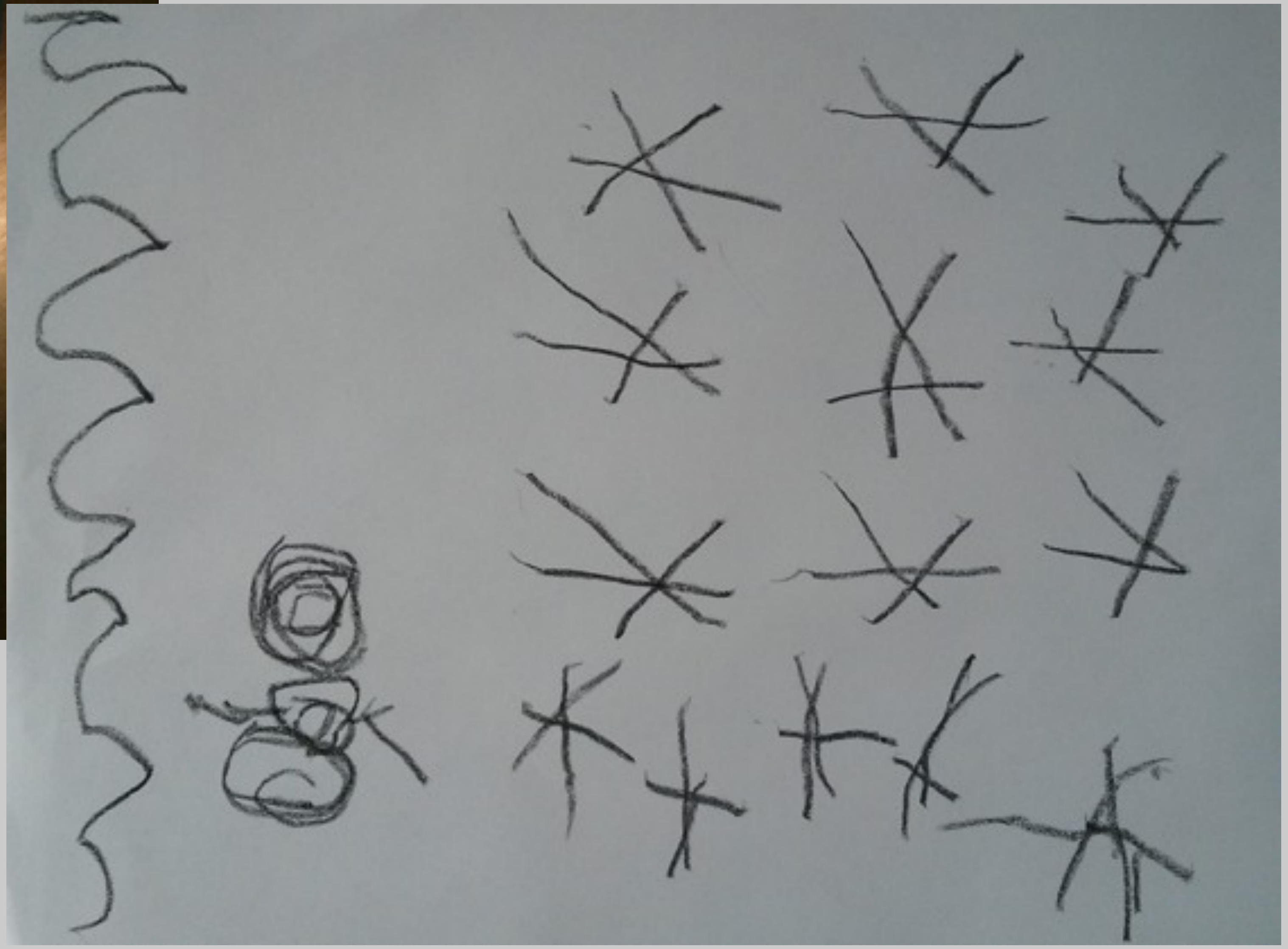
main.xml

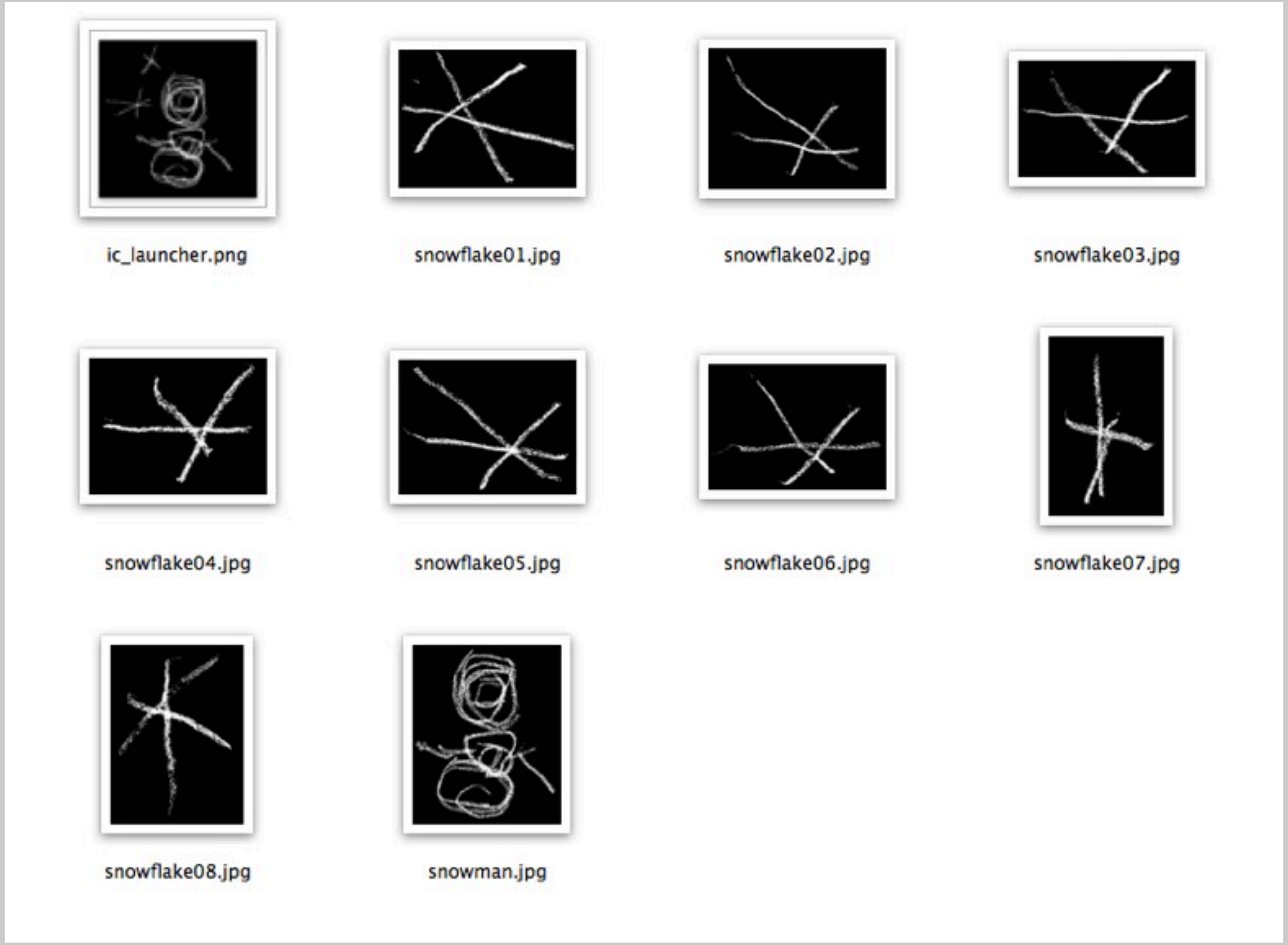
```
<FrameLayout>
    <com.android.daydream.jumper.Jumper
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" >
        <ImageView
            android:layout_width="180dp"
            android:layout_height="180dp"
            android:src="@drawable/matiasduarte" />
    </com.android.daydream.jumper.Jumper>
    <com.android.daydream.jumper.Jumper [...]>
        [...]
    </com.android.daydream.jumper.Jumper>
    <com.android.daydream.jumper.Jumper [...]>
        [...]
    </com.android.daydream.jumper.Jumper>
    <com.android.daydream.jumper.Jumper [...]>
        [...]
    </com.android.daydream.jumper.Jumper>
    [...]
```

ok I think we're done with that

2

continuous animation




ic_launcher.png
snowflake01.jpg
snowflake02.jpg
snowflake03.jpg
snowflake04.jpg
snowflake05.jpg
snowflake06.jpg
snowflake07.jpg
snowflake08.jpg
snowman.jpg

