BCSCHOOL

Empowering Digital Skills For The Jobs Of The Future

BC Soft
consulenza sistemi informativi

by

AiRED
Artificial Intelligence Ricerca e Sviluppo

Navigazione

Docente

Claudia Infante

claudia.infante@bcsoft.net

Sommario

- Navigator
- Named Routes

Navigator

In Flutter la navigazione si riferisce al processo di spostamento da una schermata o pagina a un'altra schermata o pagina in risposta agli input dell'utente o agli eventi del sistema.

Navigator

Inizialmente, la navigazione tra le pagine di Flutter era possibile solo con Navigator 1.0, con le azioni di push e pop delle pagine.

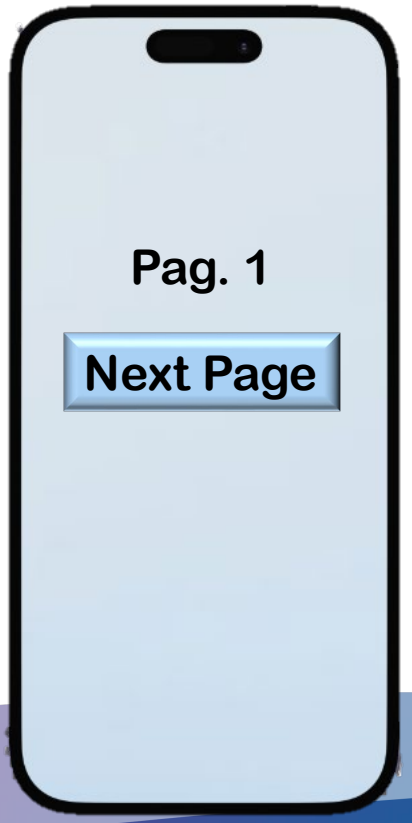
Navigator

Flutter implementa il concetto di datatype Stack per rendere possibile la navigazione.

Lo Stack è una pila di oggetti che si riferisce al concetto di LIFO (Last In First Out).
Nel caso di Flutter si fa riferimento ad una pila di schermate.

Navigator

La prima volta che si lancia un'applicazione è presente una sola schermata.

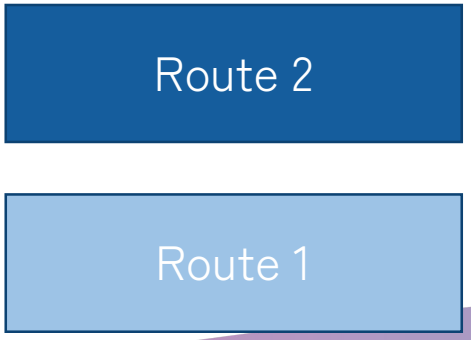


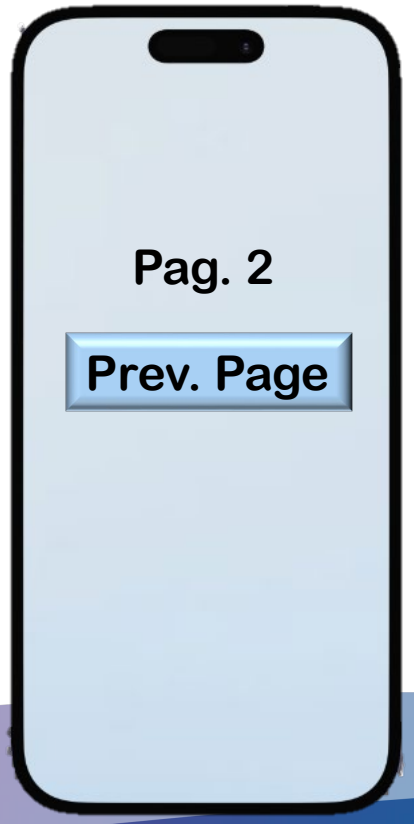
Route 1

Navigator

Nel momento in cui si passa alla visualizzazione di una seconda schermata, la precedente viene nascosta per mostrare la nuova.

L'operazione utilizza il metodo push.





Navigator

Il metodo push viene utilizzato per navigare verso una nuova schermata o pagina dell'applicazione. Prende come argomento un oggetto Route, che rappresenta la schermata o la pagina da visualizzare.

