**Практическое занятие 10. Работа с текстом и ресурсами**

**Цель:**

Познакомиться с возможностями работы с текстом и графическими ресурсами в приложениях Flutter, а также научиться запускать встроенные функции устройства, такие как вызов номера телефона.

**Необходимые библиотеки:**

1. **Flutter SDK** для работы с виджетами и построения пользовательского интерфейса.
2. **url\_launcher** для работы с внешними ресурсами (например, открытие ссылки или звонок по телефону).

**Проверка работы:**

1. Запустите проект.
2. Перейдите на экран "О приложении".
3. Нажмите на номер телефона — должно открыться стандартное приложение для вызова.

**Контрольные вопросы:**

1. Какие основные виджеты используются для работы с текстом в Flutter?
2. Как можно настроить стиль текста с помощью виджета TextStyle? Какие параметры можно изменить?
3. Что такое Container и как его можно использовать для оформления текста или других виджетов?
4. Как задать рамку и закруглённые углы для виджета в Flutter?
5. Как добавить изображение в проект Flutter и отобразить его на экране?
6. Где необходимо указать путь к изображениям в проекте Flutter?
7. Для чего используется виджет Scaffold и какие компоненты он может включать?
8. Как с помощью библиотеки url\_launcher вызвать номер телефона из приложения Flutter?
9. Что делает функция canLaunchUrl и для чего она используется?
10. Как добавить возможность открытия веб-ссылок в приложении Flutter?
11. Что такое виджет GestureDetector и как его можно использовать для обработки событий взаимодействия с пользователем?
12. Какие виджеты в Flutter используются для компоновки элементов вертикально и горизонтально?
13. Как можно задать отступы вокруг виджетов в Flutter?
14. Какие схемы URI можно использовать для взаимодействия с устройством (например, для звонков или открытия веб-ссылок)?
15. Как проверить, поддерживает ли устройство функцию звонка перед попыткой совершить вызов из приложения?

**Дополнительные задания:**

1. **Изменение стиля текста**
   1. Добавьте в экран "О приложении" ещё один текстовый блок с любым текстом. Настройте его стиль так, чтобы текст был красного цвета, размер шрифта был 18, а текст был подчеркивающимся. Используйте виджет Text и настройте стиль через TextStyle.
2. **Добавление новой иконки**
   1. В блоке с информацией о разработчике добавьте новую иконку (например, иконку электронной почты). При нажатии на неё приложение должно открывать почтовый клиент для отправки письма на заранее указанный адрес электронной почты. Используйте библиотеку url\_launcher и GestureDetector для обработки нажатия.
3. **Изменение оформления контейнера**
   1. Измените оформление контейнера, в котором находится описание приложения. Сделайте фон контейнера голубым, а текст внутри контейнера — белым. Также увеличьте ширину рамки до 3 пикселей и измените цвет рамки на красный.
4. **Создание кнопки для открытия веб-страницы**
   1. Добавьте кнопку в нижней части экрана, которая будет открывать любую веб-страницу в браузере (например, сайт вашей компании). Используйте виджет TextButton для создания кнопки и библиотеку url\_launcher для открытия веб-страницы.
5. **Добавление изображения из интернета**
   1. Вместо логотипа из локальных ресурсов добавьте изображение, загружаемое из интернета. Используйте виджет Image.network для загрузки изображения по URL. Убедитесь, что изображение отображается корректно и его размеры ограничены до 150x150 пикселей.

**Теоретическая часть**

**1. Текстовые виджеты в Flutter**

Текст — это ключевой элемент любого пользовательского интерфейса. В Flutter для отображения текста используется виджет Text. Его можно гибко настраивать с помощью TextStyle, который позволяет управлять стилем текста: изменять размер, цвет, толщину шрифта, декорирование и многое другое.

