





Unidad 6

Paquetes útiles







Guardando datos en el dispositivo





Package of the Week **shared_preferences**

shared_preferences





SharedPreferences

Si tienes una colección relativamente pequeña de pares clave-valor que te gustaría guardar, podemos usar el plugin **shared_preferences**.

Normalmente tendrías que escribir integraciones nativas de plataforma para almacenar datos en ambas plataformas.

Afortunadamente, el plugin **shared_preferences** se puede usar para conservar los datos de clave-valor en el disco.

El plugin de preferencias compartidas ajusta **NSUserDefaults** en iOS y **SharedPreferences** en Android, proporcionando un almacenamiento persistente para datos simples.





SharedPreferences

1. Añadir la dependencia

flutter pub add shared_preferences

2. Guardar Datos

final prefs = await SharedPreferences.getInstance();
prefs.setInt('counter', counter);

3. Leer datos

final counter = prefs.getInt('counter') ?? 0;

4. Eliminar datos

```
prefs.remove('counter');
```







Obtener datos desde internet





http

Obtener datos desde internet es necesario para la mayoría de las apps. Afortunadamente, Dart y Flutter proveen de herramientas para este tipo de trabajo.

- Añadir la dependencia flutter pub add http
- 2. Realiza una petición de red
 Future<http.Response> fetchPost() {
 return http.get('https://jsonplaceholder.typicode.com/posts/1');
 }





http



class Post {

final int userId;

final int id;

final String title;

final String body;

```
Post({this.userId, this.id, this.title,
this.body});
```

factory Post.fromJson(Map<String, dynamic> json)

```
return Post(
   userId: json['userId'],
   id: json['id'],
   title: json['title'],
   body: json['body'],
);
```

```
Future<Post> fetchPost() async {
  final response = await
    http.get('https://jsonplaceholder.typicode.com/posts/1');
```

```
if (response.statusCode == 200) {
  return Post.fromJson(json.decode(response.body));
} else {
  // Si esta respuesta no fue OK, lanza un error.
```

```
throw Exception('Failed to load post');
```

https://esflutter.dev/docs/cookbook/networking/fetch-data





http

4. Obtén y muestra los datos con Flutter

```
FutureBuilder<Post>(
  future: fetchPost(),
  builder: (context, snapshot) {
    if (snapshot.hasData) {
      return Text(snapshot.data.title);
    } else if (snapshot.hasError) {
      return Text("${snapshot.error}");
    }
    // Por defecto, muestra un loading spinner
    return CircularProgressIndicator();
    },
);
```



Future<Post> fetchPost() async { final response = await http.get('https://jsonplaceholder.typicode.com/posts/1'); if (response.statusCode == 200) { return Post. fromJson(json.decode(response.body)); } else { throw Exception('Failed to load post'); class Post { final int userId; final int id; final String title; final String body; Post({this.userId, this.id, this.title, this.body}); factory Post. fromJson (Map<String, dynamic> json) { return Post(userId: json['userId'], id: json['id'], title: json['title'], body: json['body'],);



class MyApp extends StatelessWidget {
 final Future<Post> post;

MyApp({Key key, this.post}) : super(key: key);

Qoverride Widget bui

(,),);

```
Widget build (BuildContext context) {
  return MaterialApp(
   title: 'Fetch Data Example',
   theme: ThemeData(
      primarySwatch: Colors.blue,
    ),
   home: Scaffold(
      appBar: AppBar(
        title: Text('Fetch Data Example'),
      ),
      body: Center(
        child: FutureBuilder<Post>(
         future: post,
         builder: (context, snapshot) {
            if (snapshot.hasData) {
              return Text(snapshot.data.title);
```

```
} else if (snapshot.hasError) {
   return Text("${snapshot.error}");
```

// Por defecto, muestra un loading spinner
return CircularProgressIndicator();





Toma una imagen usando la cámara







camera

Muchas aplicaciones requieren trabajar con las cámaras del dispositivo para tomar fotos y videos.

El plugin **camera** proporciona herramientas para obtener una lista de las cámaras disponibles, mostrar una vista previa que viene de una cámara específica, y tomar fotos o vídeos.

Utiliza 3 dependencias:

- camera Proporciona herramientas para trabajar con las cámaras del dispositivo
- path_provider Encuentra las rutas correctas para almacenar imágenes
- path Crea rutas que funcionan en cualquier plataforma





camera

1. Añadir dependencias

flutter pub add camera
flutter pub add path_provider
flutter pub add path

2. Obtener la lista de las cámaras disponibles

// Obtén una lista de las cámaras disponibles en el dispositivo.
final cameras = await availableCameras();

```
// Obtén una cámara específica de la lista de cámaras disponibles
final firstCamera = cameras.first;
```



. . .

