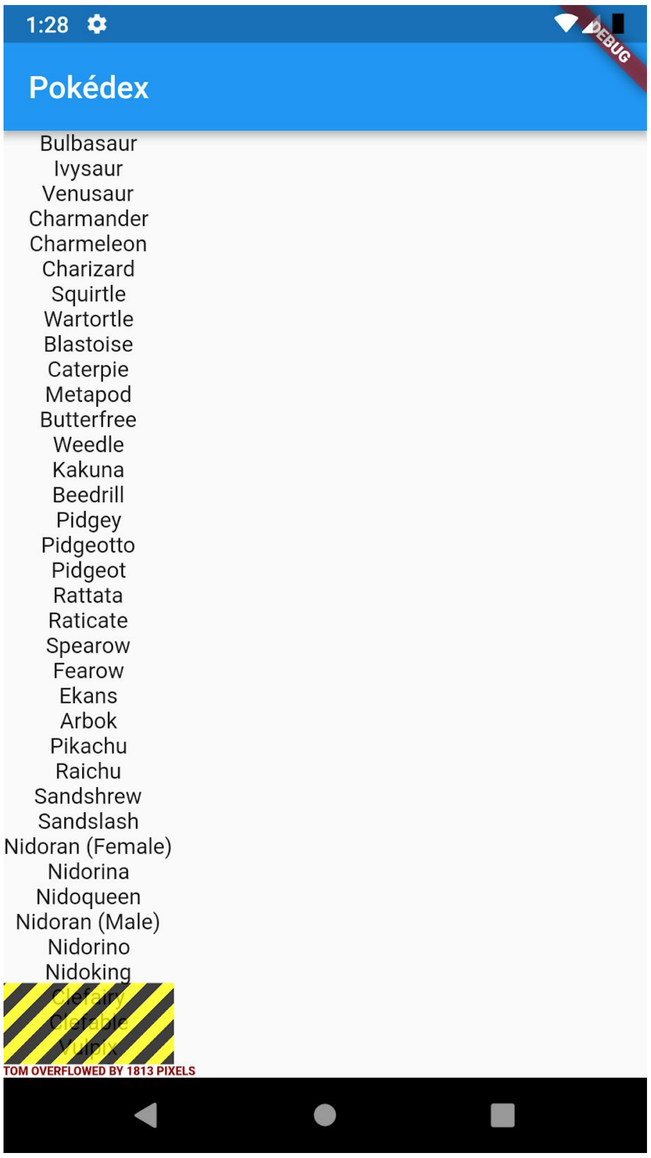
Build your Pokédex

This is your Pokédex now.

It isn't pretty and you can't scroll.

Let's fix that.



Pokédex

Bulbasaur
Ivysaur
Venusaur
Charmander
Charmeleon
Charizard
Squirtle
Wartortle
Blastoise
Caterpie
Metapod
Butterfree
Weedle
Kakuna
Beedrill
Pidgey
Pidgeotto
Pidgeot
Rattata
Raticate
Spearow
Fearow
Ekans
Arbok
Pikachu
Raichu
Sandshrew
Sandslash
Nidoran (Female)
Nidorina
Nidoqueen
Nidoran (Male)
Nidorino
Nidoking
Clefairy
Clefable
Vulpix

TOM OVERFLOWED BY 1813 PIXELS

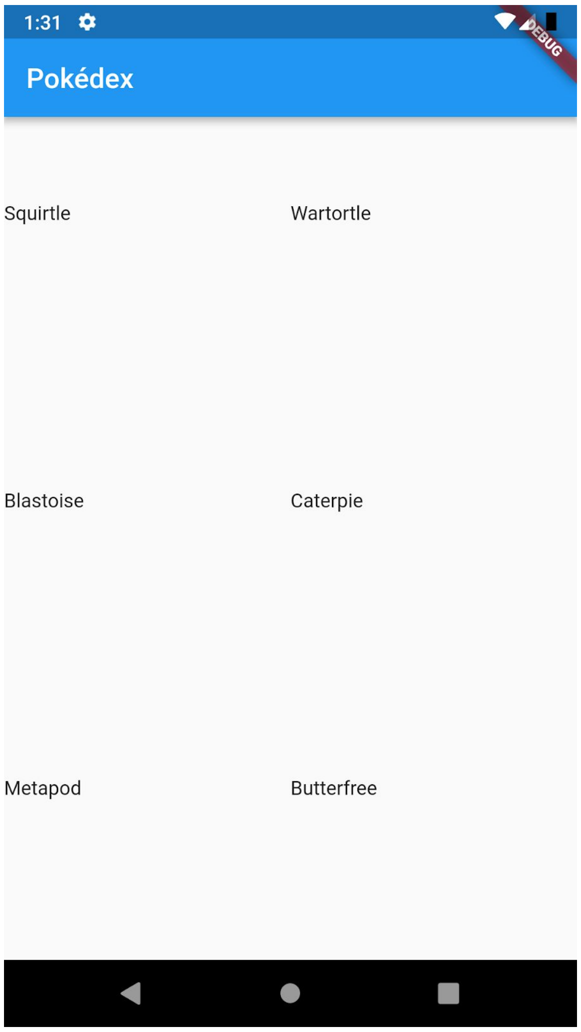
1) Using a GridView

Use a GridView to enable scrolling and show the name of the Pokémons in a grid.

