TRY
ANDROID

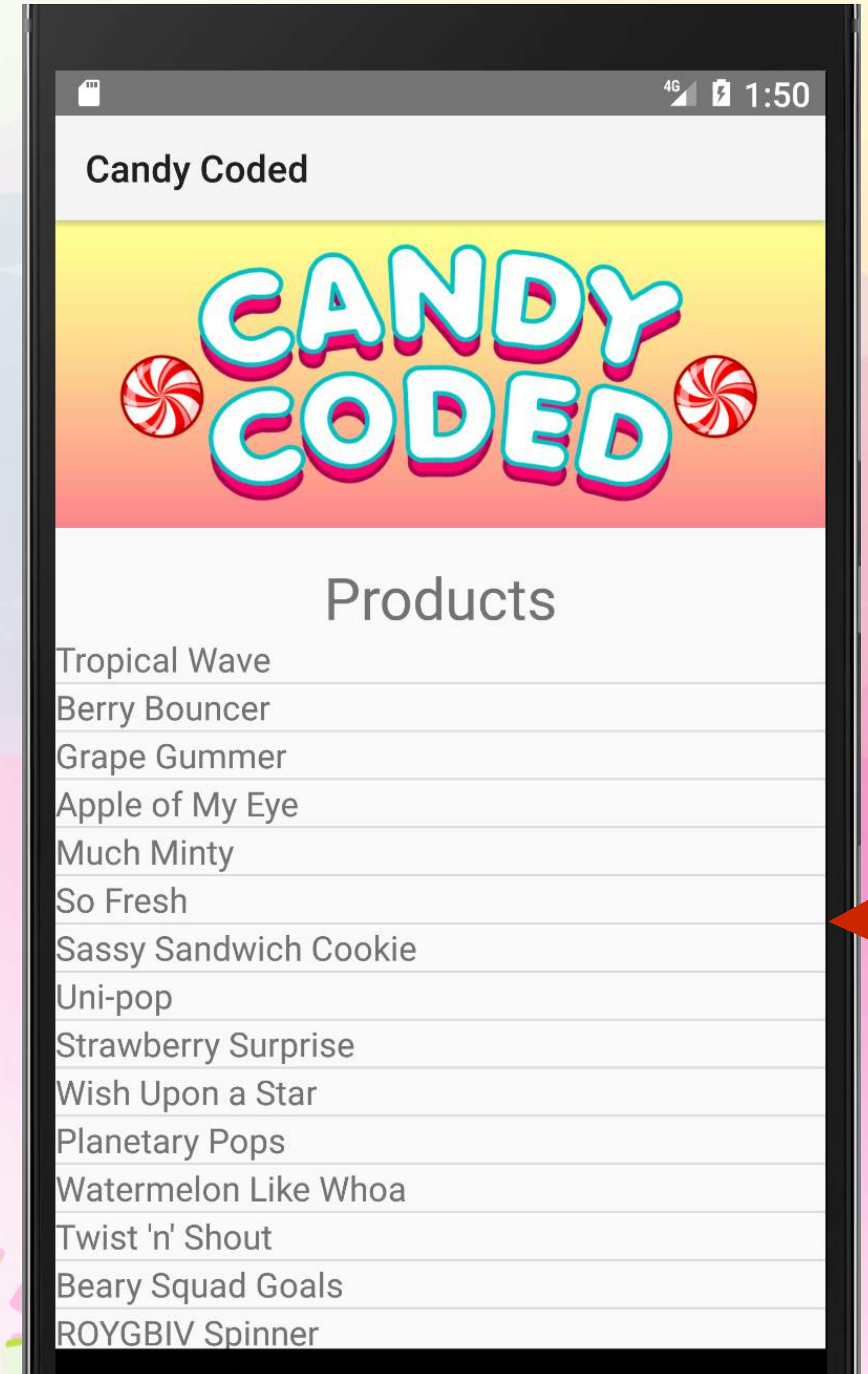
Level 4

EventListeners & Toasts

Making Our ListView Do Something

Improving the ListView's Look

We want to make the ListView look better by adding padding and updating the font and text size for the list items.