SnowyDay.java

```
public class SnowyDay extends FrameLayout
        implements TimeAnimator.TimeListener {

    TimeAnimator mAnimator; // we’ll use this to trigger each frame

    public SnowyDay(Context context, AttributeSet attrs, int flags) {
        super(context, attrs, flags);
        mAnimator = new TimeAnimator();
        mAnimator.setTimeListener(this);
    }
    public void onAttachedToWindow() {
        super.onAttachedToWindow();
        mAnimator.start();  // animation will start running continuously
    }
    public void onDetachedFromWindow() {
        mAnimator.cancel(); // halt animation
        super.onDetachedFromWindow();
    }
```

SnowyDay.java (2)

```
// you can put anything in this layout and we'll make it fall

public void addView(View v, LayoutParams lp) {
    super.addView(v, lp);

    // FlakeInfo stores positional and rotational velocity
    // initial values are random
    v.setTag(new FlakeInfo());

    // sprinkle liberally
    v.setX((float) (Math.random() * (myWidth - v.getWidth())));
    v.setY((float) (Math.random() * (myHeight - v.getHeight())));
}
```

SnowyDay.java (3)

```
// called every vsync'd frame
public void onTimeUpdate(TimeAnimator animation, long elapsed, long dt_ms) {
    final float dt = dt_ms / 1000f; // seconds
    for (int i=0; i<getChildCount(); i++) {
        final View view = getChildAt(i);
        // I hid some things here
        final FlakeInfo info = (FlakeInfo) view.getTag();

        // 1. step simulation for velocity * time
        view.setX(view.getX() + info.vx * dt);
        view.setY(view.getY() + info.vy * dt);
        view.setRotation(view.getRotation() + info.va * dt);

        // 2. wrap around all edges
        [omitted for brevity]
    }
}
```

SnowflakesDaydream.java

```
public class SnowflakesDaydream extends DreamService {
    static final int SNOWFLAKES[] = {
        R.drawable.snowflake01,
        R.drawable.snowflake02, [...]
    };

    // adjust based on weather forecast
    static final int NUM_SNOWFLAKES = 30;

    public void onDreamingStarted() {
        setFullscreen(true);
        setContentView(makeSnowyDay(this));
    }
```