Пример простого текста:

Text(

'Пример текста',

style: TextStyle(

fontSize: 18,

color: Colors.black,

),

)

Основные параметры TextStyle:

1. **fontSize** — задаёт размер текста.
2. **fontWeight** — отвечает за толщину шрифта (например, FontWeight.bold для жирного текста).
3. **color** — задаёт цвет текста.
4. **decoration** — позволяет добавить подчеркивание или зачеркивание текста (например, TextDecoration.underline).

**2. Контейнеры для оформления текста**

Текст можно помещать в различные контейнеры (Container), чтобы задавать отступы, рамки, фон или другие декоративные элементы.

Пример использования Container для текста с рамкой и закруглёнными углами:

Container(

padding: EdgeInsets.all(12),

decoration: BoxDecoration(

border: Border.all(

color: Colors.teal,

width: 2.0,

),

borderRadius: BorderRadius.circular(10.0),

),

child: Text(

'Это текст внутри контейнера',

style: TextStyle(fontSize: 16, color: Colors.black),

),

)

**3. Работа с графическими ресурсами**

В приложениях часто используются изображения, такие как логотипы, фоны или иконки. В Flutter для отображения изображений из локальных файлов используется виджет Image.asset. Чтобы приложение могло корректно загружать изображения, их необходимо добавить в проект и указать путь в файле pubspec.yaml.

Пример загрузки изображения:

Image.asset(

'assets/images/logo.png',

width: 100,

height: 100,

)

Для того чтобы Flutter знал, где искать изображения, необходимо добавить путь к ним в pubspec.yaml:

flutter:

assets:

- assets/images/logo.png

**4. Работа с ресурсами устройства**

Flutter предоставляет возможность взаимодействовать с устройством, например, открывать веб-страницы, делать телефонные звонки или отправлять сообщения. Для этого используются сторонние библиотеки, такие как url\_launcher.

**Вызов телефонного номера**

Чтобы вызвать телефонный номер из приложения, можно использовать схему tel:. Для этого необходимо установить библиотеку url\_launcher и вызвать функцию, которая передаст номер в стандартное приложение для звонков.

Пример вызова номера:

import 'package:url\_launcher/url\_launcher.dart';

Future<void> \_makePhoneCall(String url) async {

final Uri launchUri = Uri(scheme: 'tel', path: '+1234567890');

if (await canLaunchUrl(launchUri)) {

await launchUrl(launchUri);

} else {

throw 'Не удалось открыть $url';

}

}

1. **Uri(scheme: 'tel', path: 'номер')** — создаёт URI с использованием телефонной схемы.
2. **canLaunchUrl** — проверяет, может ли устройство открыть этот URI (например, поддерживается ли функция звонков).
3. **launchUrl** — открывает номер в приложении для звонков.

**Открытие внешних ссылок**

Для открытия веб-ссылок или другого контента также можно использовать url\_launcher:

await launchUrl(Uri.parse('https://example.com'));

**5. Структура экрана в Flutter**

Для построения интерфейсов в Flutter используются виджеты. Один из ключевых виджетов — это Scaffold, который предоставляет готовую структуру экрана с верхней панелью (AppBar), основным содержимым (body) и другими элементами.

Пример использования Scaffold:

Scaffold(

appBar: AppBar(

title: Text('Заголовок страницы'),

),

body: Center(

child: Text('Основное содержимое страницы'),

),

)

**6. Работа с макетом (layout)**

Flutter предоставляет несколько виджетов для компоновки элементов:

1. **Column** — располагает элементы вертикально.
2. **Row** — располагает элементы горизонтально.
3. **Center** — выравнивает дочерний элемент по центру экрана.
4. **Padding** — задаёт отступы вокруг виджета.

**7. Жесты и взаимодействие с пользователем**

Для добавления интерактивных элементов в приложение, таких как нажатие на текст или изображение, используется виджет GestureDetector. Он позволяет реагировать на различные жесты, например, нажатие, двойное нажатие, долгое нажатие и т.д.