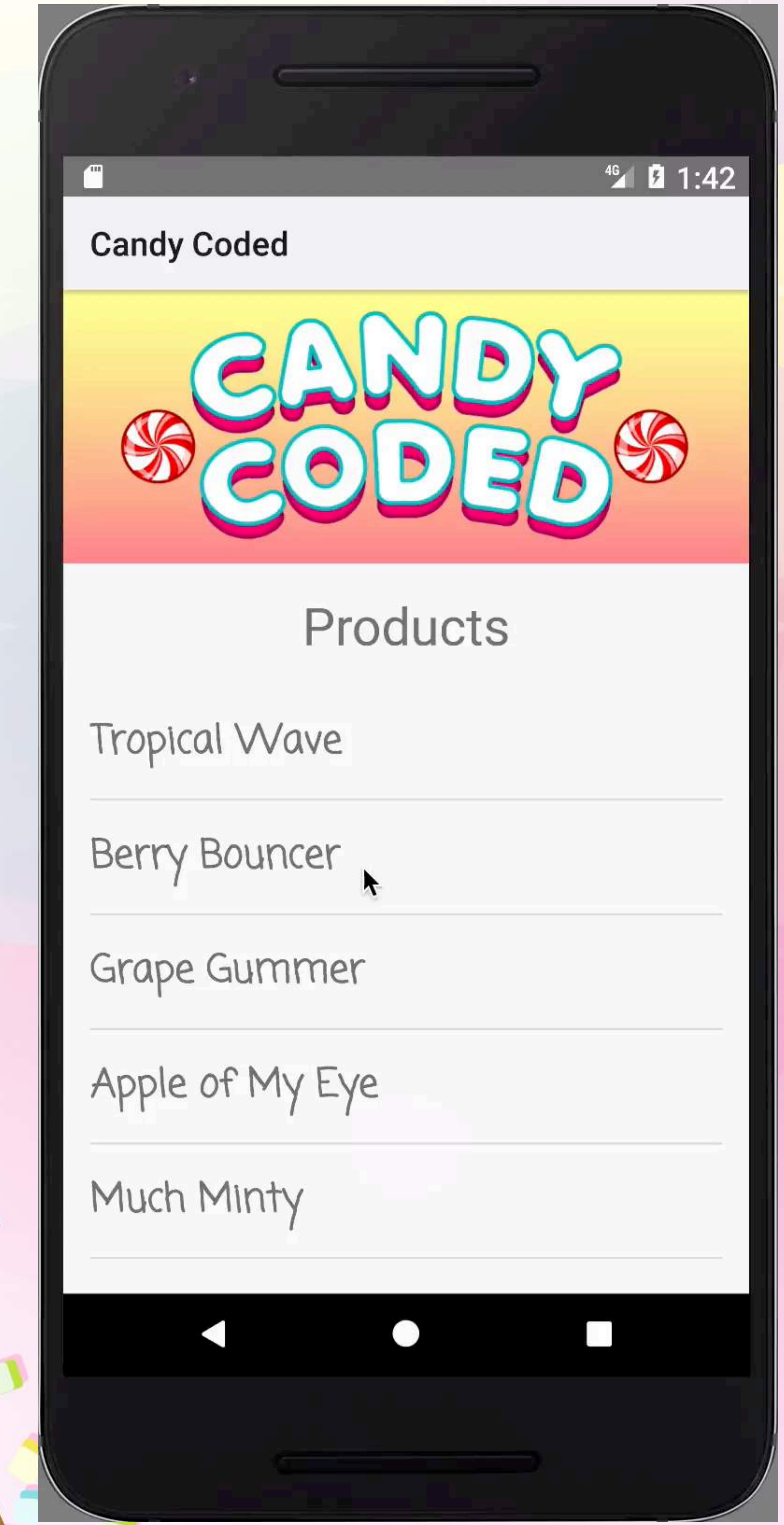
```
Candy Coded

Products
Tropical Wave
Berry Bouncer
Grape Gummer
Apple of My Eye
Much Minty
So Fresh
Sassy Sandwich Cookie
Uni-pop
Strawberry Surprise
Wish Upon a Star
Planetary Pops
Watermelon Like Whoa
Twist 'n' Shout
Beary Squad Goals
ROYGBIV Spinner
```

