Programmazione Mobile

Nicola Noviello nicola.noviello@unimol.it

Corso di Laurea in Informatica Dipartimento di Bioscienze e Territorio Università degli Studi del Molise Anno 2023/2024

Lezione: Flutter e Dart - Funzionalità avanzate (parte prima)

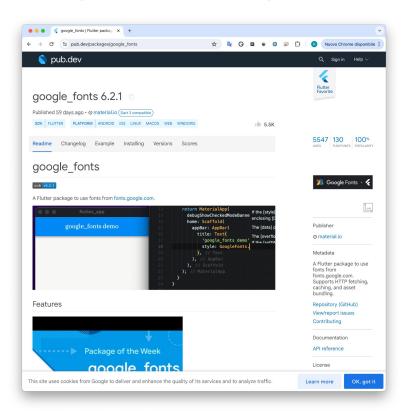
- Rendering dei contenuti in base a criteri condizionali
- Approfondimento del ciclo di vita di un Widget Stateful
- Espressioni ternarie e operatori condizionali
- Data Model
- Stili, Allineamento, Margini e Padding
- Integrazione con Package di terze parti
- Contenuti Scrollabili

Dove eravamo

Integro una funzione in grado di cambiare domanda

```
fontSize: 15.0,
      fontWeight: FontWeight.bold,
      color: ■Colors.white), // TextStyle
 textAlign: TextAlign.center,
), // Text
const SizedBox(height: 30.0),
...currentQuestion
    .getShuffledAnswers()
    .map((answer) => AnswerButton(
          answerText: answer,
          onTap: answerQuestion,
        )), // AnswerButton
```

Integrare package esterni - pub.dev



https://pub.dev/

Memorizzare le risposte

Lifting up state

```
lib > <a> first_screen.dart > <a> FirstScreenState > <a> build</a>
      class _FirstScreenState extends State<FirstScreen> {
        final List<String> selectedAnswers = []; // ok final perché con il .add non muta
        var activeScreen = 'start-screen';
         void switchNewScreen() {
          setState(() {
             activeScreen = 'questions-screen';
        void addAnswer(String answer) {
           selectedAnswers.add(answer):
         @override
         winger builtu(builtucontext context) {
          return MaterialApp(
             home: SafeArea(
               child: Scaffold(
                body: Container(
 28
                   decoration: const BoxDecoration(
                     gradient: LinearGradient(
                       begin: Alignment.bottomRight,
                       end: Alignment.topLeft,
                       colors: [■Colors.white, □Color.fromARGB(255, 22, 60, 118)],
                     ), // LinearGradient
                   ), // BoxDecoration
                   // da usare in caso di initState
                   child: activeScreen == "start-screen"
                       ? StartScreen(switchNewScreen)
                       : QuestionsWidget(onSelectAnswer: addAnswer,),
```

Aggiorno il widget con le domande

```
import 'package:lezione7/answer_button.dart';
import 'package:lezione7/data/questions.dart';

import 'package:lezione7/data/questions.dart';

class QuestionsWidget extends StatefulWidget {
   const QuestionsWidget({
      super.key,
      required this.onSelectAnswer,
   });
   final void Function(String answer) onSelectAnswer;

@override
State<QuestionsWidget> createState() => _QuestionsWidgetState();
}
```

Lifting up state

```
lib > <a> first_screen.dart > <a> FirstScreenState > <a> build</a>
      class _FirstScreenState extends State<FirstScreen> {
        final List<String> selectedAnswers = []; // ok final perché con il .add non muta
        var activeScreen = 'start-screen';
         void switchNewScreen() {
          setState(() {
            activeScreen = 'questions-screen';
        void addAnswer(String answer) {
          selectedAnswers.add(answer):
        @override
        Widget build(BuildContext context) {
          return MaterialApp(
             home: SafeArea(
              child: Scaffold(
                body: Container(
 28
                   decoration: const BoxDecoration(
                     gradient: LinearGradient(
                      begin: Alignment.bottomRight,
                      end: Alignment.topLeft,
                       colors: [■Colors.white, □Color.fromARGB(255, 22, 60, 118)],
                   ), // BoxDecoration
                   // da usare in caso di initState
                   child: activeScreen == "start-screen"
                       ? StartScreen(switchNewScreen)
                       : QuestionsWidget(onSelectAnswer: addAnswer,),
```

