INSTINCT INSTITUTE

ICT Department

លំអិតលើ ListView, Detail Page, GridView និង PageView

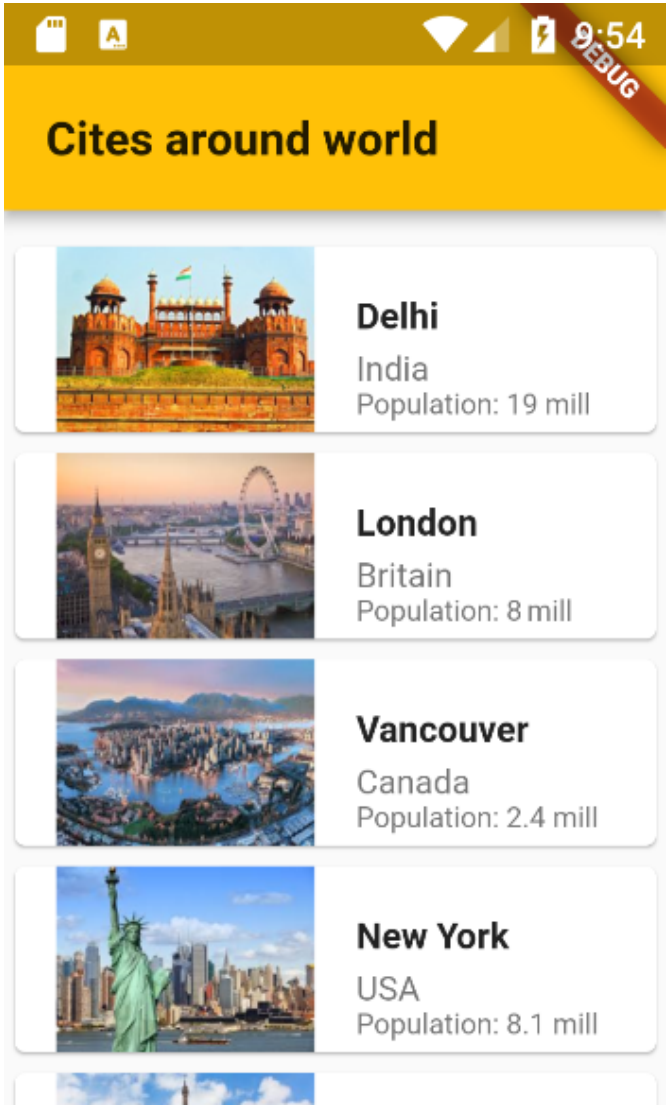
Instructor: **Oum Saokosal**, *Master of Engineering in Information Systems, South Korea '2010*

Email: oumsaokosal@gmail.com

Phone: 012 252 752 (Telegram)

ListView

- ListView គឺជា widget ដែលអាច scroll បាន ហើយគេប្រើវាស្ទើរតែគ្រប់ពេលសំរាប់ទិន្នន័យដែលច្រើនៗ។ យើងក៏អាចប្តូរទិសដៅក្នុងការ scroll ពីក្រោមឡើងលើ ពីស្តាំមកឆ្វេងក៏បាន។
- ListView នៅក្នុង Flutter គឺល្អប្រើខ្លាំងណាស់ ដែលមិនដូចជា ListView នៅក្នុង native Android ទេ ដែលវាទាមទារអោយយើងសរសេរកូដច្រើនដើម្បីទប់បញ្ហាដូចជា ViewHolder, Lazy Load, និង cache image ជាដើម។ល។ ListView នៅក្នុង Flutter អនុញ្ញាតិអោយយើងផ្ទុកទិន្នន័យប្រភេទអ្វីក៏បាន អក្សរ រូបភាព ឬទោះបីជាវីដេអូក៏ដោយ។