The `ListView` *and its items need padding*

Screencast: Adding Padding to the ListView

Making Our List Items Do Something

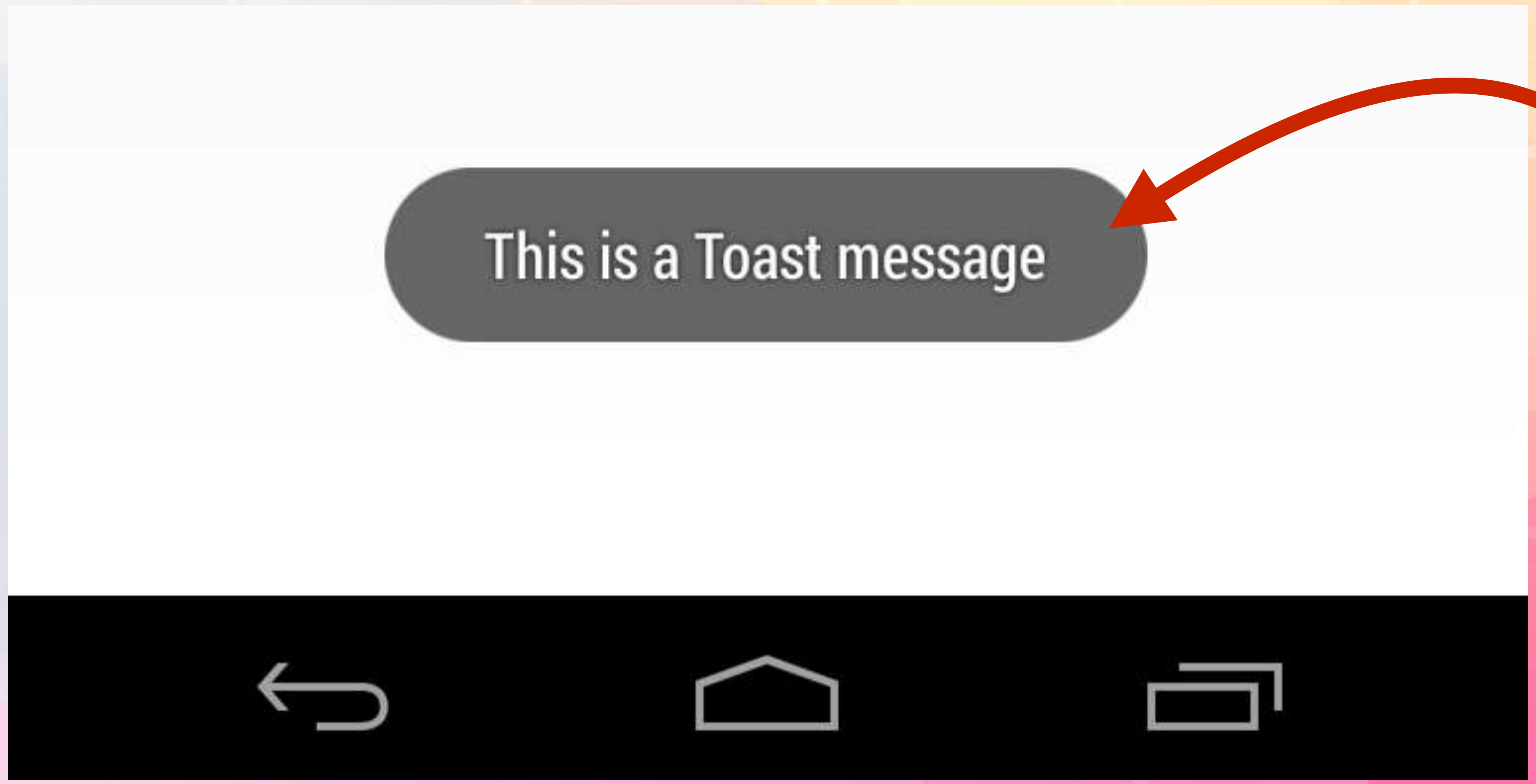


Eventually, we'd like to open a detail screen when the user clicks on a candy

As a first step, we'll display the position of the item in the list when the user clicks on it

Let’s Display a Toast

A Toast is a small popup message, and the current activity remains visible and interactive.