Пример обработки нажатия на текст:

GestureDetector(

onTap: () {

print('Текст был нажат');

},

child: Text('Нажми меня'),

)

**Заключение**

Работа с текстом и ресурсами в Flutter — это важная часть разработки любого приложения. Используя мощные инструменты Flutter, можно легко настраивать внешний вид текста, работать с изображениями и взаимодействовать с функциями устройства, такими как телефонные звонки. В данном практическом занятии мы познакомились с основными возможностями, которые помогут создать удобные и функциональные интерфейсы в приложениях.

**Практическая часть**

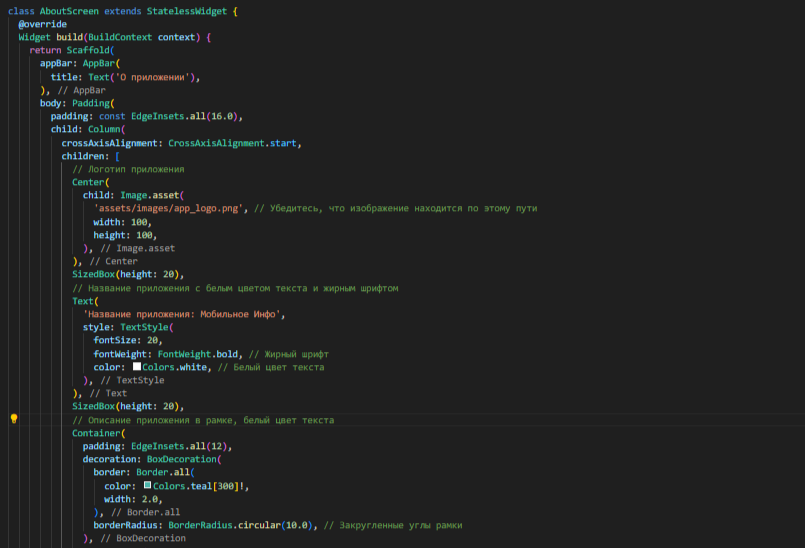
Откройте файл about\_screen.dart и внесем следующие данные.



**1. Импорт библиотек**

1. flutter/material.dart: Основная библиотека Flutter для построения пользовательского интерфейса с использованием виджетов.
2. url\_launcher/url\_launcher.dart: Библиотека для работы с внешними ресурсами, в частности, для открытия ссылок или запуска встроенных функций устройства (например, звонок).

2. **Создание класса AboutScreen**



1. Класс AboutScreen наследуется от StatelessWidget, что означает, что экран не будет изменять своё состояние (данные не будут меняться динамически).