ListView Options

ListView មាន Option សំខាន់ៗដូចជា៖

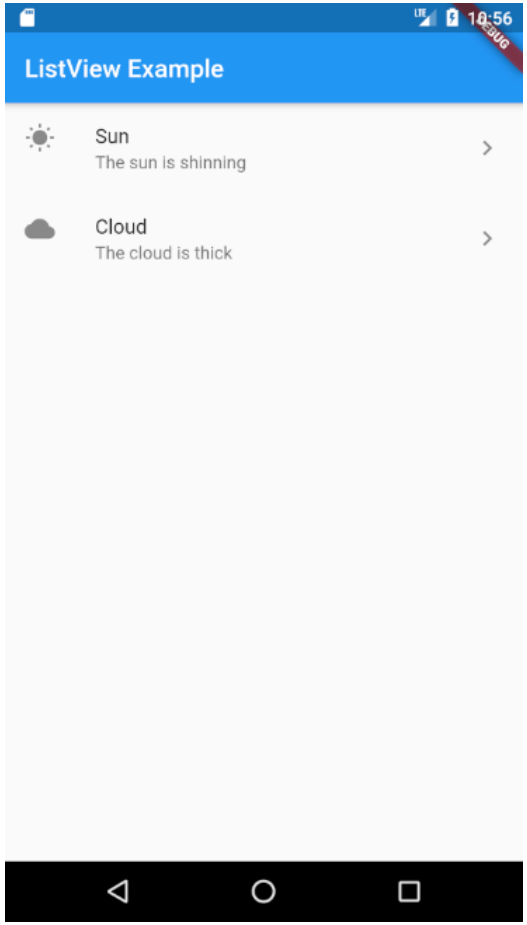
- `children:` សំរាប់ដាក់ទិន្នន័យដែលមានប្រភេទណាមួយក៏បាន លាយគ្នាក៏បាន អោយតែសណ្តានរបស់ widget។
- `scrollDirection:` ប្រើសំរាប់ប្តូរទិសដៅ scroll
  - `Axis.horizontal`៖ scroll ផ្តេកពីឆ្វេងទៅស្តាំ
  - `Axis.vertical`៖ scroll បញ្ឈរពីលើចុះក្រោម
- `reverse: true` ៖ គឺអោយវា scroll បញ្ច្រាស់
- `controller` ៖ គឺសំរាប់យើងអាចសរសេរកូដអោយវា scroll ដោយស្វ័យប្រវត្តិ
- `ListView.builder()` ៖ ប្រើសំរាប់ build ធាតុខាងក្នុងនៃ List

```dart
ListView(
  scrollDirection: Axis.vertical,
  reverse: false,
  children: <Widget>[
    Text("apple"),
    Icon(Icons.home),
    RaisedButton(
      child: Text("Click Me"),
      onPressed: () {},
    ),
    Container(
      width: 300.0,
      height: 200.0,
      child: Image.network(
        "http://bit.ly/2IGluzb",
      ),
    ),
  ],
)
```

ListView និង ListTile

- ListTile គឺជា widget មួយដែលគេនិយមប្រើជាមួយ ListView ព្រោះ ListTile មាន option ល្អៗ មួយចំនួនរួចជាស្រេចដូចជា leading, title, subtitle, trailing, និង event click ជាដើម។

```
ListView(
  children: <Widget>[
    ListTile(
      leading: Icon(Icons.wb_sunny),
      title: Text('Sun'),
      subtitle: Text("The sun is shinning"),
      trailing: Icon(Icons.keyboard_arrow_right),
    ),
    ListTile(
      leading: Icon(Icons.wb_cloudy),
      title: Text('Cloud'),
      subtitle: Text("The cloud is thick"),
      trailing: Icon(Icons.keyboard_arrow_right),
    ),
  ],
)
```