We can still see and interact with the Activity *below the* Toast

Creating Our First Toast

First, let's create a simple Toast when our app loads, and then we'll add one for clicks on the list items.

MainActivity.java

```java
...
public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        ...

        listView.setAdapter(adapter);

        // After connecting the ListView and Adapter


    }
}
```

How to Make a Toast

To create a Toast, we need to know the Context, text, and duration.

MainActivity.java

```
...
    Context context = this;
    String text = "Hello toast!";
    int duration = Toast.LENGTH_SHORT;

    Toast toast = Toast.makeText(context, text, duration);
    toast.show();
...
```

Creates the Toast

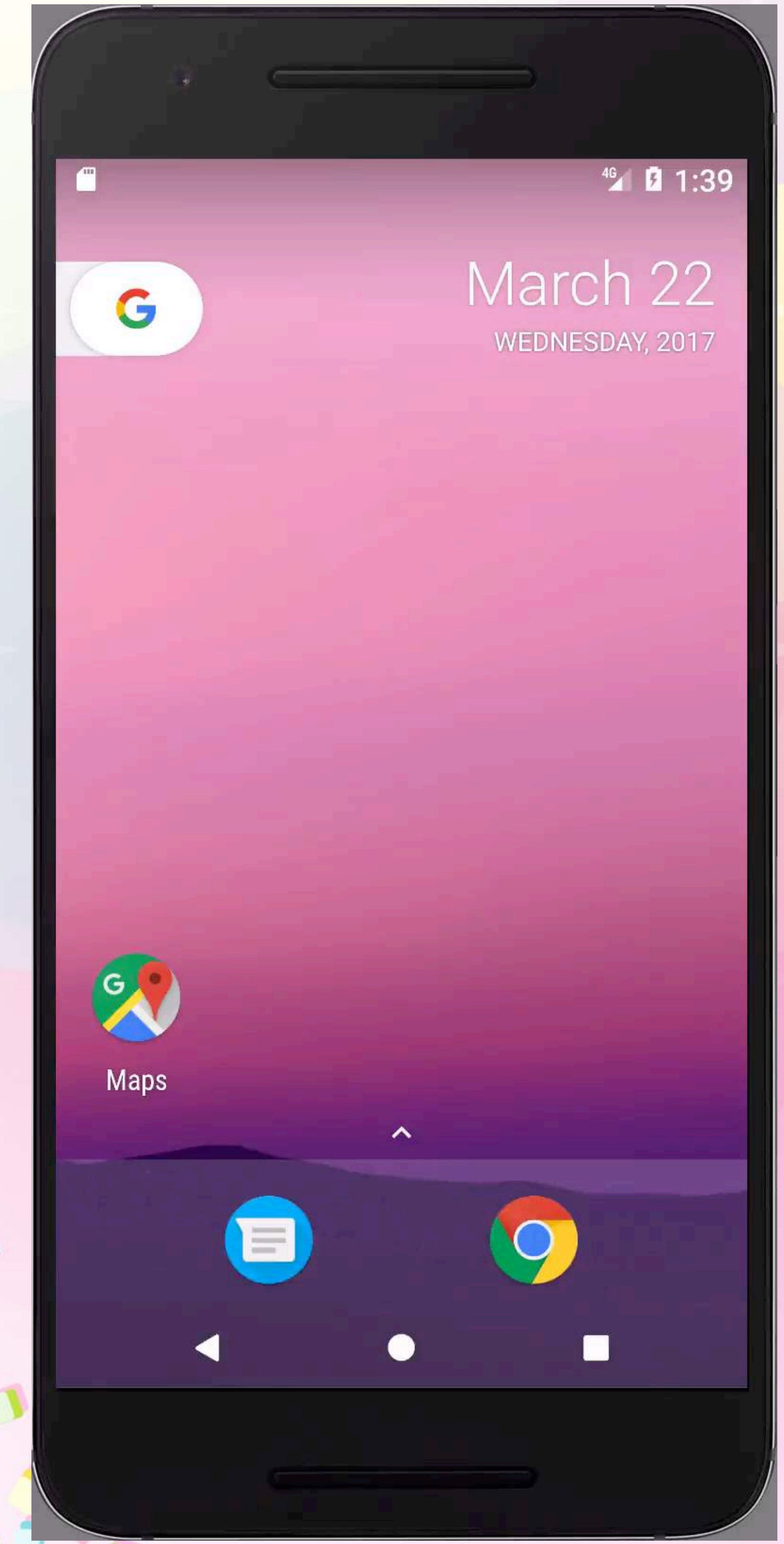
Displays the Toast

The context *is where this* Toast *will run, which for us is the* MainActivity.
So we can just use this *since we're in the* MainActivity *class.*