SnowflakesDaydream.java

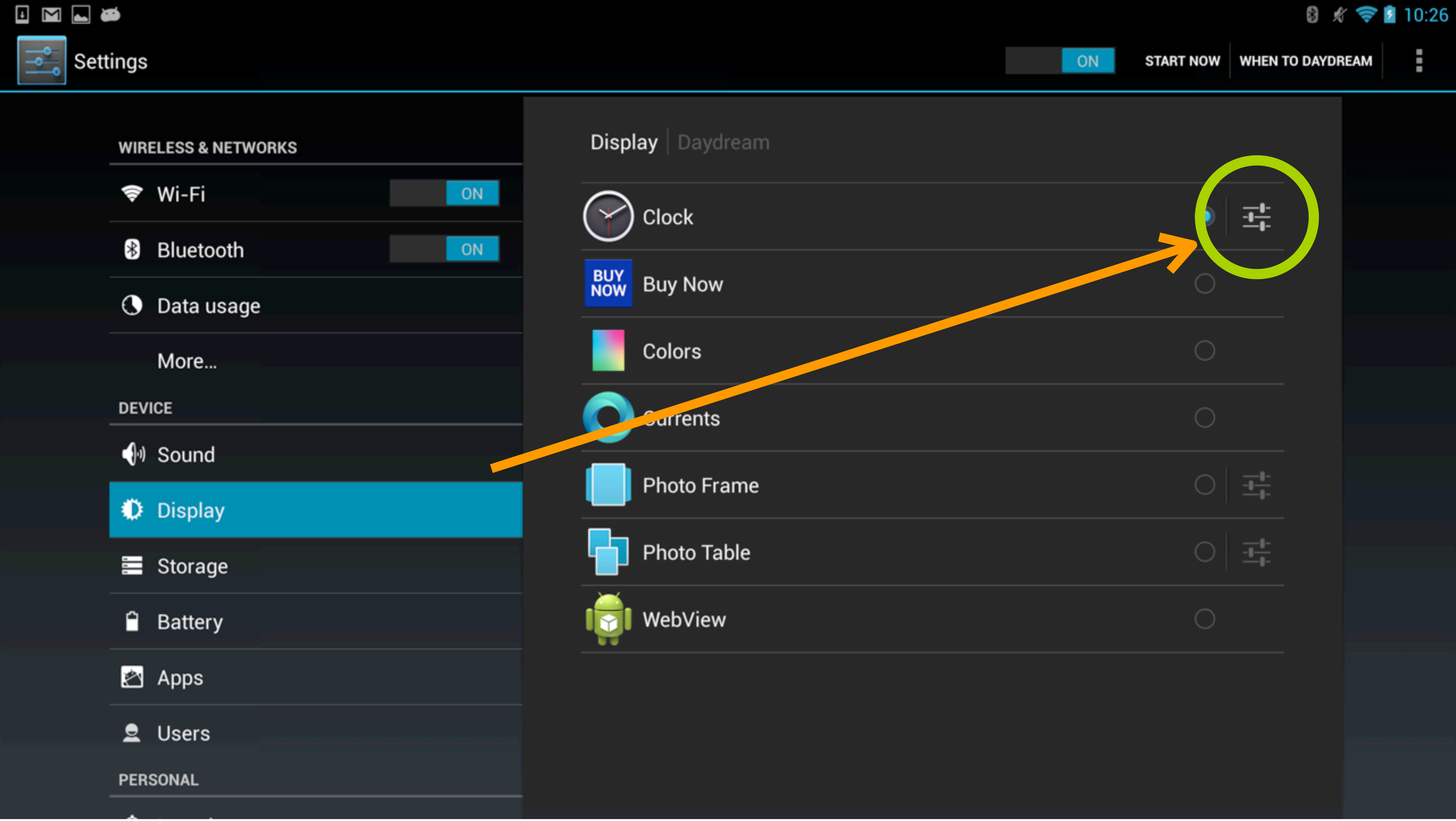
```
public static View makeSnowyDay(Context context) {
    SnowyDay scene = new SnowyDay(context);

    // a trick for eliminating the black boxes in the sprites
    final Paint screenPaint = new Paint();
    screenPaint.setXfermode(new PorterDuffXfermode(PorterDuff.Mode.SCREEN));

    for (int i=0; i<NUM_FLAKES; i++) {
        final ImageView image = new ImageView(context);
        image.setImageResource(random_choice(SNOWFLAKES));
        image.setLayerType(View.LAYER_TYPE_HARDWARE, screenPaint);
        scene.addView(image);
    }

    return scene;
}
```

3

what else?

```
setInteractive(true)
```

add a configuration Activity



Settings

ON | START NOW | WHEN TO DAYDREAM

WIRELESS & NETWORKS
- Wi-Fi ON
- Bluetooth ON
- Data usage
- More...

DEVICE
- Sound
- Display
- Storage
- Battery
- Apps
- Users

PERSONAL

Display | Daydream

- Clock
- Buy Now
- Colors
- Currents
- Photo Frame
- Photo Table
- WebView

AndroidManifest.xml

```
<service android:name=".MyDayDream">
    [...]
    <meta-data
        android:name="android.service.dream"
        android:resource="@xml/dream_info" />
</service>
```

res/xml/dream_info.xml

```
<dream
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:settingsActivity="com.example.dream.SettingsActivity"
    />
```

develop using an Activity

```
public class MyDaydream extends DreamService {
    public void onDreamingStarted() {
        super.onDreamingStarted();
        setContentView(R.layout.main);
        // go nuts, as promised
    }
}
```

```
public class MyDaydreamTest extends Activity {
    public void onStart() {
        super.onStart();
        setContentView(R.layout.main);
        // go nuts, as promised
    }
}
```

```
public class MyDaydream extends DreamService {
    public void onDreamingStarted() {
        super.onDreamingStarted();
        setContentView(new DreamView());
    }
}
public class MyDaydreamTest extends Activity {
    public void onStart() {
        super.onStart();
        setContentView(new DreamView());
    }
}
public class DreamView extends View {
    // go nuts in one place
}
```