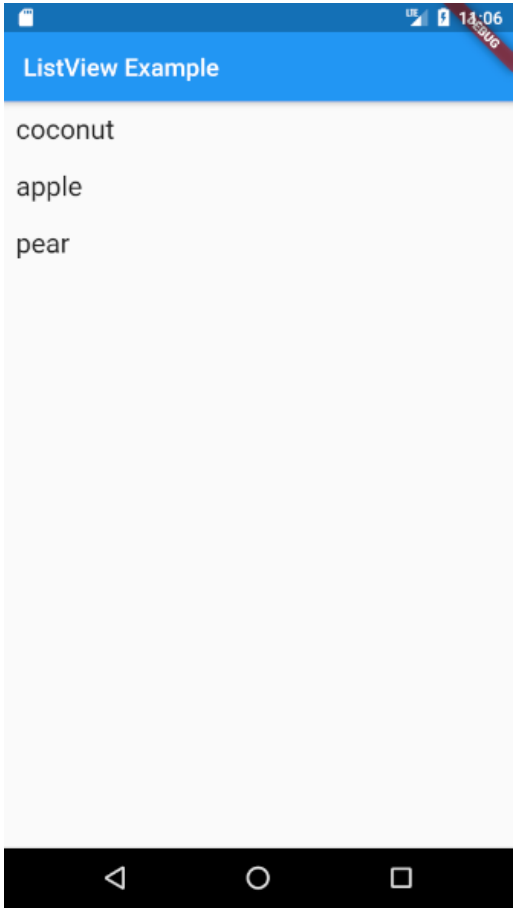


ListView.builder

ListView.builder គឺប្រើសំរាប់ដាក់ធាតុរបស់វាជាទំរង់ dynamic។ ListView.builder មាន parameter សំខាន់២ គឺ itemCount សំរាប់កំណត់ចំនួនធាតុ និង itemBuilder សំរាប់អោយយើងបង្កើតធាតុជាលក្ខណៈ dynamic។ វាធ្វើការល្អជាមួយនឹងទិន្នន័យប្រភេទជា List។

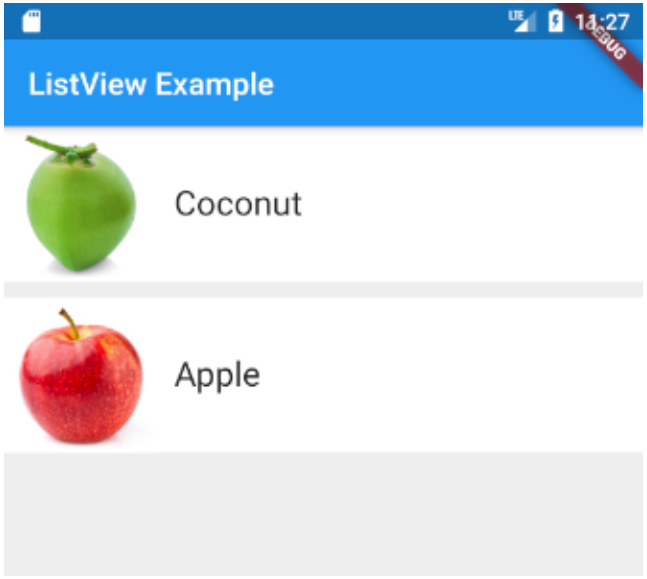
```
List<String> fruits = ["coconut", "apple", "pear"];
```

```
ListView.builder(
    itemCount: fruits.length,
    itemBuilder: (context, index) {
      return Container(
        padding: EdgeInsets.all(10.0),
        child: Text(
          fruits[index],
          style: TextStyle(fontSize: 22.0),
        ),
      );
    }),
```



ListView.builder

```dart
class Fruit {
  String title;
  String imageUrl;

  Fruit({this.title, this.imageUrl});
}

List<Fruit> fruits = [
  Fruit(title: "Coconut",
      imageUrl: "assets/coconut.jpg"),
  Fruit(title: "Apple",
      imageUrl: "assets/apple.jpg")
];
```