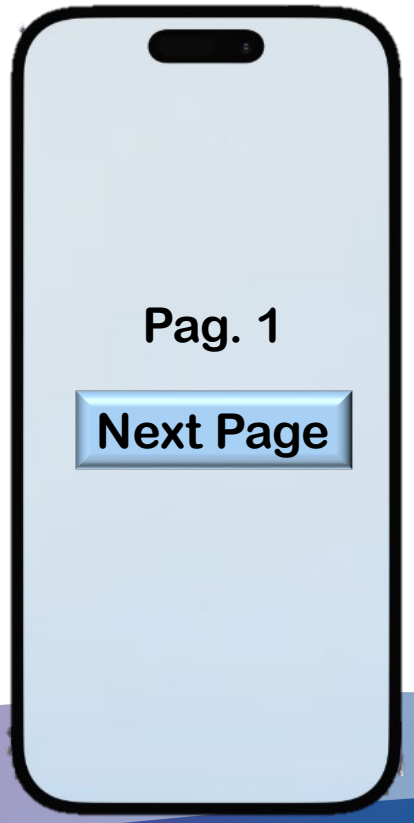
Navigator

Quando si chiama push, il nuovo percorso viene aggiunto in cima allo stack di navigazione, mentre il percorso precedente rimane nascosto.

```
Navigator.push(
  context,
  MaterialPageRoute(builder: (context) => SecondPage()),
);
```

Navigator

Il metodo pop viene utilizzato per tornare alla schermata o alla pagina precedente dell'applicazione.



Route 1

Navigator

Pila di pagine da navigare

navigator.push

navigator.pop

Navigator

Quando si invoca pop, il percorso corrente viene rimosso dalla cima dello stack di navigazione e il percorso precedente diventa visibile.

Navigator

La classe Navigator ha a disposizione una serie di metodi tra cui :

- pop
- popUntil
- canPop
- push
- pushNamed
- popAndPushNamed
- Replace
- pushAndRemoveUntil

Navigator

Navigator 1.0 utilizza le classi Navigator e Route per navigare tra le pagine.

Navigator

Flutter offre diversi modi per implementare la navigazione, tra cui:

Navigator: Il widget Navigator è un widget integrato in Flutter che fornisce un sistema di navigazione basato su stack. È possibile utilizzare il widget Navigator per navigare verso diverse schermate o pagine dell'applicazione.

Navigator

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

Run | Debug | Profile
void main() => runApp(const MyApp());

class MyApp extends StatelessWidget {
  const MyApp({super.key});

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Flutter Navigation Demo',
      theme: ThemeData(
        colorScheme: ColorScheme.fromSeed(seedColor: Colors.green),
        useMaterial3: true,
      ), // ThemeData
      home: const MyHomePage(),
    ); // MaterialApp
  }
}
```

Navigator

```dart
class MyHomePage extends StatelessWidget {
  const MyHomePage({super.key});

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: const Text('Flutter Navigation Demo'),
        backgroundColor: Theme.of(context).colorScheme.inversePrimary,
      ), // AppBar
      body: Center(
        child: TextButton(
          child: const Text('Go to Second Page'),
          onPressed: () {
            Navigator.push(
              context,
              MaterialPageRoute(builder: (context) => const SecondPage()),
            );
          },
        ), // TextButton
      ), // Center
    ); // Scaffold
  }
}

```

Navigator

```dart
class SecondPage extends StatelessWidget {
  const SecondPage({super.key});

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: const Text('Second Page'),
        backgroundColor: Theme.of(context).colorScheme.inversePrimary,
      ), // AppBar
      body: Center(
        child: TextButton(
          child: const Text('Go back to First Page'),
          onPressed: () {
            Navigator.pop(context);
          },
        ), // TextButton
      ), // Center
    ); // Scaffold
  }
}
```

NamedRoutes

Percorsi denominati: È possibile definire percorsi denominati nella propria applicazione utilizzando i widget MaterialApp o CupertinoApp.

NamedRoutes

Le NamedRoutes consentono di fare riferimento a una schermata o a una pagina specifica dell'applicazione utilizzando un nome univoco.

NamedRoutes

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

Run | Debug | Profile
void main() => runApp(const MyApp());

class MyApp extends StatelessWidget {
  const MyApp({super.key});

  @override
  Widget build(BuildContext context) {
  return MaterialApp(
      title: 'Flutter Named Routes Demo',
      theme: ThemeData(
        colorScheme: ColorScheme.fromSeed(seedColor: Colors.green),
        useMaterial3: true,
      ), // ThemeData
      initialRoute: '/',
      routes: {
        '/' : (context) => const MyHomePage(),
        '/second': (context) => const SecondPage(),
      },
    ); // MaterialApp
  }
}