Il valore deve essere passato all'interno della State Class

```
class _QuestionsWidgetState extends State __estionsWidget> {
  var currentQuestionIndex = 0;
  void answerQuestion(String selectedAnswer) {
    widget.onSelectAnswer(selectedAnswer);
    setState(() {
      currentQuestionIndex++;
    }):
```

Passa la risposta e aggiorna il contatore

```
questions_screen.dart > 2 QuestionsWidgetState > Duild
class _QuestionsWidgetState extends State<QuestionsWidget> {
 Widget build(BuildContext context) {
              textAlign: TextAlign.center,
              // Text
            const SizedBox(height: 30.0),
            ...currentQuestion.getShuffledAnswers().map(
                  (answer) => AnswerButton(
                    answerText: answer,
                    onTap: () {
                      answerQuestion(answer);
                  ). // AnswerButton
           // Column
      ). // Container
    ); // SizedBox
```

Evitare di uscire fuori dall'index delle domande

Redirect in home quando raggiungiamo la "fine"

```
void addAnswer(String answer) {
    selectedAnswers.add(answer);
    if (selectedAnswers.length == questions.length){
        setState(() {
            activeScreen = 'start-screen';
        });
    }
}
```

Perfetto, quindi funziona?

```
@ass _FirstScreenState extends State<FirstScreen> {
 List<String> selectedAnswers = []; /
 var activeScreen = 'start-screen';
 void switchNewScreen() {
   setState(() {
     activeScreen = 'questions-screen';
 void addAnswer(String answer) {
   selectedAnswers.add(answer);
   if (selectedAnswers.length == questions.length) {
     setState(() {
       selectedAnswers = [];
       activeScreen = 'start-screen';
     });
```

Mostriamo i risultati

Schermata desiderata



La schermata con il risultato sarà un Widget Stateless o Stateful?

ResultWidget

```
result_screen.dart > ...
  import 'package:flutter/material.dart';
  class ResultWidget extends StatelessWidget {
   const ResultWidget({super.key});
   @override
   Widget build(BuildContext context) {
     return SizedBox(
       width: double.infinity, // allargamento massimo
       child: Container(
         margin: const EdgeInsets.all(40.0),
         child: const Column(
           mainAxisAlignment: MainAxisAlignment.center,
           children: [
             Text('Hai risposto correttamente a X domande su Y'),
             SizedBox(
               height: 30,
             ), // SizedBox
             Text('Lista di domande e risposte'),
             SizedBox(
               height: 30,
             TextButton(
               onPressed: null,
               child: Text('Ricomincia'),
     ); // SizedBox
```

Integriamo nel flusso delle schermate il nuovo Widget

```
first_screen.dart > ...
 class _FirstScreenState extends State<FirstScreen> {
   void addAnswer(String answer) {--
   @override
   Widget build(BuildContext context) {
     Widget screenWidget = StartScreen(switchNewScreen);
     if (activeScreen == 'questions-screen') {
       screenWidget = QuestionsWidget(
         onSelectAnswer: addAnswer,
     if (activeScreen == 'results-screen') {
       screenWidget = const ResultWidget();
     return MaterialApp(
       home: SafeArea(
         child: Scaffold(
           body: Container(
             decoration: const BoxDecoration(
               gradient: LinearGradient(
                 begin: Alignment.bottomRight,
                 end: Alignment.topLeft,
                 colors: [■Colors.white, □Color.fromARGB(255, 22, 60, 118)],
             ), // BoxDecoration
             // da usare in caso di initState
             child: screenWidget,
         ), // Scaffold
       ). // SafeArea
     ); // MaterialApp
```

