



Министерство науки и высшего образования Российской Федерации  
Федеральное государственное бюджетное образовательное учреждение  
высшего образования  
«Московский государственный технический университет  
имени Н.Э. Баумана  
(национальный исследовательский университет)»  
(МГТУ им. Н.Э. Баумана)

---

ФАКУЛЬТЕТ \_\_\_\_\_ «Информатика и системы управления»

КАФЕДРА \_\_\_\_\_ «Теоретическая информатика и компьютерные технологии»

**Лабораторная работа № 2**  
**по курсу «Разработка мобильных приложений»**  
**«Реализация приложения на основе виджета»**

Студентка группы ИУ9-72Б Самохвалова П. С.

Преподаватель Посевин Д. П.

*Москва 2023*

# 1 Задание

Вариант 15.

Используя 26 виджет реализовать программу отправки электронных сообщений пользователям из списка, текст сообщения и тема сообщения вводятся из отдельных полей. Пользователь приложения вводит только имя или фамилию получателя, адрес электронной почты он видеть не должен.

## 2 Практическая реализация

Исходный код программы представлен в листинге 1.

Листинг 1: Отправка электронных сообщений

```
1 import 'dart:async';
2 import 'dart:io';
3
4 import 'package:flutter/material.dart';
5 import 'package:flutter_email_sender/flutter_email_sender.dart';
6 import 'package:image_picker/image_picker.dart';
7 import 'package:path_provider/path_provider.dart';
8
9 void main() => runApp(MyApp());
10
11
12 class MyApp extends StatelessWidget {
13   @override
14   Widget build(BuildContext context) {
15     return MaterialApp(
16       theme: ThemeData(primaryColor: Colors.red),
17       home: EmailSender(),
18     );
19   }
20 }
21
22 class EmailSender extends StatefulWidget {
23   const EmailSender({Key? key}) : super(key: key);
24
25   @override
26   _EmailSenderState createState() => _EmailSenderState();
27 }
28
29 class _EmailSenderState extends State<EmailSender> {
30   List<String> attachments = [];
```

```

31 bool isHTML = false;
32
33 final _nameController = TextEditingController(text: 'Receiver name');
34 final _recipientController = TextEditingController(text: '
    example@example.com');
35 final _subjectController = TextEditingController(text: 'The subject');
36 final _bodyController = TextEditingController(text: 'Mail body. ');
37
38 List<String> names = ['Polina', 'Dmitry', 'Name 3', 'Name 4', 'Name
    5'];
39 String? selectedItem = 'Polina';
40
41 Future<void> send() async {
42
43     if (selectedItem == "Polina") {
44         _recipientController.text = "sam53351@yandex.ru";
45         _nameController.text = "Polina";
46     } else if (selectedItem == "Dmitry") {
47         _recipientController.text = "dmitry@yandex.ru";
48         _nameController.text = "Dmitry";
49     }
50
51     final Email email = Email(
52
53         body: _bodyController.text,
54         subject: _subjectController.text,
55         recipients: [_recipientController.text],
56         attachmentPaths: attachments,
57         isHTML: isHTML,
58     );
59
60     String platformResponse;
61
62     try {
63         await FlutterEmailSender.send(email);
64         platformResponse = 'success';
65     } catch (error) {
66         print(error);
67         platformResponse = error.toString();
68     }
69
70     if (!mounted) return;
71
72     ScaffoldMessenger.of(context).showSnackBar(
73         SnackBar(
74             content: Text(platformResponse),

```

```

75     ),
76   );
77 }
78
79 @override
80 Widget build(BuildContext context) {
81   return Scaffold(
82     appBar: AppBar(
83       title: Text('Plugin example app'),
84       actions: <Widget>[
85         IconButton(
86           onPressed: send,
87           icon: Icon(Icons.send),
88         )
89       ],
90     ),
91     body: Padding(
92       padding: EdgeInsets.all(8.0),
93       child: Column(
94         mainAxisAlignment: MainAxisAlignment.max,
95         // mainAxisAlignment: MainAxisAlignment.spaceBetween,
96         crossAxisAlignment: CrossAxisAlignment.center,
97         children: <Widget>[
98
99           DropdownButton<String> (
100             value: selectedItem,
101             items: names
102               .map((name) => DropdownMenuItem<String>(
103                 value: name,
104                 child: Text(name, style: TextStyle(fontSize: 24)),
105               ))
106               .toList(),
107             onChanged: (name) => setState(() => selectedItem = name!),
108
109           ),
110
111           Padding(
112             padding: EdgeInsets.all(8.0),
113             child: TextField(
114               controller: _subjectController,
115               decoration: InputDecoration(
116                 border: OutlineInputBorder(),
117                 labelText: 'Subject',
118               ),
119             ),
120           ),