Have a look at the GridView cookbook for inspiration.

You can also have a look at the API documentation, but it is quite extensive.

The final result should look something like what you see on the right.



2) Your first widget

We want to show more than just the name of the Pokémon. Let's start by creating a widget for that.

Create a stateless widget called `PokemonGridItem`.

The widget should take a `Pokemon` as a constructor argument and save it in a property on the class.

Use the `Text` widget to show the name of the Pokémon.

Use your `PokemonGridItem` in the GridView.

An example of a basic stateless widget:

```
class MyWidget extends StatelessWidget {
  ...

  @override
  Widget build(BuildContext context) {
    return Text(...);
  }
}
```

3) Showing images

Use the `Column` and `CachedNetworkImage` widgets to show the Pokémon images.

Have a look at the Image cookbook for inspiration.

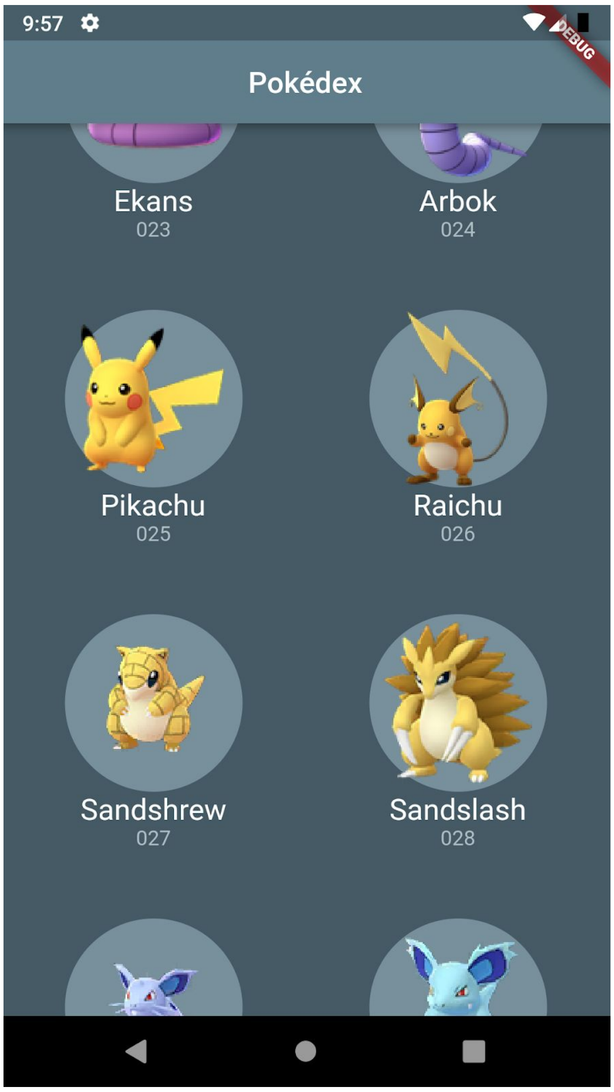
Hint: The `cached_network_image` package is already installed in the project.

Go crazy with styling.

Try using the `Container` widget to add some padding and colors.

Have a look at the widget catalog for inspiration.

The final result could look something like what you see on the right.



4) Showing details

The Pokemon class contains more information than we can show in the grid view.

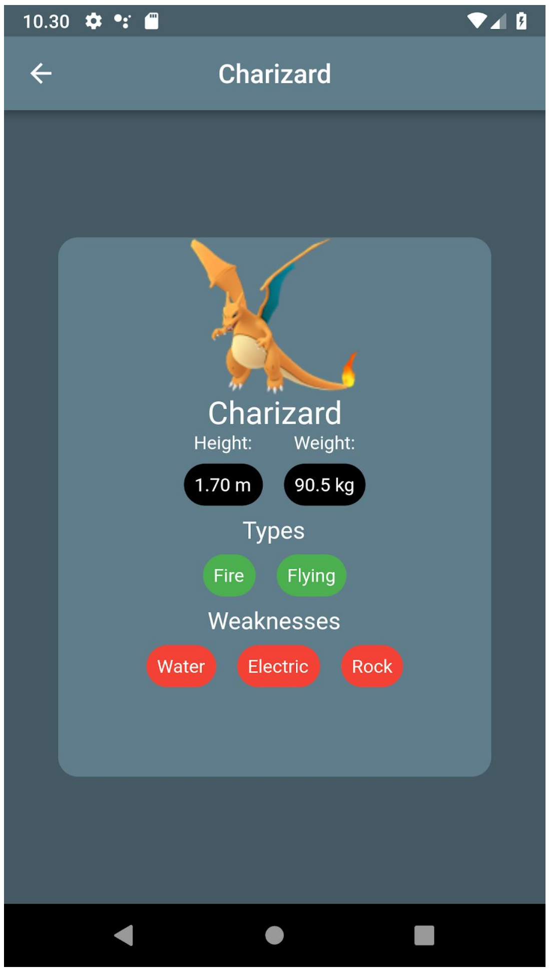
Let's create a screen where we can show more details such as the height, weight and weaknesses of the Pokémon.