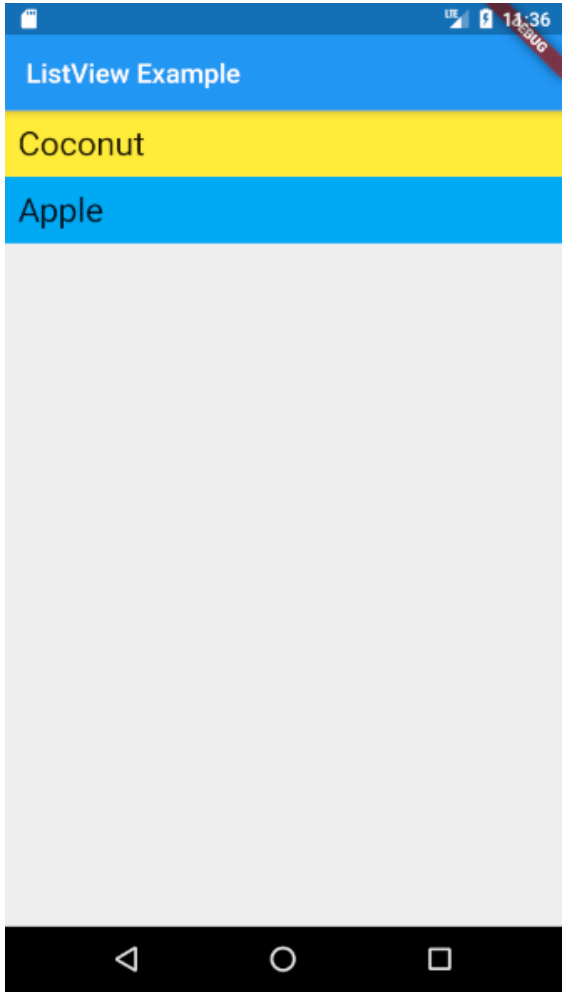



```dart
return Container(
  color: Colors.grey[200],
  child: ListView.builder(
      itemCount: fruits.length,
      itemBuilder: (context, index) {
        return Container(
          margin: EdgeInsets.only(bottom: 10.0),
          color: Colors.white,
          child: Row(
            children: <Widget>[
              Container(
                width: 100.0,
                height: 100.0,
                child: Image.asset(
                  fruits[index].imageUrl,
                  fit: BoxFit.cover,
                ),
              ),
              Container(
                padding: EdgeInsets.all(10.0),
                child: Text(
                  fruits[index].title,
                  style: TextStyle(fontSize: 22.0),
                ),
              ),
            ],
          ),
        );
      }),
);
```

ListView.builder

យើងអាចសង្កេតឃើញក្នុងកូដមុននេះថា នៅក្នុង itemBuilder គឺមាន return widge។ លក្ខណៈនេះគឺអាចអោយយើងអាចផ្លាស់ប្តូរឬកំណត់រូបរាងរបស់ធាតុបានយ៉ាងងាយស្រួល។

```
ListView.builder(
    itemCount: fruits.length,
    itemBuilder: (context, index) {
      if(index % 2 == 0){
        return Container(
          color: Colors.yellow,
          padding: EdgeInsets.all(10.0),
          child: Text(fruits[index].title,
            style: TextStyle(fontSize: 25.0),),
        );
      }
      else{
        return Container(
          color: Colors.lightBlue,
          padding: EdgeInsets.all(10.0),
          child: Text(fruits[index].title,
             style: TextStyle(fontSize: 25.0),),
        );
      }
    })
```



Detail Page

ជាធម្មតា ListView គឺអាចបង្ហាញទិន្នន័យបានច្រើនមែន តែដោយសារទំហំអេក្រង់តូច យើងមិនអាចបង្ហាញគ្រប់ទិន្នន័យទាំងអស់ទេ។ ដូច្នេះយើងត្រូវ Page មួយទៀត ដើម្បីជំនួយក្នុងបង្ហាញទិន្នន័យអោយលំអិតជាមុន។ យើងអាចដាក់ event tab លើ item និមួយៗបានតាមរយៈ widget មួយចំនួនដូចជា ListTile, InkWell, RaisedButton, FlatButton, IconButton ជាដើម។ ហើយពេលយើងចុចលើ item និមួយៗនោះហើយ យើងនឹងបើក Page មួយទៀតប្រើ Navigator។

```
ListView.builder(
    itemCount: fruits.length,
    itemBuilder: (context, index) {
      return ListTile(
        title: Text(fruits[index].title),
        trailing: Icon(Icons.keyboard_arrow_right),
        onTap: (){
          Navigator.push(context,
               MaterialPageRoute(builder: (context)=> DetailPage()));
        },
      );
    })
```

Navigator សំរាប់បើក Page មួយទៀត

Navigator មានតួរនាទីសំរាប់បើកផ្ទាំង Page មួយផ្សេងទៀត។ យើងអាចប្រើ push សំរាប់បើក Page ថ្មីតាម constructor ឬប្រើ pushNamed សំរាប់បើក Page ដែលបានកំណត់ហើយក្នុង routes។