port your live wallpaper

BUY NOW

BuyNowDaydream.java

```java
public void onDreamingStarted() {
    super.onDreamingStarted();
    mView = new View(this);
    mView.setBackgroundDrawable(new BuyNowDrawable(...));
    setContentView(mView);
    drawFrame();
}

void drawFrame() {
    mView.postInvalidate();

    mHandler.removeCallbacks(mRedraw);
    mHandler.postDelayed(mRedraw, REFRESH_INTERVAL);
}
```

use TextureView and OpenGL

prevent Daydream from running

…by keeping the screen on

```
window.addFlags(
    WindowManager.LayoutParams.
        FLAG_KEEP_SCREEN_ON)
```

```
View.setKeepScreenOn(true)
```

```
android:keepScreenOn=”true”
```

or manage your own wake lock

4

a magic trick

a **clock**

in zero* lines of

code

(well, large values of zero)

```
// this is just boilerplate, it doesn’t even count
public class Daydream extends DreamService {
    public void onDreamingStarted() {
        super.onDreamingStarted();
        setFullscreen(true);
        setScreenBright(false);
        setContentView(R.layout.fullscreen_clock);
    }
}
```

```xml
<!-- here we go, this is it. are you ready? -->
<FrameLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent" >
    <TextClock
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:fontFamily="monospace"
        android:format12Hour="hh:mm"
        android:textColor="#CC3300"
        android:textSize="100sp" />
</FrameLayout>
```

12:46

but wait!

the colon
needs to blink

I said no code

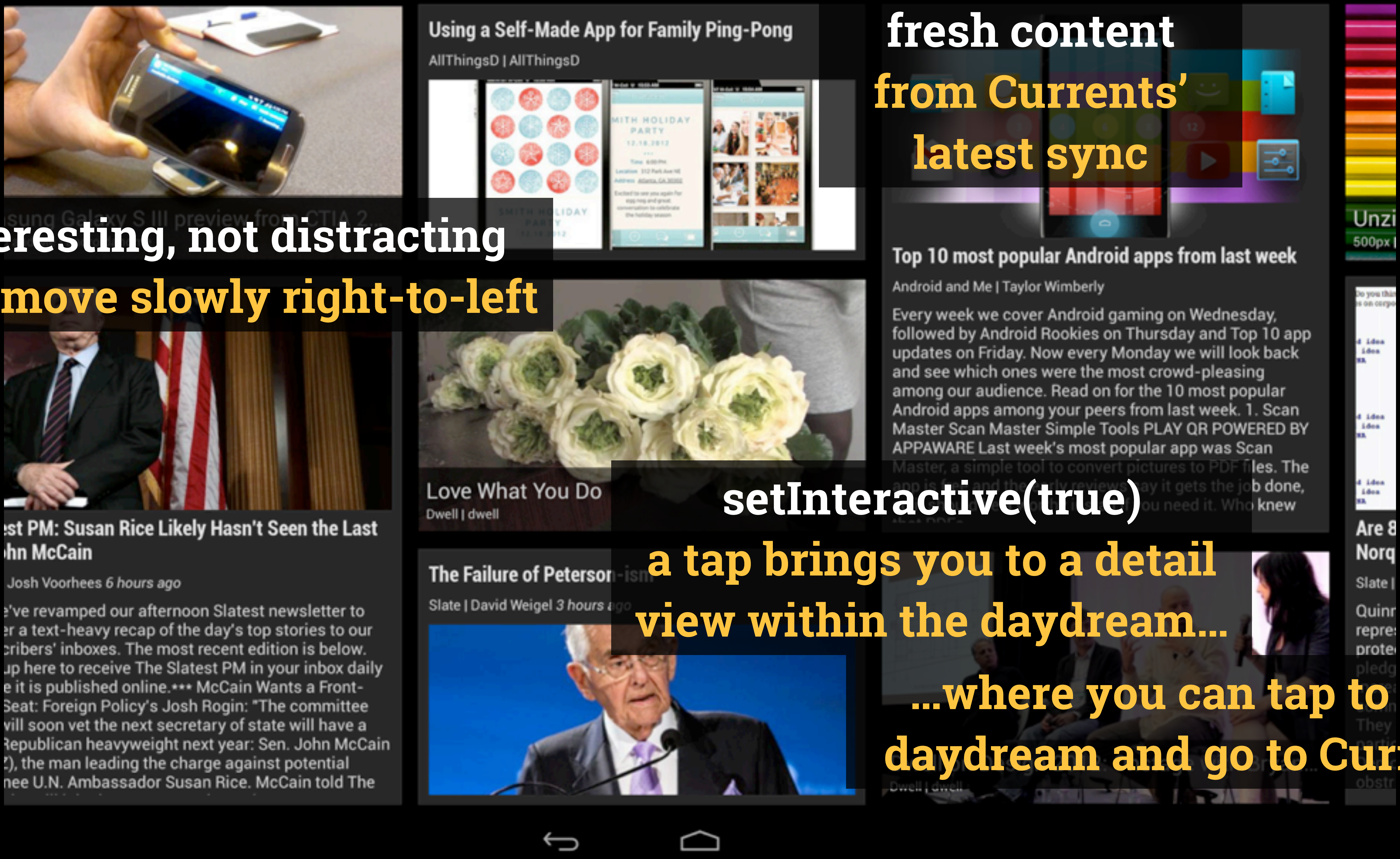
```
<FrameLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent" >
    <TextClock
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:fontFamily="monospace"
        android:format12Hour="hh mm"
        android:textColor="#CC3300"
        android:textSize="100sp" />
    <blink
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_gravity="center" >
        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="center"
            android:fontFamily="monospace"
            android:text="  :  "
            android:textColor="#CC3300"
            android:textSize="100sp"/>
    </blink>
</FrameLayout>
```

