```dart
import 'dart:async';

import 'package:flutter/material.dart';
import 'package:flutter_bluetooth_serial/flutter_bluetooth_serial.dart';

import 'package:puttrv5/BTDevices.dart';

class SelectDevice extends StatefulWidget {
  final bool FindAvailable;

  const SelectDevice({this.FindAvailable = true});

  @override
  _SelectDevice createState() => new _SelectDevice();
}

enum _DeviceFound {
  no,
  maybe,
  yes,
}

class _DeviceisAvailable {
  BluetoothDevice device;
  _DeviceFound available;
  int? rssi;

  _DeviceisAvailable(this.device, this.available, [this.rssi]);
}

class _SelectDevice extends State<SelectDevice> {
  List<_DeviceisAvailable> devices =
  List<_DeviceisAvailable>.empty(growable: true);

  // available
  StreamSubscription<BluetoothDiscoveryResult>? _FindSubscription;
  bool _isDiscovering = false;

  _SelectDevice();

  @override
  void initState() {
    super.initState();

    _isDiscovering = widget.FindAvailable;

    if (_isDiscovering) {
      _startDiscovery();
    }

    // Setup a list of the bonded devices
    FlutterBluetoothSerial.instance
        .getBondedDevices()
        .then((List<BluetoothDevice> bondedDevices) {
      setState(() {
```

```
      devices = bondedDevices
          .map(
            (device) => _DeviceisAvailable(
          device,
          widget.FindAvailable
              ? _DeviceFound.maybe
              : _DeviceFound.yes,
        ),
      )
          .toList();
    });
  });
}

void _restartDiscovery() {
  setState(() {
    _isDiscovering = true;
  });

  _startDiscovery();
}

void _startDiscovery() {
  _FindSubscription =
      FlutterBluetoothSerial.instance.startDiscovery().listen((r) {
        setState(() {
          Iterator i = devices.iterator;
          while (i.moveNext()) {
            var _device = i.current;
            if (_device.device == r.device) {
              _device.available = _DeviceFound.yes;
              _device.rssi = r.rssi;
            }
          }
        });
      });

  _FindSubscription?.onDone(() {
    setState(() {
      _isDiscovering = false;
    });
  });
}

@override
void dispose() {
  // Avoid memory leak
  _FindSubscription?.cancel();

  super.dispose();
}

@override
Widget build(BuildContext context) {
  List<BTDevices> list = devices
```

```
        .map((_device) => BTDevices(
      device: _device.device,
      rssi: _device.rssi,
      enabled: _device.available == _DeviceFound.yes,
      onTap: () {
        Navigator.of(context).pop(_device.device);
      },
    ))
        .toList();
    return Scaffold(
      appBar: AppBar(
        title: Text('Select PUTR device'),
        actions: <Widget>[
          _isDiscovering
              ? FittedBox(
            child: Container(
              margin: new EdgeInsets.all(16.0),
              child: CircularProgressIndicator(
                valueColor: AlwaysStoppedAnimation<Color>(
                  Colors.white,
                ),
              ),
            ),
          )
              : IconButton(
            icon: Icon(Icons.replay),
            onPressed: _restartDiscovery,
          )
        ],
      ),
      body: ListView(children: list),
    );
  }
}
```