- Push:

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

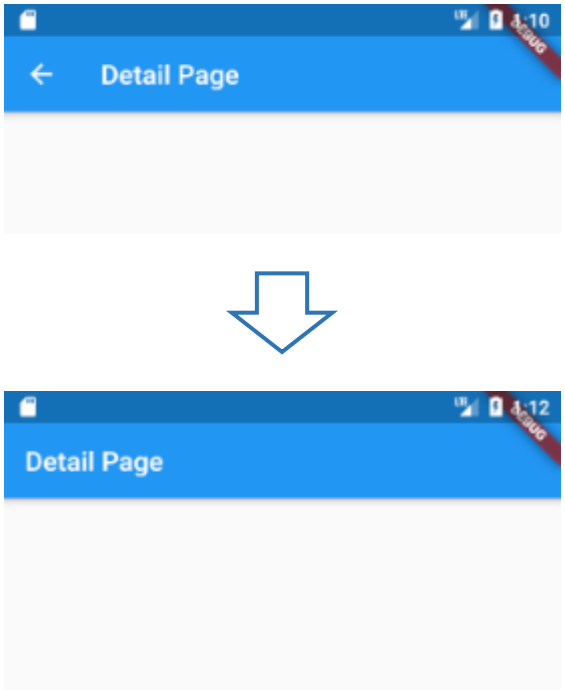
- PushNamed ជាមួយ Route:

```
Navigator.of(context).pushNamed("/nextpage");
```

```
class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
//      home: MyHome(), //home cannot be with routes
      routes: {
        "/" : (context) => MyHome(),
        "/nextpage" : (context) => NextPage(),
      },
    );
  }
```

Navigating ដោយមិនត្រឡប់ក្រោយ

ពេលខ្លះយើងបើក Page ថ្មីហើយគឺយើងមិនចង់អោយគេបកក្រោយបានទេ។ ដើម្បីធ្វើចឹងបានគឺយើងអាចប្រើ pushReplacement ឬ pushReplacementNamed។



- PushReplacement:

```
Navigator.pushReplacement(context,
          MaterialPageRoute(builder: (context) =>
                    NextPage()));
```

- PushReplaceNamed:

```
Navigator.of(context).pushReplacementNamed("/nextpage");
```

បញ្ជូនទិន្នន័យជាមួយ Navigator

- យើងអាចប្រើ constructor ដើម្បី បញ្ជូនទិន្នន័យជាមួយនឹង Navigator។

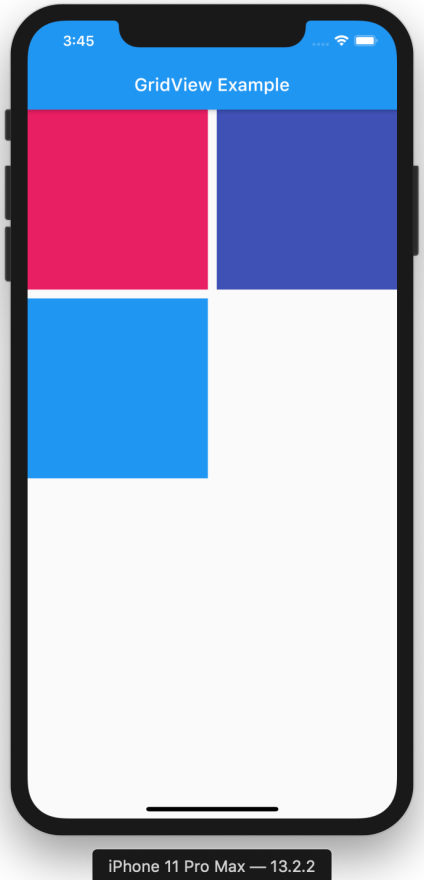
```
Navigator.push(context, MaterialPageRoute(builder: (context) =>
        NextPage(text: "some text")
));
```

```
class NextPage extends StatefulWidget {

  final String text;
   NextPage({this.text});

  @override
  _NextPageState createState() => _NextPageState();
}

class _NextPageState extends State<NextPage> {
  @override
  Widget build(BuildContext context) {
    return Container(child: Text(this.widget.text));
  }
}
```