2. Scaffold: Структура страницы с готовыми элементами, такими как AppBar (верхняя панель) и body (основная часть).

3. AppBar: Создаёт верхнюю панель с заголовком "О приложении".

4. Padding: Задаёт отступы вокруг содержимого страницы.

5. Column: Виджет, который располагает дочерние элементы вертикально.

3. **Добавление логотипа приложения**

1. Image.asset: Загружает изображение из локальных ресурсов. Путь к изображению должен быть указан в pubspec.yaml.

2. Center: Центрирует изображение на экране.

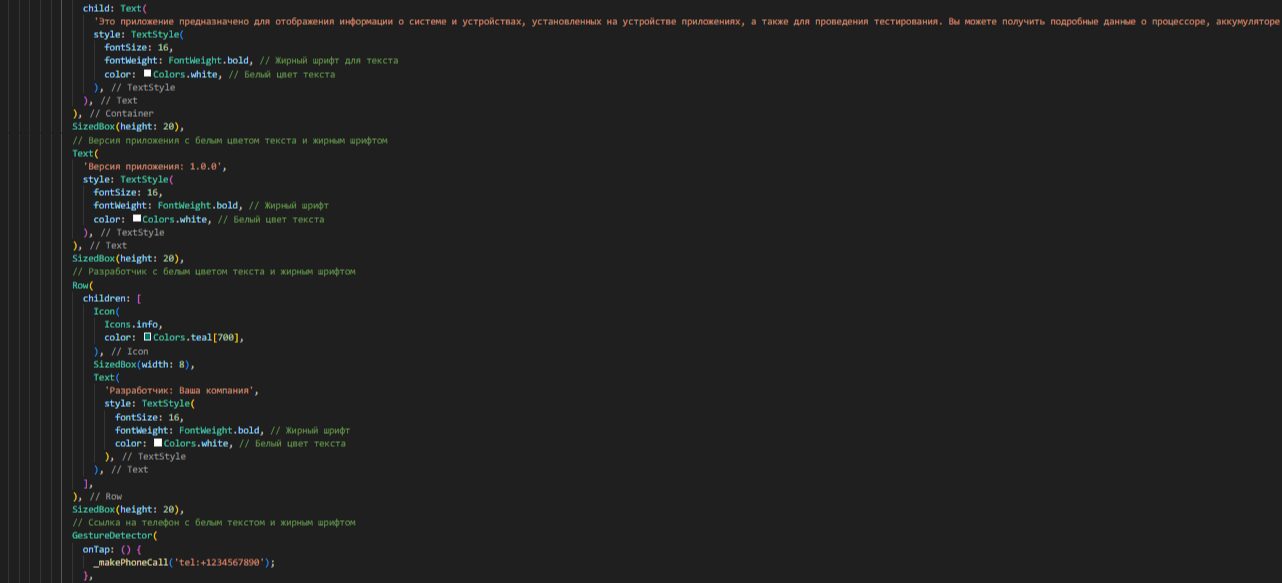
3. Ширина и высота изображения заданы как 100 пикселей.

4. Text: Отображает текст на экране.

5. TextStyle: Оформление текста, включая размер шрифта, цвет и жирность.

6. В коде несколько таких блоков для отображения названия приложения, версии, разработчика и описания.

5. **Рамка вокруг описания приложения**



1. Container: Контейнер для оформления текста.

2. padding: Внутренний отступ текста от границ контейнера.

3. BoxDecoration: Оформление контейнера — добавлена рамка (border) и закругленные углы (borderRadius).

4. Внутри контейнера находится текст с описанием приложения.

6. **Отображение информации о разработчике**

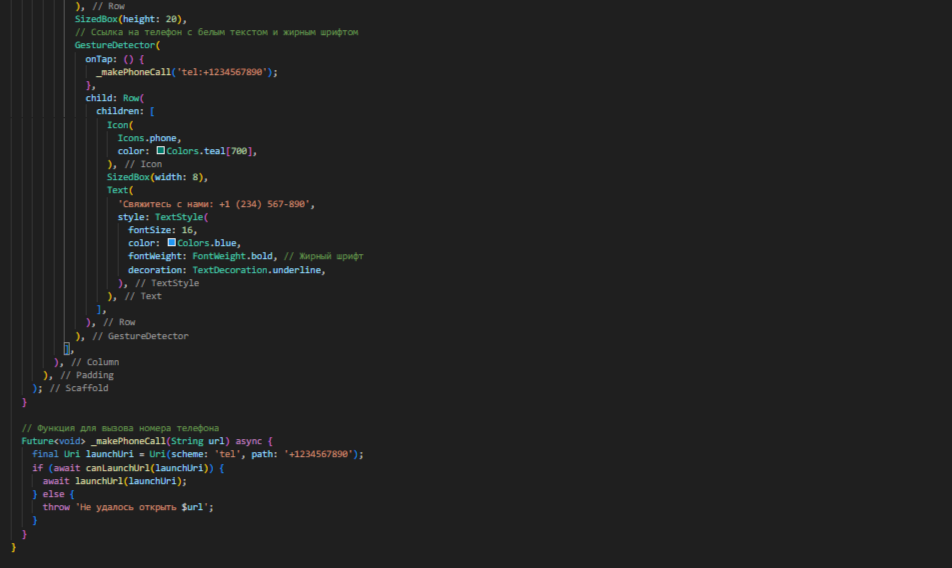
1. Row: Располагает элементы горизонтально.