```

```

121
122 Expanded(
123   child: Padding(
124     padding: EdgeInsets.all(8.0),
125     child: TextField(
126       controller: _bodyController,
127       maxLines: null,
128       expands: true,
129       textAlignVertical: TextAlignVertical.top,
130       decoration: InputDecoration(
131         labelText: 'Body', border: OutlineInputBorder()),
132     ),
133   ),
134 ),
135 CheckboxListTile(
136   contentPadding:
137     EdgeInsets.symmetric(vertical: 0.0, horizontal: 8.0),
138   title: Text('HTML'),
139   onChanged: (bool? value) {
140     if (value != null) {
141       setState(() {
142         isHTML = value;
143       });
144     }
145   },
146   value: isHTML,
147 ),
148
149 Padding(
150   padding: EdgeInsets.all(8.0),
151   child: Column(
152     children: <Widget>[
153       for (var i = 0; i < attachments.length; i++)
154         Row(
155           children: <Widget>[
156             Expanded(
157               child: Text(
158                 attachments[i],
159                 softWrap: false,
160                 overflow: TextOverflow.fade,
161               ),
162             ),
163             IconButton(
164               icon: Icon(Icons.remove_circle),
165               onPressed: () => {_removeAttachment(i)},
166             )

```

```

167         ],
168     ),
169     Align(
170         alignment: Alignment.centerRight,
171         child: IconButton(
172             icon: Icon(Icons.attach_file),
173             onPressed: _openImagePicker,
174         ),
175     ),
176     TextButton(
177         child: Text('Attach file in app documents directory
178 '),
179         onPressed: () => _attachFileFromAppDocumentsDirectory
180     ),
181 ),
182 ),
183 ],
184 ),
185 ),
186 );
187 }
188
189 void _openImagePicker() async {
190     final picker = ImagePicker();
191     XFile? pick = await picker.pickImage(source: ImageSource.gallery);
192     if (pick != null) {
193         setState(() {
194             attachments.add(pick.path);
195         });
196     }
197 }
198
199 void _removeAttachment(int index) {
200     setState(() {
201         attachments.removeAt(index);
202     });
203 }
204
205 Future<void> _attachFileFromAppDocumentsDirectory() async {
206     try {
207         final appDocumentDir = await getApplicationDocumentsDirectory();
208         final filePath = appDocumentDir.path + '/file.txt';
209         final file = File(filePath);
210         await file.writeAsString('Text file in app directory');

```

```

211
212     setState(() {
213         attachments.add(filePath);
214     });
215 } catch (e) {
216     ScaffoldMessenger.of(context).showSnackBar(
217         SnackBar(
218             content: Text('Failed to create file in applicion directory'),
219         ),
220     );
221 }
222 }
223 }

```

### 3 Результаты

Результаты работы программы представлены на рисунках 1 – 4.

### 4 Выводы

В результате выполнения лабораторной работы была реализована программа отправки электронных сообщений пользователям из списка.

Plugin example app

Polina

Subject


The subject

Body

Mail body.