GridView: GridView.count

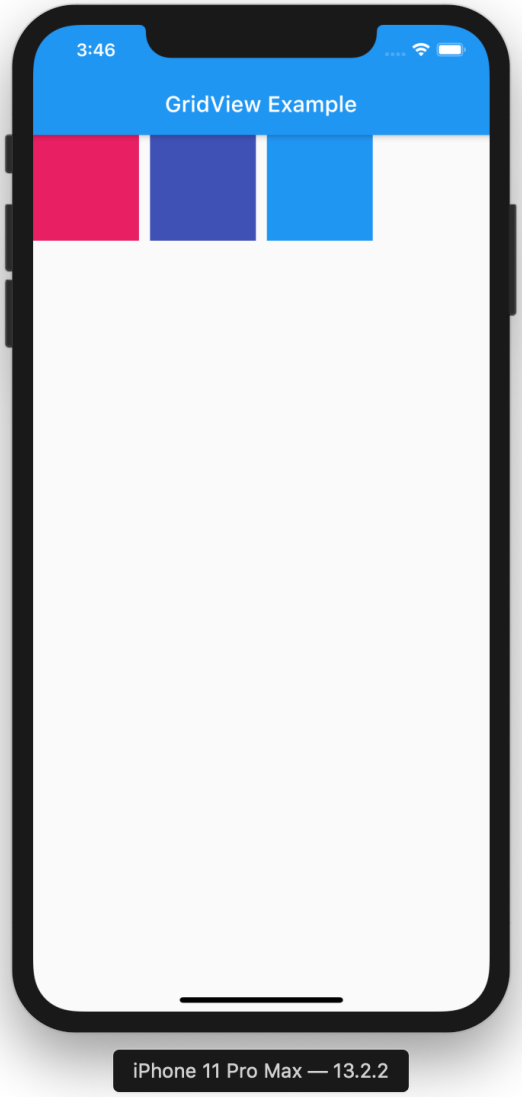
- GridView គឺសំរាប់បង្ហាញ Item ជាលក្ខណៈ Grid ដែលំទំរង់ដូចជា Matrix។
- GridView.count()៖ យើងអាចប្រើវា ប្រសិនបើចង់បានចំនួន Column ជាក់លាក់
- GridView.extent()៖ យើងអាចប្រើវា ប្រសិនបើចង់កំណត់ប្រវែងផ្តេករបស់ Item

```
GridView.count(
  crossAxisCount: 2,
  mainAxisSpacing: 10,
  crossAxisSpacing: 10,
  children: <Widget>[
    Container(color: Colors.pink),
    Container(color: Colors.indigo),
    Container(color: Colors.blue),
  ],
),
```



GridView.extent

```
GridView.extent(
  maxCrossAxisExtent: 100,
  mainAxisSpacing: 10,
  crossAxisSpacing: 10,
  children: <Widget>[
    Container(color: Colors.pink),
    Container(color: Colors.indigo),
    Container(color: Colors.blue),
  ],
)
```



GridView ជាមួយ gridDelegate

```
GridView(
  gridDelegate: SliverGridDelegateWithFixedCrossAxisCount(
    crossAxisCount: 3,
    mainAxisSpacing: 10,
    crossAxisSpacing: 10,
  ),
  children: <Widget>[
    Container(color: Colors.pink),
    Container(color: Colors.indigo),
    Container(color: Colors.blue),
  ],
),
```

gridDelegate អាចជា៖

- SliverGridDelegateWithFixedCrossAxisCount
- SliverGridDelegateWithMaxCrossAxisExtent

GridView.builder

GridView.builder គឺប្រើសំរាប់ build item ដែលមានទំរង់ស្រដៀងនឹង ListView.builder ដែរៗ វាមាន Option សំខាន់៣៖

1. itemCount: ចំនួនរបស់ item
2. gridDelegate ដែលអាចជា៖
    1. SliverGridDelegateWithFixedCrossAxisCount
    2. SliverGridDelegateWithMaxCrossAxisExtent
3. itemBuilder ប្រើសំរាប់ build item