2. Icon: Добавляет иконку "Информация" (предустановленный значок в Flutter).

3. SizedBox: Добавляет отступ между иконкой и текстом (8 пикселей).

4. Текст отображает информацию о разработчике.

7. **Нажатие на номер телефона для звонка**



1. GestureDetector: Этот виджет позволяет "слушать" события касания (например, нажатие на текст или иконку).

2. Когда пользователь нажимает на текст, вызывается функция \_makePhoneCall с номером телефона.

8. **Функция для вызова телефона**

1. Эта функция использует библиотеку url\_launcher, чтобы вызвать номер телефона.
2. Uri(scheme: 'tel', path: '+1234567890'): Здесь создаётся URI с телефонным номером.
3. canLaunchUrl: Проверяет, может ли устройство открыть этот URI (в данном случае — начать вызов).
4. launchUrl: Открывает номер телефона в стандартном приложении для звонков.

Если устройство не может совершить звонок, выбрасывается ошибка.

Этот код создаёт простую страницу "О приложении", которая показывает название, описание, информацию о разработчике и даёт возможность сделать телефонный звонок. Используются стандартные компоненты Flutter для построения пользовательского интерфейса и библиотека url\_launcher для взаимодействия с функциями устройства, такими как звонки.

**Контрольные вопросы:**

1. Какие основные виджеты используются для работы с текстом в Flutter?
2. Как можно настроить стиль текста с помощью виджета TextStyle? Какие параметры можно изменить?
3. Что такое Container и как его можно использовать для оформления текста или других виджетов?
4. Как задать рамку и закруглённые углы для виджета в Flutter?
5. Как добавить изображение в проект Flutter и отобразить его на экране?
6. Где необходимо указать путь к изображениям в проекте Flutter?
7. Для чего используется виджет Scaffold и какие компоненты он может включать?
8. Как с помощью библиотеки url\_launcher вызвать номер телефона из приложения Flutter?
9. Что делает функция canLaunchUrl и для чего она используется?
10. Как добавить возможность открытия веб-ссылок в приложении Flutter?
11. Что такое виджет GestureDetector и как его можно использовать для обработки событий взаимодействия с пользователем?
12. Какие виджеты в Flutter используются для компоновки элементов вертикально и горизонтально?
13. Как можно задать отступы вокруг виджетов в Flutter?
14. Какие схемы URI можно использовать для взаимодействия с устройством (например, для звонков или открытия веб-ссылок)?
15. Как проверить, поддерживает ли устройство функцию звонка перед попыткой совершить вызов из приложения?

**Дополнительные задания:**

1. **Изменение стиля текста**
   1. Добавьте в экран "О приложении" ещё один текстовый блок с любым текстом. Настройте его стиль так, чтобы текст был красного цвета, размер шрифта был 18, а текст был подчеркивающимся. Используйте виджет Text и настройте стиль через TextStyle.
2. **Добавление новой иконки**
   1. В блоке с информацией о разработчике добавьте новую иконку (например, иконку электронной почты). При нажатии на неё приложение должно открывать почтовый клиент для отправки письма на заранее указанный адрес электронной почты. Используйте библиотеку url\_launcher и GestureDetector для обработки нажатия.
3. **Изменение оформления контейнера**
   1. Измените оформление контейнера, в котором находится описание приложения. Сделайте фон контейнера голубым, а текст внутри контейнера — белым. Также увеличьте ширину рамки до 3 пикселей и измените цвет рамки на красный.
4. **Создание кнопки для открытия веб-страницы**
   1. Добавьте кнопку в нижней части экрана, которая будет открывать любую веб-страницу в браузере (например, сайт вашей компании). Используйте виджет TextButton для создания кнопки и библиотеку url\_launcher для открытия веб-страницы.
5. **Добавление изображения из интернета**
   1. Вместо логотипа из локальных ресурсов добавьте изображение, загружаемое из интернета. Используйте виджет Image.network для загрузки изображения по URL. Убедитесь, что изображение отображается корректно и его размеры ограничены до 150x150 пикселей.