The text *is whatever message you want to display*

The duration *is either* Toast.LENGTH_SHORT *or* Toast.LENGTH_LONG

Demo of Our First Toast

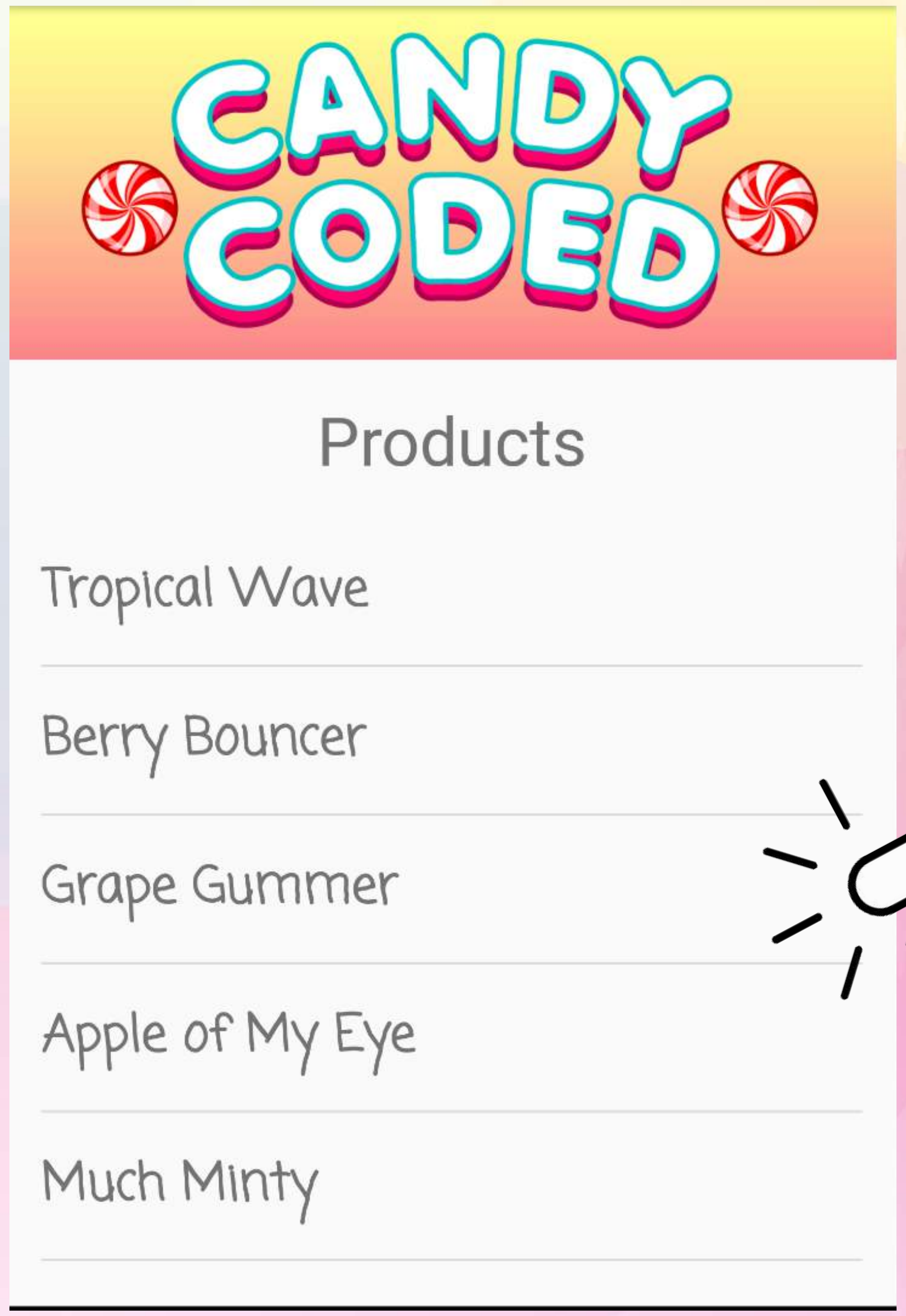


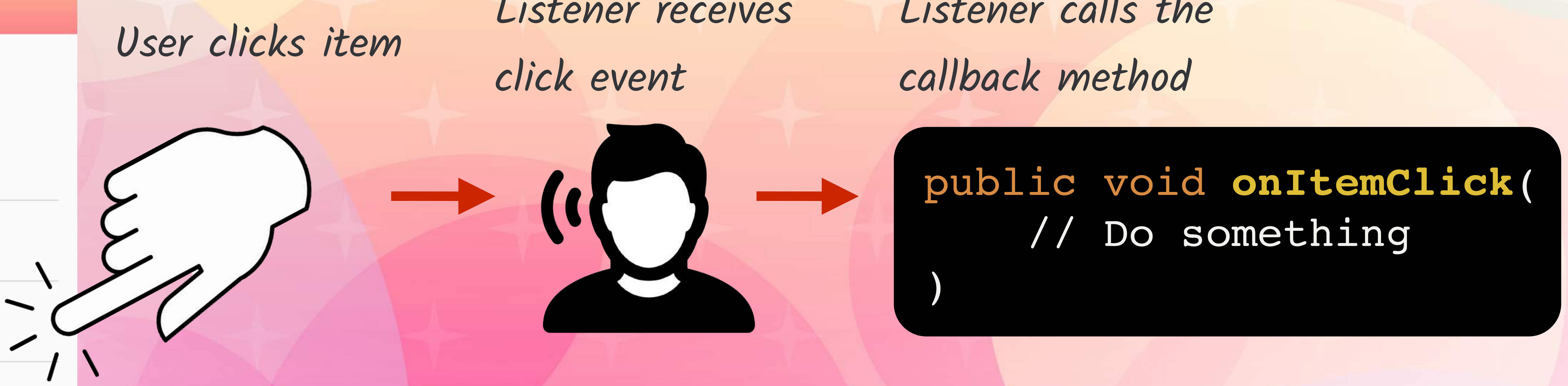
We can see our "Hello Toast!" Toast **comes up when the app starts.**

Now we want to show a `Toast` *when a list item is clicked!*

EventListeners

To trigger a Toast **on a click event, we can use an** EventListener**. An** EventListener **contains a single callback method, which gets called when the user triggers the event.**





User clicks item
Listener receives click event
Listener calls the callback method

```
public void onItemClick(
    // Do something
)
```

Adding an EventListener

An EventListener **contains a single callback method, which gets called when the user triggers the event. In this case,** onItemClick() **will get called when the user touches an item in the list.**

Our `ListView`   *Set its* `ItemClickListener`

```
listView.setOnItemClickListener(




);
```

Adding an EventListener

An EventListener **contains a single callback method, which gets called when the user triggers the event. In this case,** onItemClick() **will get called when the user touches an item in the list.**

Setting the item's click listener a `new ItemClickListener`

```
listView.setOnItemClickListener(new AdapterView.OnItemClickListener() {




});
```

Adding an EventListener

An EventListener **contains a single callback method, which gets called when the user triggers the event. In this case,** onItemClick() **will get called when the user touches an item in the list.**

```
listView.setOnItemClickListener(new AdapterView.OnItemClickListener() {
    @Override
    public void onItemClick(AdapterView<?> adapterView, View view,
                            int i, long l) {

        // Do something here

    }
});
```

This is the callback method that gets called when the user touches an item in our list

Adding an EventListener

We'll add our EventListener **code to the bottom of our** onCreate() **method again.**