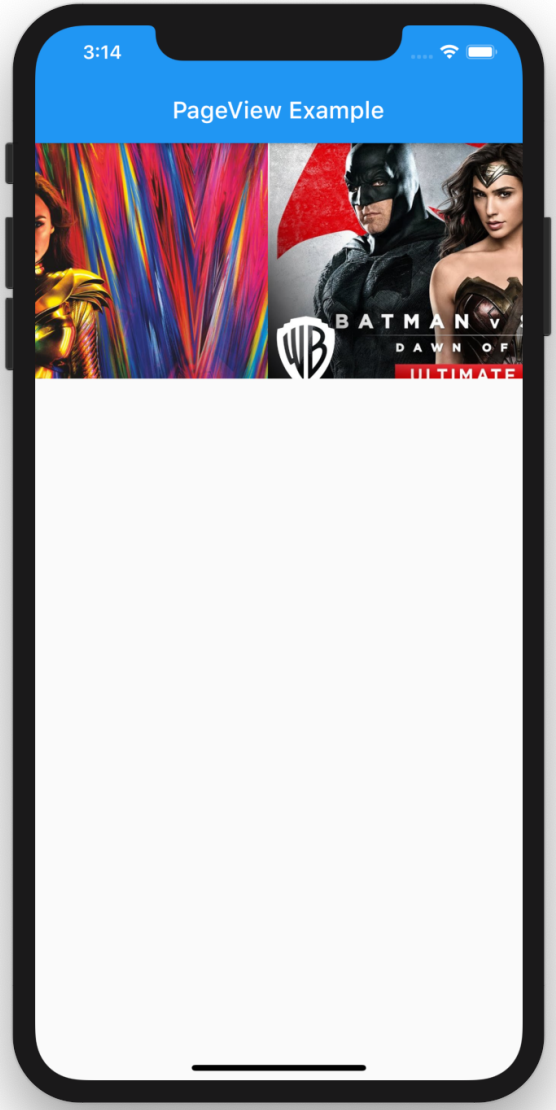
```
GridView.builder(
  itemCount: 10,
  gridDelegate:
  SliverGridDelegateWithFixedCrossAxisCount(
    crossAxisCount: 2,
    mainAxisSpacing: 10,
    crossAxisSpacing: 20,
  ),
  itemBuilder: (context, index) {
    return Container(
      alignment: Alignment.center,
      color: Colors.pink[100 * (index % 9)],
      child: Text('grid item $index'),
    );
  },
```

