08

메시지 채널

네이티브 연동

플랫폼 채널이란?

- 플랫폼별 기능을 흔히 네이티브native 기능이라고 하는데 플러터에서는 이 네이티브 기능을 API로 제공하지 않습니다.
- 플러터에서는 개발자가 작성한 네이티브 코드와 다트 코드가 서로 연동할 수 있는 채널만 제공

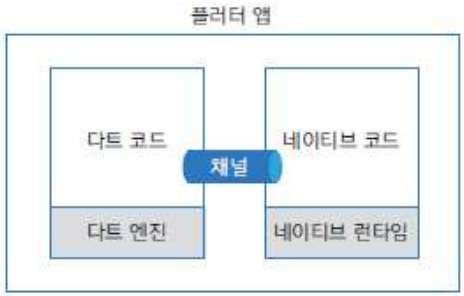

그림 22-1 플랫폼 채널

메시지 채널 이용하기

다트에서 네이티브로 보내기

- 다트 코드
  - 플러터는 BasicMessageChannel 클래스를 제공
  - 생성자의 첫 번째 매개변수로 지정한 문자열은 채널의 이름
  - 두 번째 매개변수에 지정한 StringCodec() 함수는 다트와 네이티브 간에 주고받는 바이너리와 문자열을 변환
  - 채널을 만들었으면 채널 객체의 send() 함수로 문자열 데이터를 네이티브에 보냅니다.
  - 네이티브에서 문자열 데이터를 받아서 로직을 실행한 후 결괏값을 send() 함수의 반환 타입으로 받을 수 있습니다.

• 메시지 채널로 네이티브에 데이터 보내기(다트)

```
const channel = BasicMessageChannel<String>('myMessageChannel', StringCodec());
String? result = await channel.send('Hello from Dart');
```

메시지 채널 이용하기

다트에서 네이티브로 보내기

- 안드로이드 코드
  - MainActivity에 configureFlutterEngine() 함수를 재정의
  - 다트 코드에 지정한 채널과 같은 이름으로 채널을 생성
  - 다트에서 보내는 메시지를 받으려고 setMessageHandler() 함수의 매개변수에 데이터가 전달될 때 자동으로 호출할 함수를 지정
  - message 매개변수는 다트에서 보낸 문자열
  - reply 매개변수는 로직을 실행한 후 결과를 보낼 때 사용

• 다트에서 보낸 데이터 받기(코틀린)

```
class MainActivity: FlutterActivity() {
  override fun configureFlutterEngine(flutterEngine: FlutterEngine) {
    GeneratedPluginRegistrant.registerWith(flutterEngine)

    val channel = BasicMessageChannel<String>(flutterEngine.dartExecutor,
        "myMessageChannel", StringCodec.INSTANCE )
    channel.setMessageHandler { message, reply ->
        Log.d("msg", "receive: $message")
        reply.reply("Reply from Android")
    }
  }
}
```

메시지 채널 이용하기

다트에서 네이티브로 보내기

- iOS 코드
  - App Delegate 클래스에 application() 함수가 선언되어 있으며 이 함수에 FlutterBasicMessage Channel 클래스의 객체를 생성.
  - 객체의 name에 지정한 문자열이 채널 이름이며 다트에서 지정한 채널 이름과 같아야 합니다.
  - setMessageHandler() 함수의 매개변수에 문자열을 받을 때 자동으로 실행할 함수를 지정
  - 첫 번째 매개변수는 다트에서 보낸 데이터
  - 두 번째 매개변수는 실행 결과를 다시 다트에 보내는 FlutterReply 객체

• 다트에서 보낸 데이터 받기(스위프트)

```
@UIApplicationMain
@objc class AppDelegate: FlutterAppDelegate {
  override func application(
    _ application: UIApplication,
    didFinishLaunchingWithOptions launchOptions: [UIApplication.LaunchOptionsKey: Any]?
  ) -> Bool {
    let controller: FlutterViewController =
        window?.rootViewController as! FlutterViewController
    let channel = FlutterBasicMessageChannel(
        name: "myMessageChannel",
        binaryMessenger: controller.binaryMessenger,
        codec: FlutterStringCodec.sharedInstance()
    )

    channel.setMessageHandler {
      (message: Any?, reply: FlutterReply) -> Void in reply("Hi from iOS")
    }

    GeneratedPluginRegistrant.register(with: self)
    return super.application(application, didFinishLaunchingWithOptions: launchOptions)
  }
}
```

메시지 채널 이용하기

네이티브에서 다트로 보내기

- 다트 코드
  - BasicMessageChannel을 이용해 채널을 만들고 데이터가 전달될 때 호출할 함수를 setMessageHandler() 함수의 매개변수로 지정

• 네이티브에서 메시지 채널로 보낸 데이터 받기(다트)

```
const channel = BasicMessageChannel<String>('myMessageChannel', StringCodec());
channel.setMessageHandler((String? message) async {
  ... (생략) ...
  return 'Reply from Dart';
});
```

메시지 채널 이용하기

네이티브에서 다트로 보내기

- 안드로이드 코드
  - BasicMessage Channel 객체로 채널을 만들고 이 채널 객체의 send() 함수로 다트에 문자열을 보냅니다.

• 안드로이드에서 메시지 보내기(코틀린)

```
val channel = BasicMessageChannel<String>(flutterEngine.dartExecutor,
    "myMessageChannel", StringCodec.INSTANCE )
... (생략) ...
channel.send("Hello from Android"){replay -> Log.d("msg", "reply : $replay")
```

메시지 채널 이용하기

네이티브에서 다트로 보내기

- iOS 코드
  - FlutterBasicMessageChannel을 이용해 채널 객체를 만들고 이 객체의 sendMessage() 함수를 호출하여 다트에 문자열을 보냅니다.

• iOS에서 메시지 보내기(스위프트)

```
let controller: FlutterViewController =
    window?.rootViewController as! FlutterViewController
let channel = FlutterBasicMessageChannel(
    name: "myMessageChannel",
    binaryMessenger: controller.binaryMessenger,
    codec: FlutterStringCodec.sharedInstance()
)
... (생략) ...
channel.sendMessage("Hello i am ios native") {
  (reply: Any?) -> Void in print("%@", reply as! String)
}
```

감사합니다

단단히 마음먹고 떠난 사람은
산꼭대기에 도착할 수 있다.
산은 올라가는 사람에게만 정복된다.

윌리엄 셰익스피어
William Shakespeare