MINISTERIO DE TECNOLOGÍAS DE LA INFORMACIÓN Y LAS COMUNICACIONES



camera

Crear e inicializar el CameraController. 3

class TakePictureScreen extends StatefulWidget final CameraDescription camera;

const TakePictureScreen({super.key, required this.camera});

```
Roverride
  TakePictureScreenState createState() =>
TakePictureScreenState();
```

```
class TakePictureScreenState extends State<TakePictureScreen>
 CameraController controller;
  Future<void> initializeControllerFuture;
```

```
Coverride
void initState()
  super.initState();
   controller = CameraController(
    widget.camera,
    ResolutionPreset.medium,
```

initializeControllerFuture = controller.initialize();

```
@override
void dispose()
  controller.dispose();
  super.dispose();
```

https://esflutter.dev/docs/cookbook/plugins/picture-using-camera





camera

- 4. Usar un **CameraPreview** para mostrar la captura de la cámara **FutureBuilder<void>**(
 - future: _initializeControllerFuture,
 - builder: (context, snapshot) {
 - if (snapshot.connectionState == ConnectionState.done) {
 return CameraPreview(_controller);
 - } else {

},

```
return Center(child: CircularProgressIndicator());
```



Mision tribuida creatigue deretta

camera

```
5.
     Tomar una foto con el CameraController.
FloatingActionButton(
 child: Icon(Icons.camera alt),
 onPressed: () async {
    try {
      await initializeControllerFuture;
      final path = join(
        (await getTemporaryDirectory()).path,
        '${DateTime.now()}.png',
      );
      await controller.takePicture(path);
     catch (e) {
      print(e);
```

6. Muestra la imagen con un widget Image Image.file(File('path/to/my/picture.png'))

https://esflutter.dev/docs/cookbook/plugins/picture-using-camera







Reproducir y pausar un vídeo





VideoPlayer

Reproducir videos es una tarea común en el desarrollo de aplicaciones, y las aplicaciones de Flutter no son la excepción.

Para reproducir videos, el equipo de Flutter proporciona el complemento video_player.

Puedes usar el complemento **video_player** tanto para reproducir videos almacenados en el sistema de archivos, como un asset, o desde internet.

En iOS, el complemento **video_player** hace uso de AVPlayer para manejar la reproducción. En Android, utiliza ExoPlayer.

1. Añadir la dependencia flutter pub add video player





2. Agrega permisos a tu aplicación

Android

Agrega el siguiente permiso a **AndroidManifest.xml** justo después de la definición **<a prime station**.

/android/app/src/main/AndroidManifest.xml

```
<manifest
xmlns:android="http://schemas.android.com
/apk/res/android">
<application ...>
```

</application>

```
<uses-permission
android:name="android.permission.INTERNET"
/>
</manifest>
```

iOS

Para iOS, debe agregar lo siguiente a tu archivo Info.plist.

/ios/Runner/Info.plist





3. Crear e inicializar un VideoPlayerController

class VideoPlayerScreen extends StatefulWidget {
 VideoPlayerScreen({Key key}) : super(key: key);

@override
_VideoPlayerScreenState createState() =>
VideoPlayerScreenState();

class _VideoPlayerScreenState extends State<VideoPlayerScreen> ·
 VideoPlayerController _controller;
 Future<void> initializeVideoPlayerFuture;

@override
void initState() {
 __controller = VideoPlayerController.network(
 'https://flutter.github.io/assets-for-api-docs/assets/

videos/butterfly.mp4');

. . .

_initializeVideoPlayerFuture = _controller.initialize();

super.initState();

@override
void dispose() {
 _controller.dispose();

super.dispose();

@override
Widget build(BuildContext context) {

https://esflutter.dev/docs/cookbook/plugins/play-video





4. Mostrar el reproductor de vídeo

```
FutureBuilder(
  future: initializeVideoPlayerFuture,
 builder: (context, snapshot) {
    if (snapshot.connectionState == ConnectionState.done) {
      return AspectRatio(
        aspectRatio: controller.value.aspectRatio,
        child: VideoPlayer( controller),
     else {
      return Center(child: CircularProgressIndicator());
  },
```





5. Reproducir y pausar el vídeo

```
FloatingActionButton(
  onPressed: ()
    setState(()
      if ( controller.value.isPlaying)
        controller.pause();
       else {
        controller.play();
    });
  },
  child: Icon(
    controller.value.isPlaying ? Icons.pause : Icons.play arrow,
```