iPhone 11 Pro Max — 13.2.2

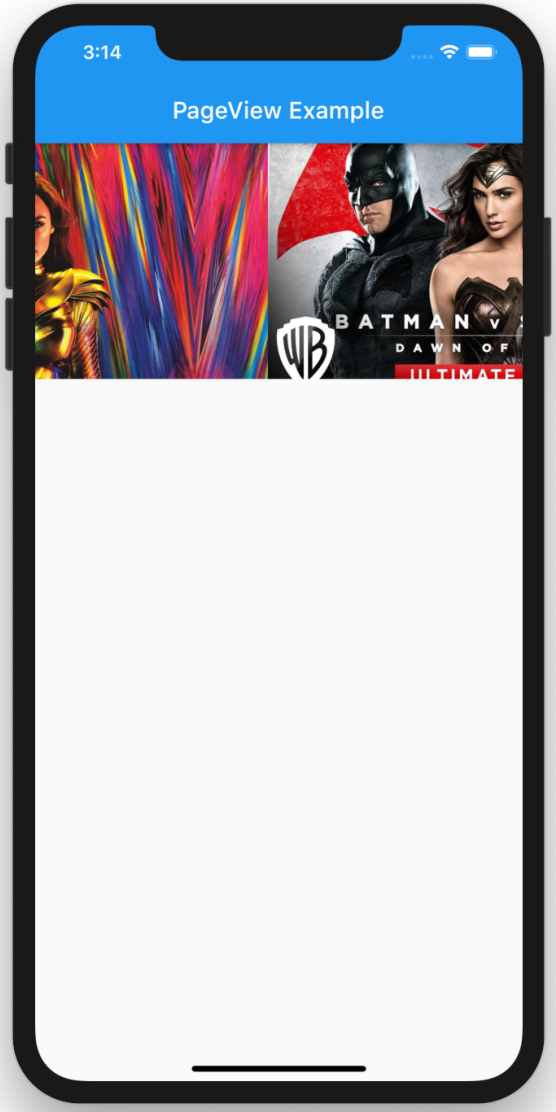
PageView

PageView មានទំរងដូចជា ListView ដែរ តែវា scroll ម្តងមួយ Page ៗ។ Option មាន៖

- scrollDirection
- physics
- controller



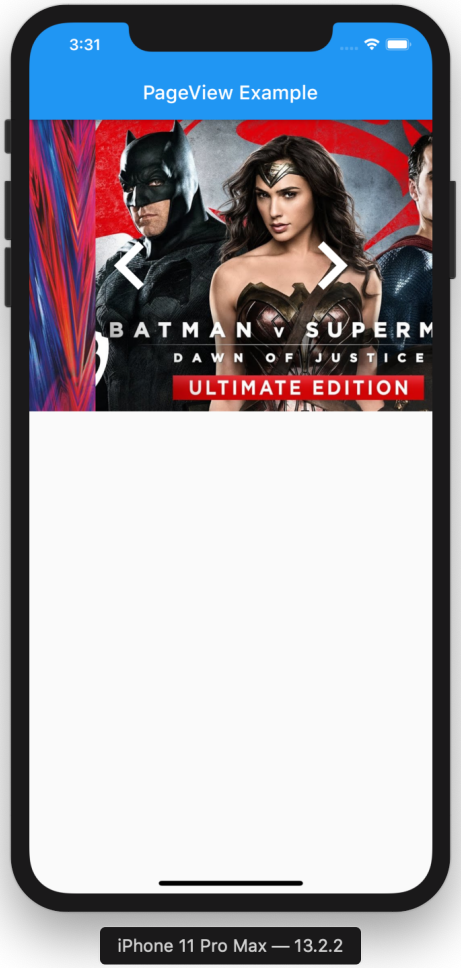
```
Container(
     alignment: Alignment.topCenter,
     height: 200,
     physics: BouncingScrollPhysics(),
     scrollDirection: Axis.horizontal,
     reverse: false,
     pageSnapping: true,
     child: PageView(
       children: <Widget>[
         Image.network(_img1, fit: BoxFit.cover,),
         Image.network(_img2, fit: BoxFit.cover,),
         Image.network(_img3, fit: BoxFit.cover,),
       ],
     ),
);
```



PageView ជាមួយ PageController

```
PageController _scroller = PageController();
int _currentIndex = 0;

get _buildBody {
  return Stack(
    alignment: Alignment.center,
    children: <Widget>[
      _buildPageView,
      _buildNavigations,
    ],
  );
}
```

```
get _buildPageView {
  return Container(
    alignment: Alignment.topCenter,
    height: 300,
    child: PageView(
      controller: _scroller,
      physics: BouncingScrollPhysics(),
      scrollDirection: Axis.horizontal,
      reverse: false,
      pageSnapping: true,
      onPageChanged: (index) {
        setState(() {
          _currentIndex = index;
        });
      },
      children: <Widget>[
        Image.network(_img1, fit: BoxFit.cover),
        Image.network(_img2, fit: BoxFit.cover),
        Image.network(_img3, fit: BoxFit.cover),
      ],
    ),
  );
}
```



```
get _buildNavigations {
  return Row(
    mainAxisAlignment: MainAxisAlignment.spaceAround,
    children: <Widget>[
      Container(
        width: 120, height: 120,
        child: IconButton(
          icon: Icon(Icons.navigate_before, color: Colors.white, size: 100),
          onPressed: () {
            _scroller.animateToPage(_currentIndex - 1,
              duration: Duration(milliseconds: 300),
              curve: Curves.easeInOut,
            );
          },
        ),
      ),
      Container(
        width: 120, height: 120,
        child: IconButton(icon: Icon(Icons.navigate_next, color: Colors.white, size: 100),
          onPressed: () {
            _scroller.animateToPage(_currentIndex + 1,
              duration: Duration(milliseconds: 300),
              curve: Curves.easeInOut,
            );
          },
        ),
      ),
    ],
  );
}
```