WAT

12:46

more examples

Google Currents



Using a Self-Made App for Family Ping-Pong

AllThingsD | AllThingsD

sung Galaxy S III preview from CTIA 2

Top 10 most popular Android apps from last week

Android and Me | Taylor Wimberly

Every week we cover Android gaming on Wednesday, followed by Android Rookies on Thursday and Top 10 app updates on Friday. Now every Monday we will look back and see which ones were the most crowd-pleasing among our audience. Read on for the 10 most popular Android apps among your peers from last week. 1. Scan Master Scan Master Simple Tools PLAY QR POWERED BY APPAWARE Last week's most popular app was Scan Master, a simple tool to convert pictures to PDF files. The app is f and th y it gets the job done, ou need it. Who knew

Love What You Do

Dwell | dwell

st PM: Susan Rice Likely Hasn't Seen the Last hn McCain

Josh Voorhees 6 hours ago

've revamped our afternoon Slatest newsletter to r a text-heavy recap of the day's top stories to our cribers' inboxes. The most recent edition is below. up here to receive The Slatest PM in your inbox daily e it is published online.*** McCain Wants a Front-Seat: Foreign Policy's Josh Rogin: "The committee vill soon vet the next secretary of state will have a Republican heavyweight next year: Sen. John McCain), the man leading the charge against potential nee U.N. Ambassador Susan Rice. McCain told The

The Failure of Peterson-ism

Slate | David Weigel 3 hours ago

Unzi

500px

Are 8 Norq

Slate

Quinn repre prote pledg

They

obstr

Dwell | dwell

fresh content

from Currents' latest sync

interesting, not distracting

tiles move slowly right-to-left

setInteractive(true)

