**GESTURE TRONG FLUTTER**

*Gesture*(cử chỉ) là cách mà người dùng tương tác với các thiết bị di động. Có rất nhiều cử chỉ khác nhau như vuốt, chạm, lắc... *Gesture* trong Flutter giúp ta xử lý các cử chỉ của người dùng, tương tự như việc bắt sự kiện trong Android.

Một số cử chỉ được sử dụng rộng rãi:

* **Tap** − Chạm vào bề mặt thiết bị bằng đầu ngón tay trong thời gian ngắn sau đỏ thả ngón tay ra ngay
* **Double Tap** − Tap 2 lần trong thời gian ngắn
* **Drag** − Chạm vào bề mặt của thiết bị bằng đầu ngón tay và sau đó di chuyển đầu ngón tay một cách ổn định và cuối cùng thả ngón tay ra.
* **Flick** − Tương tự như drag nhưng thực hiện nhanh hơn.
* **Pinch** − Chụm bề mặt của thiết bị bằng hai ngón tay
* **Spread/Zoom** − Ngược lại với **Pinch**.
* **Panning** − Chạm vào bề mặt của thiết bị bằng đầu ngón tay và di chuyển nó theo bất kỳ hướng nào mà không nhả đầu ngón tay.

Flutter cung cấp một sự hỗ trợ tuyết vời để xử lý tất cả các loại cử chỉ thông qua một tiện ích duy nhất **GestureDetector**.  Để xác định các cử chỉ tác động lên một widget, ta chỉ cần đặt widget đó bên trong GestureDetector widget. GestureDetector sẽ bắt các cử chỉ và gửi nhiều sự kiện dựa trên cử chỉ đó.

Một số cử chỉ và các sự kiện tương ứng được đưa ra dưới đây

Tap

* onTapDown
* onTapUp
* onTap
* onTapCancel

Double tap

* onDoubleTap

Long press

* onLongPress

Vertical drag

* onVerticalDragStart
* onVerticalDragUpdate
* onVerticalDragEnd

Horizontal drag

* onHorizontalDragStart
* onHorizontalDragUpdate
* onHorizontalDragEnd

Pan

* onPanStart
* onPanUpdate
* onPanEnd

Bây giờ chúng ta sẽ mở lại ứng dụng Hello world và thêm vào việc xử lý cử chỉ bằng việc sửa lại *MyHomePage widget*như sau:

body: Center(

child: GestureDetector(

onTap: () {

\_showDialog(context);

},

child: Text('Hello World', )

)

),

Nhìn vào đoạn code trên các bạn có thể thấy *widget Text* được đặt trong *GestureDetector widget*, để bắt sự kiện *onTap* và hiển thị một *Dialog* khi ta chạm vào *Text:*

Ở hàm *\_showDialog* ta sẽ gọi một widget là *AlertDialog*để hiển thị một thông báo, sử dụng đoạn code như sau:

// user defined function void \_showDialog(BuildContext context) {

// flutter defined function

showDialog(

context: context, builder: (BuildContext context) {

// return object of type Dialog

return AlertDialog(

title: new Text("Message"),

content: new Text("Hello World"),

actions: <Widget>[

new FlatButton(

child: new Text("Close"),

onPressed: () {

Navigator.of(context).pop();

},

),

],

);

},

);

}

Toàn bộ code của (main.dart) sau khi chỉnh sửa

import 'package:flutter/material.dart';

void main() => runApp(MyApp());

class MyApp extends StatelessWidget {

// This widget is the root of your application.

@override

Widget build(BuildContext context) {

return MaterialApp(

title: 'Hello World Demo Application',

theme: ThemeData( primarySwatch: Colors.blue,),

home: MyHomePage(title: 'Home page'),

);

}

}

class MyHomePage extends StatelessWidget {

MyHomePage({Key key, this.title}) : super(key: key);

final String title;

// user defined function

void \_showDialog(BuildContext context) {

// user defined function void \_showDialog(BuildContext context) {

// flutter defined function

showDialog(

context: context, builder: (BuildContext context) {

// return object of type Dialog

return AlertDialog(

title: new Text("Message"),

content: new Text("Hello World"),

actions: <Widget>[

new TextButton(

child: new Text("Close"),

onPressed: () {

Navigator.of(context).pop();

},

),

],

);

},

);

}

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(title: Text(this.title),),

body: Center(

child: GestureDetector(

onTap: () {

\_showDialog(context);

},

child: Text( 'Hello World', )

)

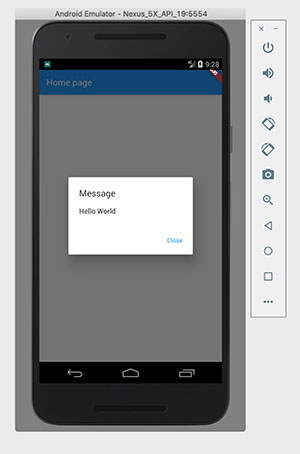
),

);

}

}

Chạy thử trên máy ảo, chạm vào chữ Hello Word sẽ có một cửa sổ dialog hiện ra như sau:



Cuối cùng, **Flutter** cũng cung cấp một cơ chế phát hiện cử chỉ cấp thấp thông qua *Listener widget*. Nó sẽ phát hiện tất cả các tương tác của người dùng và sau đó gửi các sự kiện sau:

* PointerDownEvent
* PointerMoveEvent
* PointerUpEvent
* PointerCancelEvent

Flutter còn cung cấp một số nhỏ các widget để thực hiện các cử chỉ cụ thể đơn giản cũng như phức tạp:

* **Dismissible** − Hỗ trợ flick gesture để đóng widget.
* **Draggable** − Hỗ trợ drag gesture để di chuyển widget.
* **LongPressDraggable** − Hỗ trợ drag gesture để duy chuyển widget, khi widget cha có thể kéo thả
* **DragTarget** − Chấp nhận *Draggable* widget
* **IgnorePointer** − Ẩn widget
* **AbsorbPointer** − Dừng việc xử lý cử chỉ trên widget
* **Scrollable** − Hỗ trợ cuộn nội dung trong widget