Passare le informazioni a ResultWidget

```
lib > ( result_screen.dart > ( ResultWidget > ( build
       import 'package:flutter/material.dart';
       class ResultWidget extends StatelessWidget {
         const ResultWidget({
           super key.
           required this chosenAnswers,
         });
         final List<String> chosenAnswers;
         @override
 11
         Widget build(BuildContext context) {
           return SizedBox(
 13
             width: double.infinity, // allargamento massimo
             child: Container(
               margin: const EdgeInsets.all(40.0),
               child: const Column(
                 mainAxisAlignment: MainAxisAlignment.center,
 17
                 children: [
```

Passare le informazioni a ResultWidget

```
class _FirstScreenState extends State<FirstScreen> {
 List<String> selectedAnswers = [];
 var activeScreen = 'start-screen';
 void switchNewScreen() {
   setState(() {
     activeScreen = 'questions-screen';
 void addAnswer(String answer) {
   selectedAnswers.add(answer);
   if (selectedAnswers.length == questions.length) {
     setState(() {
       selectedAnswers = [];
       activeScreen = 'results-screen';
 @override
 Widget build(BuildContext context) {
   Widget screenWidget = StartScreen(switchNewScreen);
   if (activeScreen == 'questions-screen') ₹
     screenWidget = QuestionsWidget(
       onSelectAnswer: addAnswer,
   if (activeScreen == 'results-screen') {
     screenWidget = ResultWidget(chosenAnswers: selectedAnswers,);
   return MaterialApp(
```

Controlliamo le risposte

```
result screen.dart
lib > <a> result_screen.dart > <a> ResultWidget > <a> build</a>
       import 'package:flutter/material.dart';
       class ResultWidget extends StatelessWidget {
         const ResultWidget({
           super key,
           required this chosenAnswers,
         final List<String> chosenAnswers;
         List<Map<String, Object>> getSummaryData() {
           final List<Map<String, Object>> summary = [];
           // qui il codice per controllare i risultati
           return summary;
```

Una Map è una raccolta di **coppie chiave-valore**, dove ogni chiave è associata ad un valore.

Le **chiavi devono essere uniche** all'interno della mappa, ma i **valori possono essere duplicati**.

Controlliamo le risposte

```
List<Map<String, Object>> getSummaryData() {
  final List<Map<String, Object>> summary = [];
  for (var i=0; i< chosenAnswers.length; i++)</pre>
  summary.add(
      'indice_domanda': i,
      'domanda': questions[i].question,
      'risposta corretta': questions[i].answers[0],
      'risposta dell\'utente': chosenAnswers[i];
  return summary;
```

Widget specifico per le risposte

```
lib > ( question_summary.dart > ...
       import 'package:flutter/material.dart';
       class QuestionSummary extends StatelessWidget {
        const QuestionSummary(this.summaryData, {super.key});
        final List<Map<String, Object>> summaryData;
        @override
        Widget build(BuildContext context) {
           return Column(
            children:
               summaryData.map( // crea dei widget popolandoli con i dostri dati
                 (data) {
                   return const Row(
                     children: [
                       Text(data['domanda']),
                     1,
               ).toList(), // è un iterabile, ci serve una lista di Widget
           ); // Column
 22
```

Cast di una variabile

```
result_screen.dart 1
                         question_summary.dart 2
question.dart
lib > ( question_summary.dart > ...
       import 'package:flutter/material.dart';
       class QuestionSummary extends StatelessWidget {
        const QuestionSummary(this.summaryData, {super.key});
        final List<Map<Stri
                             The argument type 'Object?' can't be assigned to the parameter type 'String'.
        @override
                              dart(argument_type_not_assignable)
         Widget build(BuildC
           return Column(
                             Arguments of a constant creation must be constant expressions.
                             Try making the argument a valid constant, or use 'new' to call the
             child ren:
              sum_naryData.m constructor. dart(const_with_non_constant_argument)
                 (lata) {
                             Type: String
                   return co
                     childre View Problem (飞F8) Quick Fix... (黑.)
                       Text(data['domanda']),
                   ); // Row
               ).toList(), // è un iterabile, ci serve una lista di Widget
           ); // Column
 22
```