a tap brings you to a detail view within the daydream…

…where you can tap to exit daydream and go to Currents

Beautiful Widgets

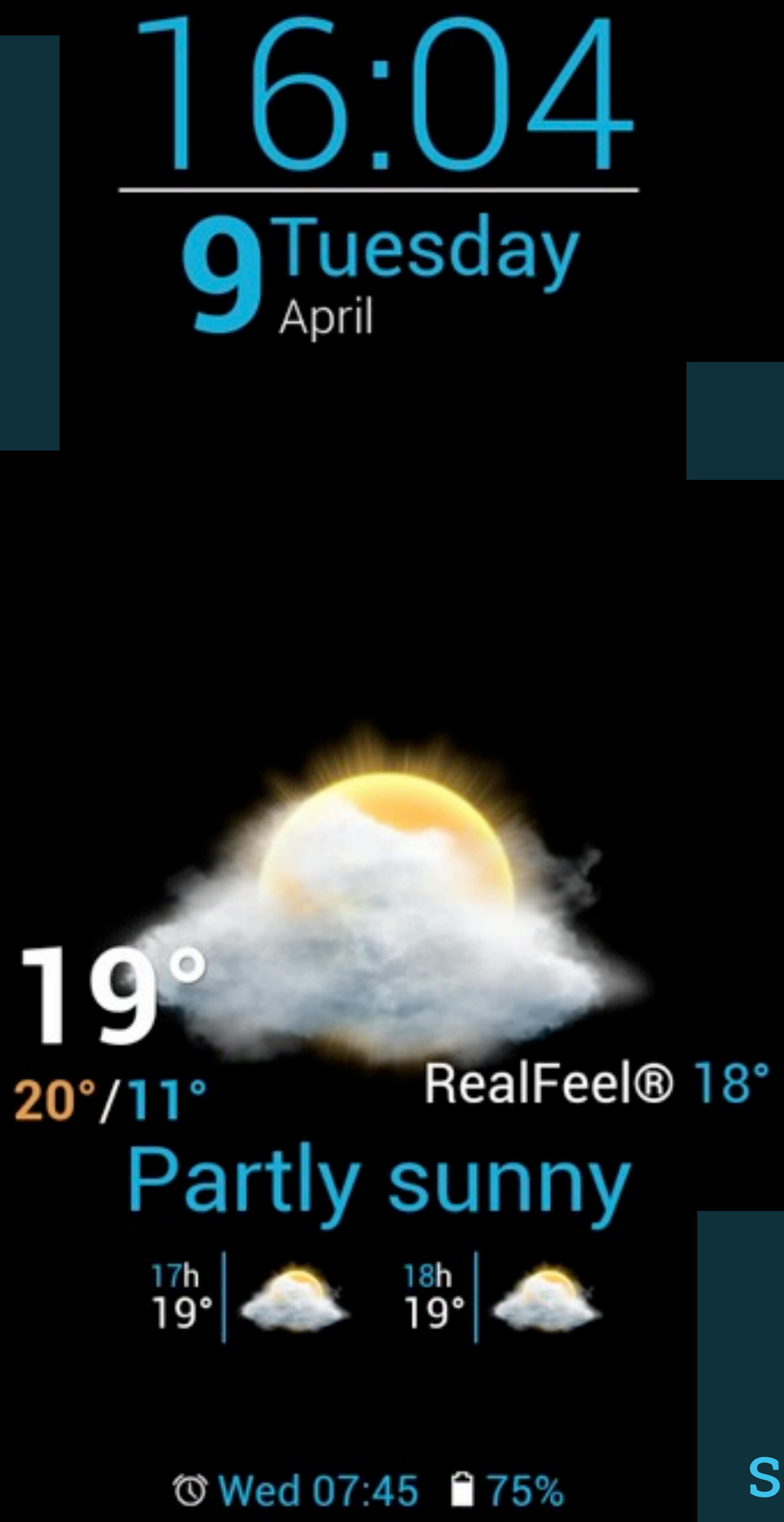
glanceable
what you need to know first thing in the morning (or late at night)

live data

weather effects

visual, artistic UI

handy battery info (could also show the status bar in lights out)