```

NamedRoutes

```dart
25
26  class MyHomePage extends StatelessWidget {
27    const MyHomePage({super.key});
28
29    @override
30    Widget build(BuildContext context) {
31      return Scaffold(
32        appBar: AppBar(
33          title: const Text('Home Page'),
34          backgroundColor: Theme.of(context).colorScheme.inversePrimary,
35        ), // AppBar
36        body: Center(
37          child: TextButton(
38            child: const Text('Go to Second Page'),
39            onPressed: () {
40                Navigator.pushNamed(context, '/second');
41            },
42          ), // TextButton
43        ), // Center
44      ); // Scaffold
45    }
46  }
47
```

NamedRoutes

```dart

class SecondPage extends StatelessWidget {
  const SecondPage({super.key});

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: const Text('Second Page'),
        backgroundColor: Theme.of(context).colorScheme.inversePrimary,
      ), // AppBar
      body: Center(
        child: TextButton(
          child: const Text('Go back to Home Page'),
          onPressed: () {
            Navigator.pop(context);
          },
        ), // TextButton
      ), // Center
    ); // Scaffold
  }
}

```

NamedRoutes

La navigazione in flutter consente anche il passaggio di informazioni da una pagina all'altra.

NamedRoutes

Tramite il NamedRouting queste proprietà vengono passate attraverso il routing come argomenti.

NamedRoutes

Gli argomenti forniti vengono passati alla rotta ‘spinta’ tramite [RouteSettings.arguments].

```
26  class MyHomePage extends StatelessWidget {
27    const MyHomePage({super.key});
28
29    @override
30    Widget build(BuildContext context) {
31      return Scaffold(
32        appBar: AppBar(
33          title: const Text('Home Page'),
34          backgroundColor: Theme.of(context).colorScheme.inversePrimary,
35        ), // AppBar
36        body: Center(
37          child: TextButton(
38            child: const Text('Go to Second Page'),
39            onPressed: () {
40                Navigator.pushNamed(context, '/second', arguments: "Username");
41            },
42          ), // TextButton
43        ), // Center
44      ); // Scaffold
45    }
46  }
47
```

NamedRoutes

Si indica il parametro da ricevere sfruttando la classe ModalRoute .

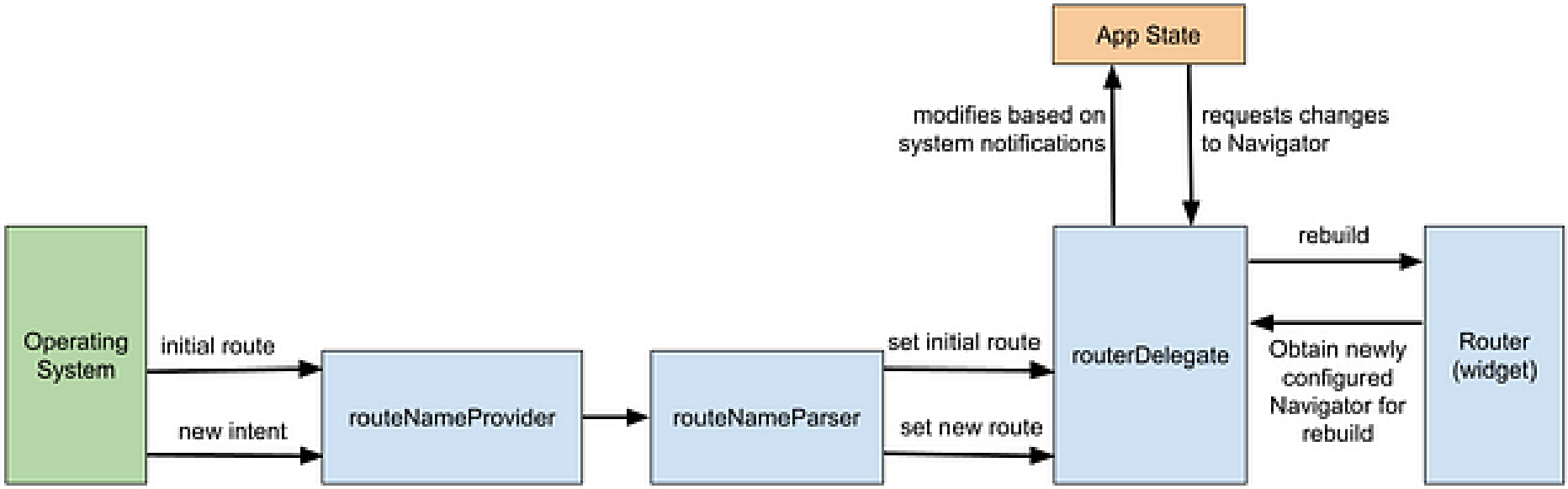
```dart
class SecondPage extends StatelessWidget {
  const SecondPage({super.key});

  @override
  Widget build(BuildContext context) {
    final arg = ModalRoute.of(context)!.settings.arguments;
    return Scaffold(
      appBar: AppBar(
        title: Text('Second Page, hello $arg'),
        backgroundColor: Theme.of(context).colorScheme.inversePrimary,
      ), // AppBar
      body: Center(
        child: TextButton(
          child: const Text('Go back to Home Page'),
          onPressed: () {
            Navigator.pop(context);
          },
        ), // TextButton
      ), // Center
    ); // Scaffold
  }
}
```