Integriamo le risposte

```
lib > 🐧 question_summary.dart > ...
      import 'package:flutter/material.dart';
      class QuestionSummary extends StatelessWidget {
        const QuestionSummary(this.summaryData, {super.key});
        final List<Map<String, Object>> summaryData;
        @override
        Widget build(BuildContext context) {
          return Column(
            children: summaryData.map(
              // crea dei widget popolandoli con i dostri dati
              (data) {
                return Row(
                  children:
                    Text(
                      ((data['indice_domanda'] as int) + 1).toString(),
                    Column(
                      children: [
                        Text(data['domanda'] as String),
                        const SizedBox(height: 10),
                        Text(data['risposta corretta'] as String),
                        Text(data['risposta dell\'utente'] as String),
                    ), // Column
                ); // Row
            ).toList(), // è un iterabile, ci serve una lista di Widget
          ): // Column
      3+.
 32
```

Aggiungiamo il Widget dinamico alla schermata di

riepilogo

Funziona?

```
result screen.dart •  question summary.dart
                                                    questions screen.dart
lib > ( result_screen.dart > ...
       class ResultWidget extends StatelessWidget {
         List<Map<String, Object>> getSummaryData() {
         @override
         Widget build(BuildContext context) {
           return SizedBox(
             width: double.infinity, // allargamento massimo
             child: Container(
               margin: const EdgeInsets.all(40.0),
               child: Column(
                 mainAxisAlignment: MainAxisAlignment.center,
                 children: [
                   const Text('Hai risposto correttamente a X domande su Y'),
                   const SizedBox(
                     height: 30,
                   ), // SizedBox
                   QuestionSummary(getSummaryData()),
                   const SizedBox(
                     height: 30,
                   ), // SizedBox
                   const TextButton(
                     onPressed: null,
                     child: Text('Ricomincia'),
                   ), // TextButton
               ), // Column
             ). // Container
           ); // SizedBox
```

Aggiungiamo il Widget dinamico alla schermata di

riepilogo

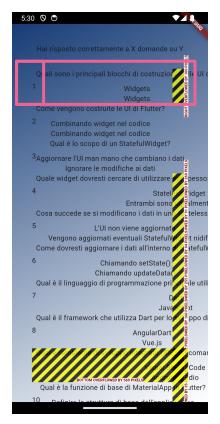
Hai risposto correttamente a X domande su Y Ricomincia

Funziona? No

```
void addAnswer(String answer) {
    selectedAnswers.add(answer);
    if (selectedAnswers.length == questions.length) {
        setState(() {
            selectedAnswers = [];
            activeScreen = 'results-screen';
        });
    }
}
```

Aggiungiamo il Widget dinamico alla schermata di riepilogo

Funziona? Alla grande



Out of Boundary: ci aiuta Expanded

```
result screen.dart
                       question summary.dart ×
lib > ● question_summary.dart > 	 QuestionSummary > 	 build
       import 'package:flutter/material.dart';
       import 'package:flutter/widgets.dart';
       class QuestionSummary extends StatelessWidget {
        const QuestionSummary(this.summaryData, {super.key});
        final List<Map<String, Object>> summaryData;
        Widget build(BuildContext context) {
           return Column(
             children: summaryData.map(
              // crea dei widget popolandoli con i dostri dati
               (data) {
                 return Row(
                   children: [
                     Text(
                       ((data['indice_domanda'] as int) + 1).toString(),
                     ), // Text
                     Expanded(
                       child: Column(
                         children:
                           Text(data['domanda'] as String),
                          const SizedBox(height: 10),
 23
                           Text(data['risposta corretta'] as String),
                           Text(data['risposta dell\'utente'] as String),
                     ), // Expanded
                 ); // Row
             ).toList(), // è un iterabile, ci serve una lista di Widget
           ): // Column
```

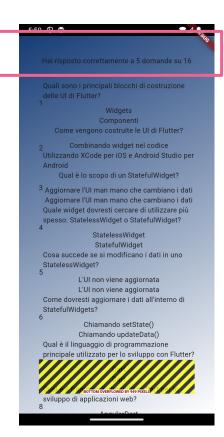


Adatta il contenuto allo spazio disponibile!