Create a stateless widget called DetailsScreen. It should get a Pokemon just like the PokemonGridItem and show all its details.

The Pokédex should open this screen when you tap on a Pokémon in the grid view.

Use the GestureDetector widget for detecting taps. Have a look at the GestureDetector cookbook for inspiration.

Use the Navigator to navigate to the details screen on tap. Have a look at the Navigator cookbook step 2 for inspiration.



5) Animations

To add a hero animation to your pokémon images, wrap the images in a Hero widget.

Give the Hero widget a unique tag, like the pokémons name or number.

```
Hero(
  child: CachedNetworkImage(imageUrl: pokemon.image),
  tag: pokemon.number,
),
```

IT MINDS

Extra improvements

Show evolutions

The API also exposes evolutions. Update the `DetailsScreen` to support evolutions by showing both name and image of the evolutions. You will need to update your model.

An example from the API:

```
"prev_evolution": [
  {
    "number": "001",
    "name": "Bulbasaur"
  }
],
"next_evolution": [
  {
    "number": "003",
    "name": "Venusaur"
  }
]
```

Catch Pokémons

Make it possible to mark which pokémons you have caught.

You could implement this by adding a “onLongPress” callback to the `GestureDetector`, and use a stateful widget to store the list.

The `PokemonGridItem` should also be edited, so it reflects the changes by showing some kind of icon on the caught pokémons.

Search field

There are a lot of pokémons in the list. A useful feature would be to search for names. Implement a search feature on the homepage.

You could also implement filtering by type or sorting by height.

Use the Provider package

Before using the Provider package we need to add it to the pubspec.yaml file. This is done by opening the pubspec.yaml file, and adding provider to the dependencies. Adding the dependency should end up with a pubspec.yaml looking like this:

```
# ...
dependencies:
  flutter:
     sdk: flutter
  http: ^0.12.0+1
  cached_network_image: ^2.0.0
  provider: ^4.0.4
# ...
```

Saving the file will install the package. If this does not happen, run `flutter pub get` from the terminal.

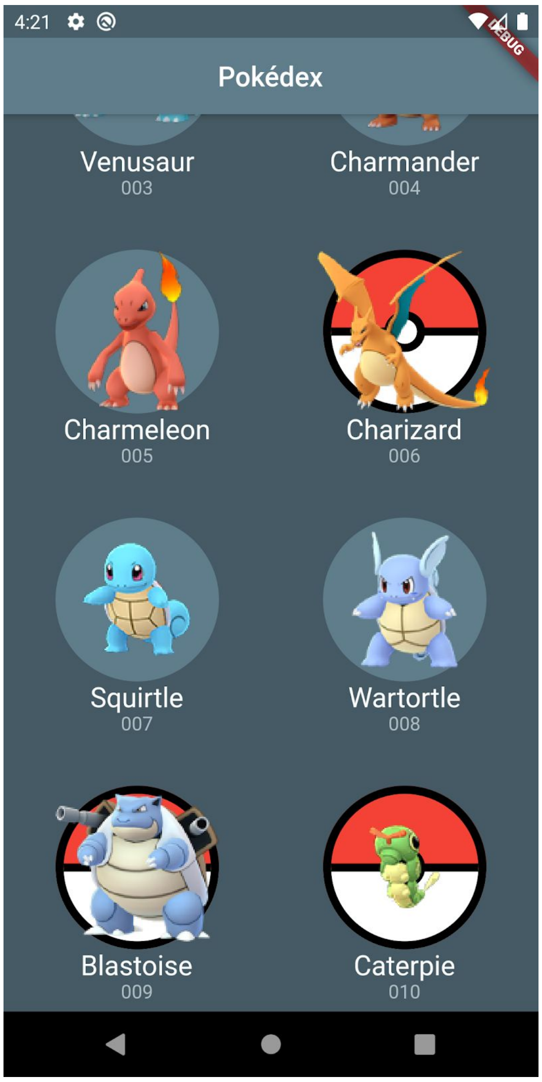
Having installed the package, you are now ready to use Provider. If you managed to solve the "Catch Pokémons" assignment, you now have a list in a `StatefulWidget` containing caught pokémons. This is not ideal, since it can be hard to reach that list from other widgets. So let's move the list to a `ChangeNotifier`.

Make a class that extends `ChangeNotifier` and move the list to the new class. In that class implement functions to modify the list.

Hint: Remember to provide the `ChangeNotifier` in the root of the app using the `MultiProvider` widget, and use a `Consumer` to access the class.

Draw a Pokéball

Use the [CustomPaint] widget and a [CustomPainter] to draw a Pokéball behind each caught pokemon.


4:21
DEBUG
Pokédex
Venusaur
003
Charmander
004
Charmeleon
005
Charizard
006
Squirtle
007
Wartortle
008
Blastoise
009
Caterpie
010