Flipboard

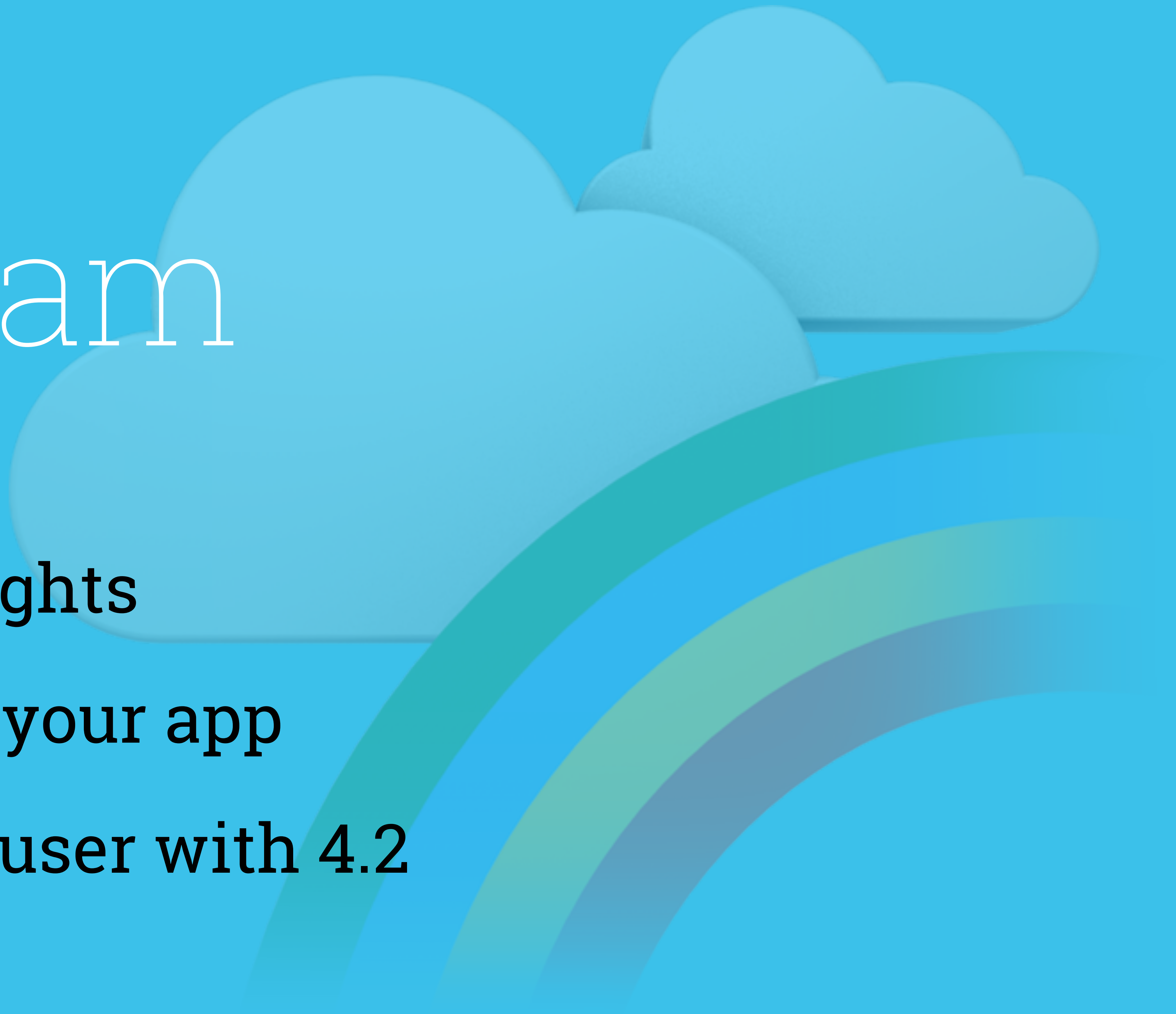
Flipboard

PLUS
VIVEKJENA
YOUCEF BENDRAOU
ERWAN MAITRE
CECILIA DOAN
& MORE

Cover Stories
Out for a Sunday drive.
Deanne Chen / www.flickr.com

"nearly 50% of Daydream sessions result in a Flipboard app launch" —

so.



daydream

easy to write

attracts and delights

a little break for your app

available to any user with 4.2

special thanks

Android Dream Team

Jeff Brown, Rachel Garb, Tom Karlo, John Spurlock, Chris Wren

Flipboard

LevelUp Studio

Reto Meier, Android DevRel, the Google I/O team

Cameos: Dianne, Matias, Baron

questions

Thanks for watching!

#io13 attendees: See you at office hours, right outside the talk

Code from this talk: https://code.google.com/p/android-daydream-samples/

Email	dsandler@google.com
Google+	dsandler.org/+
Elsewhere	dsandler

Google
Developers