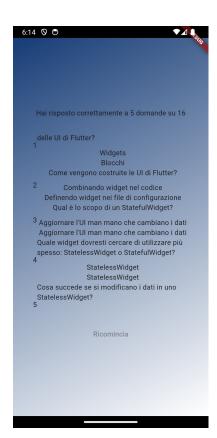
Aggiungiamo il numero di risposte corrette

```
result screen.dart X
question summary.dart
                                                start screen.dart
                                                                                □ C * ↑ * □ Q
lib > ( result_screen.dart > ( ResultWidget > ( build
      List<Map<String, Object>> getSummaryData() {
         final List<Map<String, Object>> summary = [];
          for (var i = 0; i < chosenAnswers.length; i++) {</pre>
            summary.add(
                'indice domanda': i.
                'domanda': questions[i].question,
                'risposta corretta': questions[i].answers[0],
                'risposta dell\'utente': chosenAnswers[i],
          return summary;
        @override
        Widget build(BuildContext context) {
          final summaryData = getSummaryData();
          final nunTotalDomande = questions.length;
          final numRisposteCorrette = 0;
          return SizedBox(
            width: double.infinity, // allargamento massimo
            child: Container(
              margin: const EdgeInsets.all(40.0),
              child: Column(
               mainAxisAlignment: MainAxisAlignment.center,
                children:
                  Text(
                      'Hai risposto correttamente a $numRisposteCorrette domande su $nunTotalDomande'), //
                  const SizedBox(
                   height: 30,
                  ), // SizedBox
                  QuestionSummary(summaryData),
                  const SizedBox
                   height: 30,
```

Aggiungiamo il numero di risposte corrette



Contenuti scrollabili: SingleChildScrollView



```
import 'package:flutter/material.dart';
import 'package:flutter/widgets.dart';
class QuestionSummary extends StatelessWidget {
 const QuestionSummary(this.summaryData, {super.key});
 final List<Map<String, Object>> summaryData;
 @override
 Widget build(BuildContext context) {
   return SizedBox(
     height: 350.
     child: SingleChildScrollView(
       child: Column(
         children: summaryData.map(
           // crea dei widget popolandoli con i dostri dati
           (data) {
             return Row(
               children: [
                   ((data['indice_domanda'] as int) + 1).toString(),
                 Expanded(
                   child: Column(
                     children: [
                       Text(data['domanda'] as String),
                       const SizedBox(height: 10),
                       Text(data['risposta corretta'] as String),
                       Text(data['risposta dell\'utente'] as String),
                    // Column
                 , // Expanded
         ).toList(), // è un iterabile, ci serve una lista di Widget
   ): // SizedBox
```

Esercizio

Esercizio: Finalizzazione del progetto svolto nelle ultime due lezioni

L'obiettivo di questo esercizio è quello di verificare le conoscenze acquisite fino a questo punto del corso.

Scaricate lo zip con il nome esercizio ed integrate quello che manca per completare il progetto: pulsante per ritorno alla home ed uno o più Widget che permettano di rappresentare graficamente le risposte come da screen mostrato nella slide successiva.

Esercizio: Finalizzazione del progetto svolto nelle

ultime due lezioni



Esercizio: Finalizzazione del progetto svolto nelle ultime due lezioni

Consegna:

- Inviare il progetto in uno zip a <u>nicola.noviello@unimol.it</u> utilizzando come oggetto della e-mail "Lezione8: Nome Cognome Matricola"; In alternativa è possibile usare GitHub, garantendomi l'accesso al progetto, ma è sempre necessario mandarmi una mail, nel caso, con il link al progetto;
- Da completare tassativamente nelle ore di lezione residue della giornata odierna, dopo aver completato la lezione teorica;
- Non obbligatorio, ma valutato positivamente ai fini dell'esame.