

mutable data

```
add to lib/main.dart: This class
stores mutable data which could
change over the lifetime
class _FavoriteWidgetState
extends State<FavoriteWidget> {
  // ...
  @override
  Widget build(BuildContext context) {
    return Row(
      mainAxisAlignment:
MainAxisAlignment.min,
      children: [
        Container(
          padding:
EdgeInsets.symmetric(),
          child:
IconButton(
            icon:
(_isFavorited ? Icon(Icons.star) : Icon(Icons.star_border)),
            color:
Colors.red[500],
            onPressed: _toggleFavorite,
          ),
          SizedBox(
            width: 18,
            child:
Container(
              child:
Text('$_favoriteCount'),
            ),
          ),
        ],
      );
  }
}
```

tapbox

```
the tap box is managing its own
state
class TapboxA extends StatefulWidget {
  TapboxA({Key key}) :
super(key: key);
  @override
  _TapboxAState createState() => _TapboxAState();
}
class _TapboxAState extends
State<TapboxA> {
  bool _active = false;
  void _handleTap() {
    setState(() {
      _active = !_active;
    });
  }
  Widget build(BuildContext context) {
    return GestureDetector(
      onTap: _handleTap,
      child: Container(
        child: Center(
          child: Text(
            _active
? 'Active' : 'Inactive',
            style:
TextStyle(fontSize: 32.0,
color: Colors.white),
          ),
          width: 200.0,
          height: 200.0,
          decoration:
BoxDecoration(
            color:
_active ? Colors.lightGreen[
700] : Colors.grey[600],
          ),
        ),
      );
  }
}
```

tapbox (cont)

```
>}
```

parent widget

```
manages the activity state,
implements handle checkbox
class ParentWidget extends
StatefulWidget {
  @override
  _ParentWidgetState
createState() => _ParentWidgetState();
}
class _ParentWidgetState
extends State<ParentWidget> {
  bool _active = false;
  void _handleTapboxChanged(
    bool newValue) {
    setState(() {
      _active = newValue;
    });
  }
  @override
  Widget build(BuildContext context) {
    return Container(
      child: TapboxC(
        active: _active,
        onChanged:
_handleTapboxChanged,
      ),
    );
  }
}
```

creating API

When creating API, consider using the @required annotation for any parameters(which re-exports Dart's meta.dart library):	import 'package:flutter/foundation.dart';
---	---