Navigator 2.0

I problemi legati all'uso di Navigator 1.0 sono la mancanza di un pieno controllo sullo stack, le navigazioni innestate e il supporto web.

A differenza di Navigator 1.0, che utilizza uno stile di programmazione imperativo, Navigator 2.0 utilizza uno stile dichiarativo.

Navigator 2.0



Navigator 2.0

Una comprensione di Navigator 2.0 implica la comprensione di alcuni concetti quali:

Page: Una classe astratta che descrive la configurazione di una rotta.

Router: Una classe che gestisce l'apertura e la chiusura delle pagine di un'applicazione.

Navigator 2.0

RouteInformationParser: Una classe astratta usata dal widget del router per analizzare le informazioni sulle rotte in una configurazione.

RouteInformationProvider: Una classe astratta che fornisce informazioni sulle rotte per il widget del Router.

Navigator 2.0

RouterDelegate: Una classe astratta usata dal widget del Router per costruire e configurare un widget di navigazione.

BackButtonDispatcher: Segnala a un router quando l'utente tocca il pulsante indietro su piattaforme che supportano i pulsanti indietro (come Android).

TransitionDelegate: Il delegato che decide come le pagine passano all'interno o all'esterno dello schermo quando vengono aggiunte o rimosse.

Navigator 2.0

Con Navigator 2.0 è possibile definire percorsi di navigazione nella propria applicazione utilizzando i widget MaterialApp o CupertinoApp.

Navigator 2.0

Navigator 2.0 mette a disposizione una navigazione dichiarativa che supporta i deep linking e le data-driven routes.

Navigator 2.0

Questi percorsi consentono di definire una schermata o una pagina specifica dell'applicazione e di navigare verso di essa utilizzando un URL o un link.

Navigator 2.0

```dart
import 'package:flutter/material.dart';
import 'package:go_router/go_router.dart';

Run | Debug | Profile
void main() => runApp(const MyApp());
final _router = GoRouter(
  initialLocation: '/',
  routes: [
    GoRoute(
      name: 'home',
      path: '/',
      builder: (context, state) => const MyHomePage()
      ), // GoRoute
  GoRoute(
      name: 'second',
      path: '/second',
      builder: (context, state) => const SecondPage()), // GoRoute
  ]
  ); // GoRouter

```

Navigator 2.0

```dart
19
20  class MyApp extends StatelessWidget {
21    const MyApp({super.key});
22
23    @override
24    Widget build(BuildContext context) {
25   return MaterialApp.router(
26        title: 'My App',
27          theme: ThemeData(
28          colorScheme: ColorScheme.fromSeed(seedColor: Colors.green),
29          useMaterial3: true,
30        ), // ThemeData
31         routerConfig: _router,
32   ); // MaterialApp.router
33    }
34  }
35
```