HTML

☐



Attach file in app documents directory

Рис. 1 — Форма ввода электронного сообщения



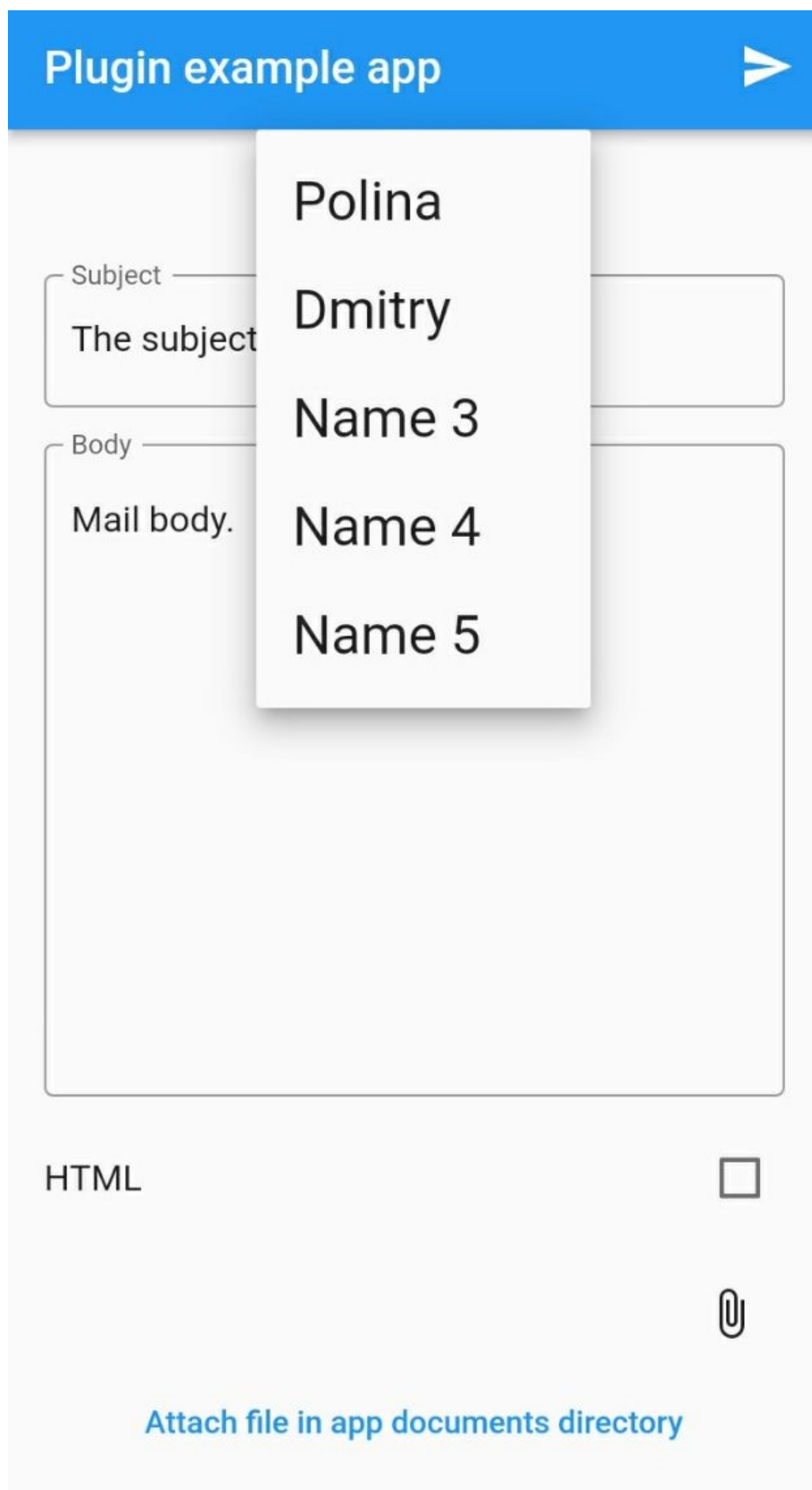


Рис. 2 — Выбор пользователя из списка

Plugin example app

Polina

Subject


The subject

Body

Hello!

HTML

☐



Attach file in app documents directory

Рис. 3 — Ввод текста письма

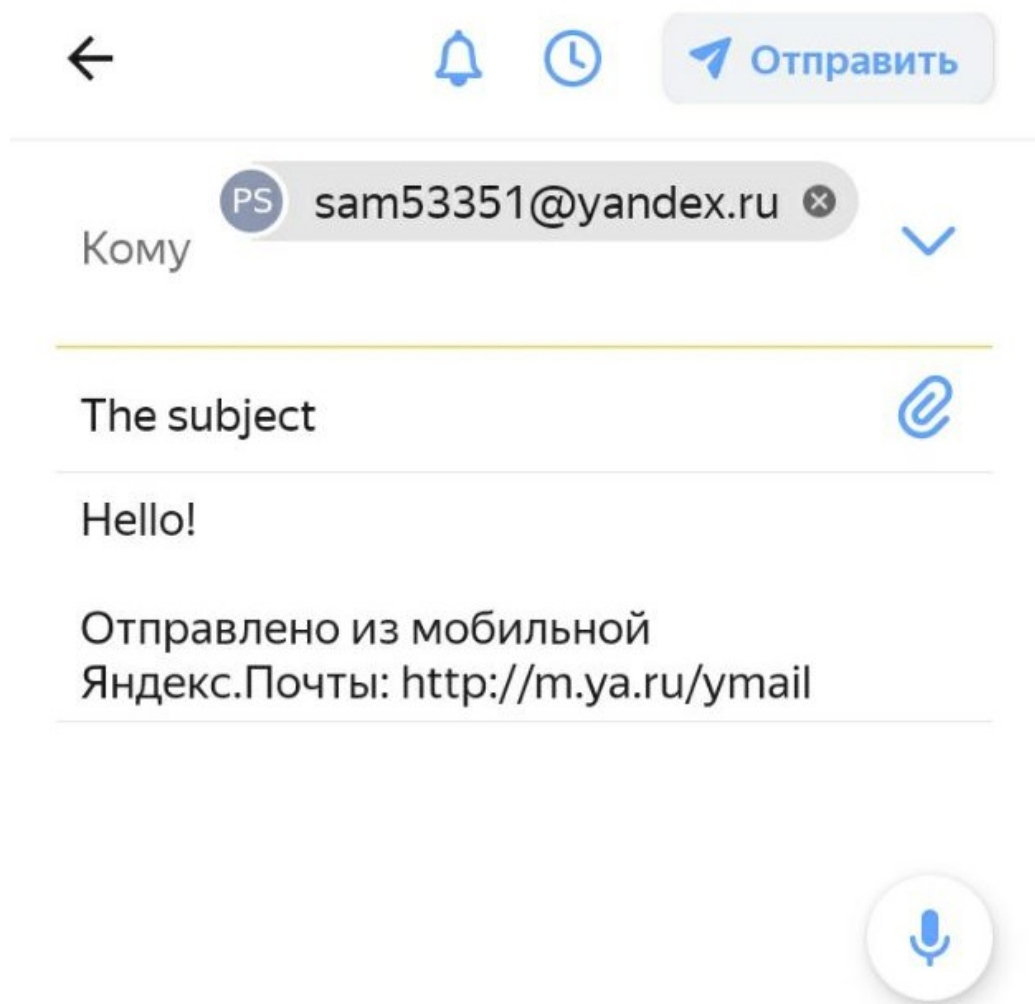


Рис. 4 — Отправка письма