MainActivity.java

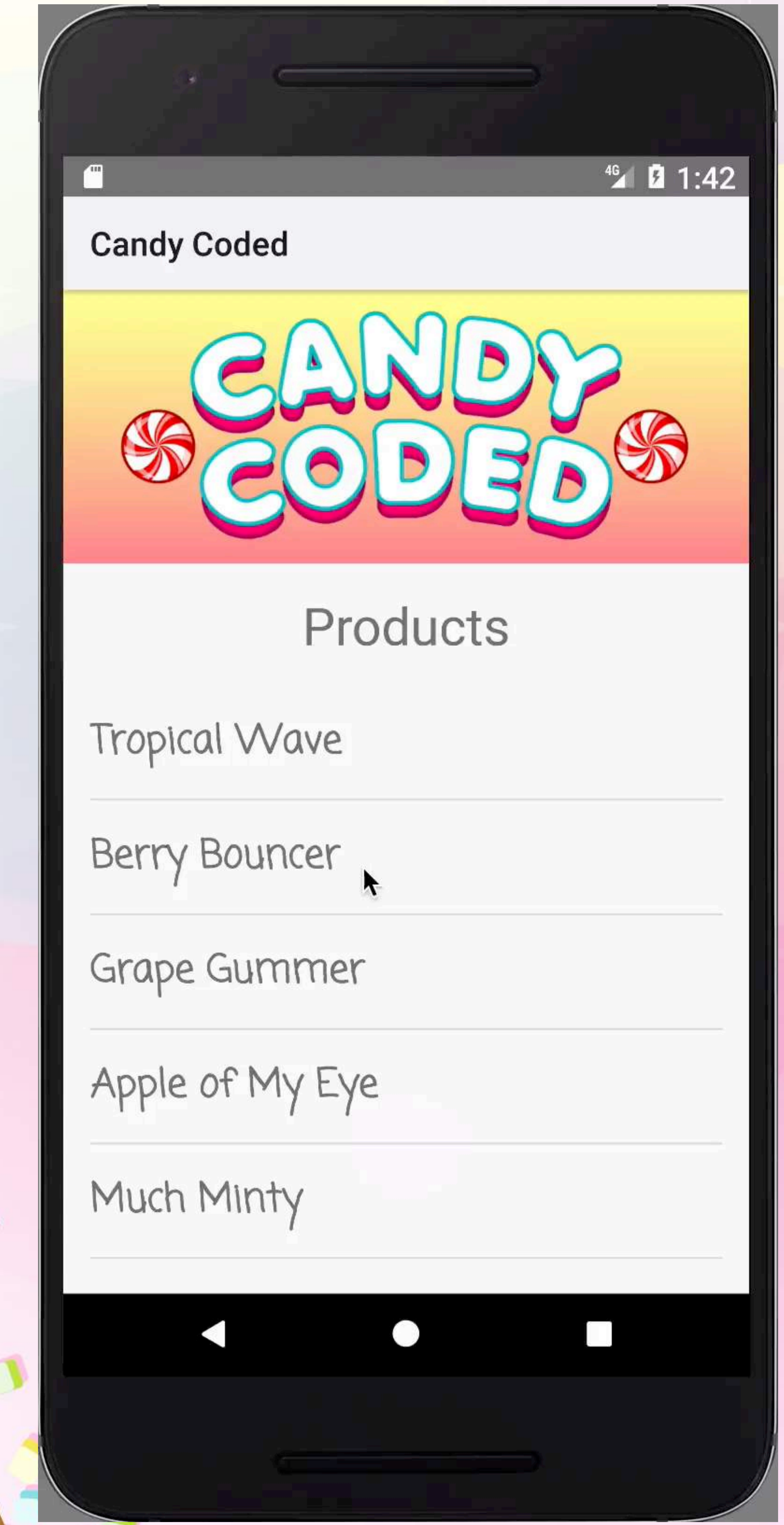
```
...
listView.setOnItemClickListener(new AdapterView.OnItemClickListener() {
    @Override
    public void onItemClick(AdapterView<?> adapterView, View view,
                                        int i, long l) {

    Toast toast = Toast.makeText(MainActivity.this, ""+i, Toast.LENGTH_SHORT);
    toast.show();
    }
});
...
```

Let's add a `Toast` *to show the item's position* `i`

We need to specify `MainActivity.this` *as the context, instead of just* `this`*, since we're inside the* `ClickListener`

Demoing the ClickListener **&** Toast **in Action**


1:42

Candy Coded

Products

Tropical Wave

Berry Bouncer

Grape Gummer

Apple of My Eye

Much Minty

Now we can see our list items actually doing something!

TRY
ANDROID