Navigator 2.0

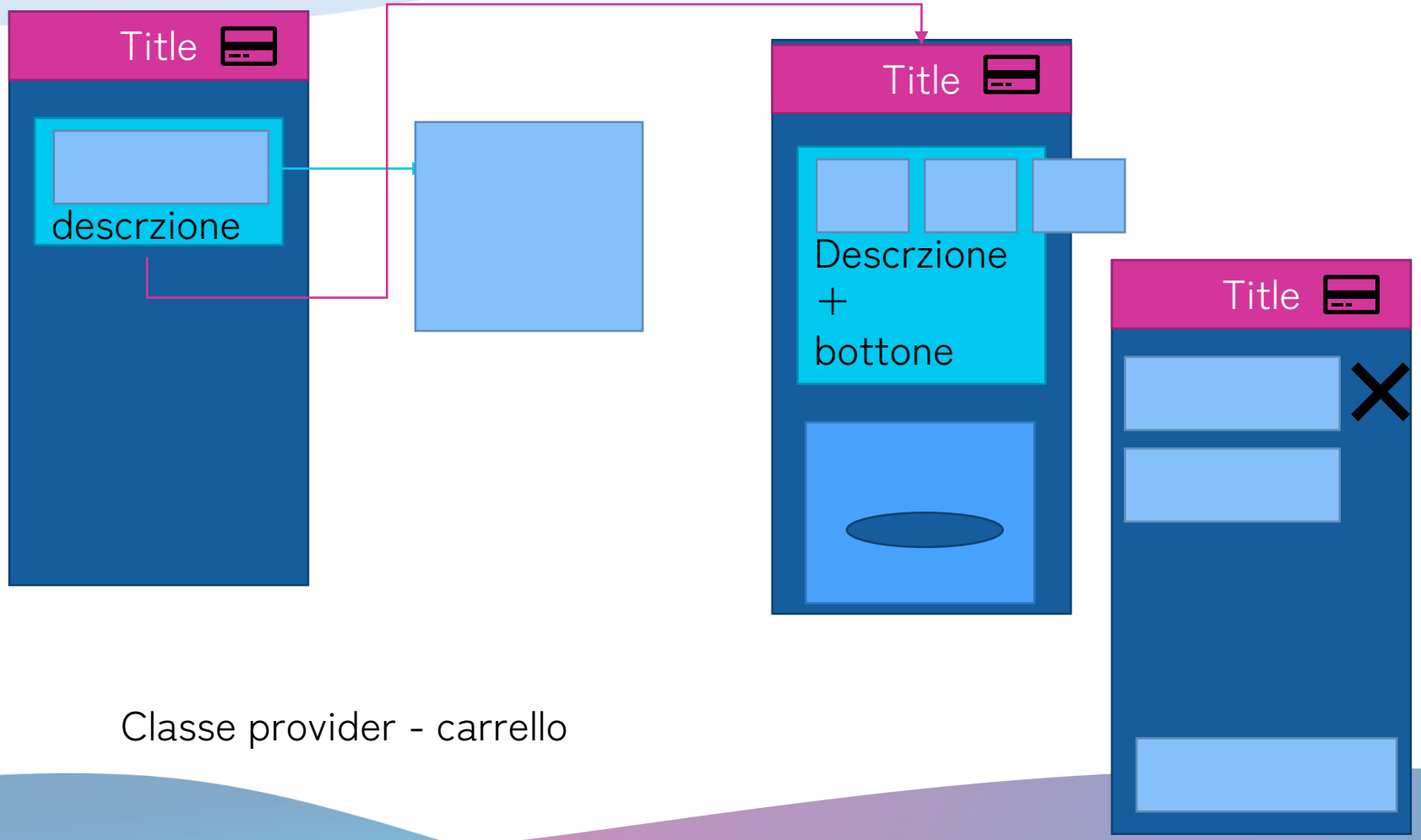
```dart
class MyHomePage extends StatelessWidget {
  const MyHomePage({super.key});

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: const Text('Home Page'),
        backgroundColor: Theme.of(context).colorScheme.inversePrimary,
      ), // AppBar
      body: Center(
        child: TextButton(
          child: const Text('Go to Second Page'),
          onPressed: () {
            context.go('/second');
          },
        ), // TextButton
      ), // Center
    ); // Scaffold
  }
}
```

Navigator 2.0

```dart
57
58  class SecondPage extends StatelessWidget {
59    const SecondPage({super.key});
60
61    @override
62    Widget build(BuildContext context) {
63      return Scaffold(
64        appBar: AppBar(
65          title: const Text('Second Page'),
66          backgroundColor: Theme.of(context).colorScheme.inversePrimary,
67        ), // AppBar
68        body: Center(
69          child: TextButton(
70            child: const Text('Go to Home Page'),
71            onPressed: () {
72            context.go('/');
73            },
74          ), // TextButton
75        ), // Center
76      ); // Scaffold
77    }
78  }
79
```

Domande e approfondimenti

Esempio struttura



Classe provider - carrello