Abrir otras aplicaciones del dispositivo





Package of the Week url_launcher

url_launcher





url_launcher

Esquema	Ejemplo	Acción
https: <url></url>	https://flutter.dev	Abra <url> en el navegador predeterminado</url>
mailto: <email address>?subject=<subject>& body=<body></body></subject></email 	mailto:smith@example.org?subje ct=News&body=New%20plugin	Crear correo electrónico para <email address=""> en la aplicación de correo electrónico predeterminada</email>
tel : <phone number=""></phone>	tel:+1-555-010-999	Realice una llamada telefónica a <número de="" teléfono=""> utilizando la aplicación de teléfono predeterminada</número>
sms: <phone number=""></phone>	sms:5550101234	Envía un mensaje SMS a <número de="" teléfono=""> usando la aplicación de mensajería predeterminada</número>
file: <path></path>	file:/home	Abra un archivo o carpeta usando la asociación de aplicaciones predeterminada, compatible con plataformas de escritorio

https://pub.dev/packages/url_launcher





Abrir URL en el Navegador

```
Future<void> launchInBrowser(Uri url) async {
   if (!await launchUrl(
     url,
     mode: LaunchMode.externalApplication,
    throw 'Could not launch $url';
```





Hacer una llamada

```
Future<void> _makePhoneCall(String phoneNumber) async {
   final Uri launchUri = Uri(
      scheme: 'tel',
      path: phoneNumber,
   );
   await launchUrl(launchUri);
}
```







Ubicación en el mundo

Ø

MINISTERIO DE TECNOLOGÍAS DE LA INFORMACIÓN Y LAS COMUNICACIONES









A veces, para proporcionar la mejor experiencia de usuario posible, es necesario conocer la ubicación GPS de su dispositivo.

El paquete Ubicación le permite obtener la ubicación geográfica actual del dispositivo y escuchar los cambios.

¡Puede usar estos datos para mostrar mapas, calcular distancias, determinar la dirección hacia la que mira el dispositivo y más!



- 1. Agregar dependencias flutter pub add location
- 2. Verificar GPS activo

3. Verificar permisos GPS



```
bool _serviceEnabled;
PermissionStatus _permissionGranted;
__serviceEnabled = await location.serviceEnabled();
if (!_serviceEnabled) {
    _serviceEnabled = await location.requestService();
    if (!_serviceEnabled) {
        return;
      }
    }
    _permissionGranted = await location.hasPermission();
    if (_permissionGranted == PermissionStatus.denied) {
        _permissionGranted = await location.requestPermission();
        if (_permissionGranted != PermissionStatus.granted) {
            return;
        }
    }
}
```

https://medium.flutterdevs.com/location-in-flutter-27ca6fa1126c





4. Obtener la posición actual

```
_currentPosition = await location.getLocation();
_initialcameraposition = LatLng( currentPosition.latitude, currentPosition.longitude);
location.onLocationChanged.listen((LocationData currentLocation) {
 print("${currentLocation.longitude} : ${currentLocation.longitude}");
 setState(() {
   _currentPosition = currentLocation;
   __initialcameraposition = LatLng(_currentPosition.latitude,_currentPosition.longitude);
   DateTime now = DateTime.now();
   _dateTime = DateFormat('EEE d MMM kk:mm:ss ').format(now);
   _getAddress(_currentPosition.latitude, _currentPosition.longitude)
        .then((value) {
     setState(() {
        _address = "${value.first.addressLine}";
     });
   });
 });
});
```

https://medium.flutterdevs.com/location-in-flutter-27ca6fa1126c







5. Obtener la dirección desde las coordenadas

```
Future<List<Address>> _getAddress(double lat, double lang) async {
   final coordinates = new Coordinates(lat, lang);
   List<Address> add = await Geocoder.local.findAddressesFromCoordinates(coordinates);
   return add;
```

6. Mostrar la dirección y coordenadas

```
Text(
    "Latitude: ${_currentPosition.latitude}, Longitude:
${_currentPosition.longitude}",
    style: TextStyle(
        fontSize: 22,
            color: Colors.white,
        fontWeight: FontWeight.bold),
    ),
    if (_address != null)
    Text(
        "Address: $_address",
        style: TextStyle(
        fontSize: 16,
        color: Colors.white,
        ),
```

https://medium.flutterdevs.com/location-in-flutter-27ca6fa1126c





7. Abrirlo en Google Maps

