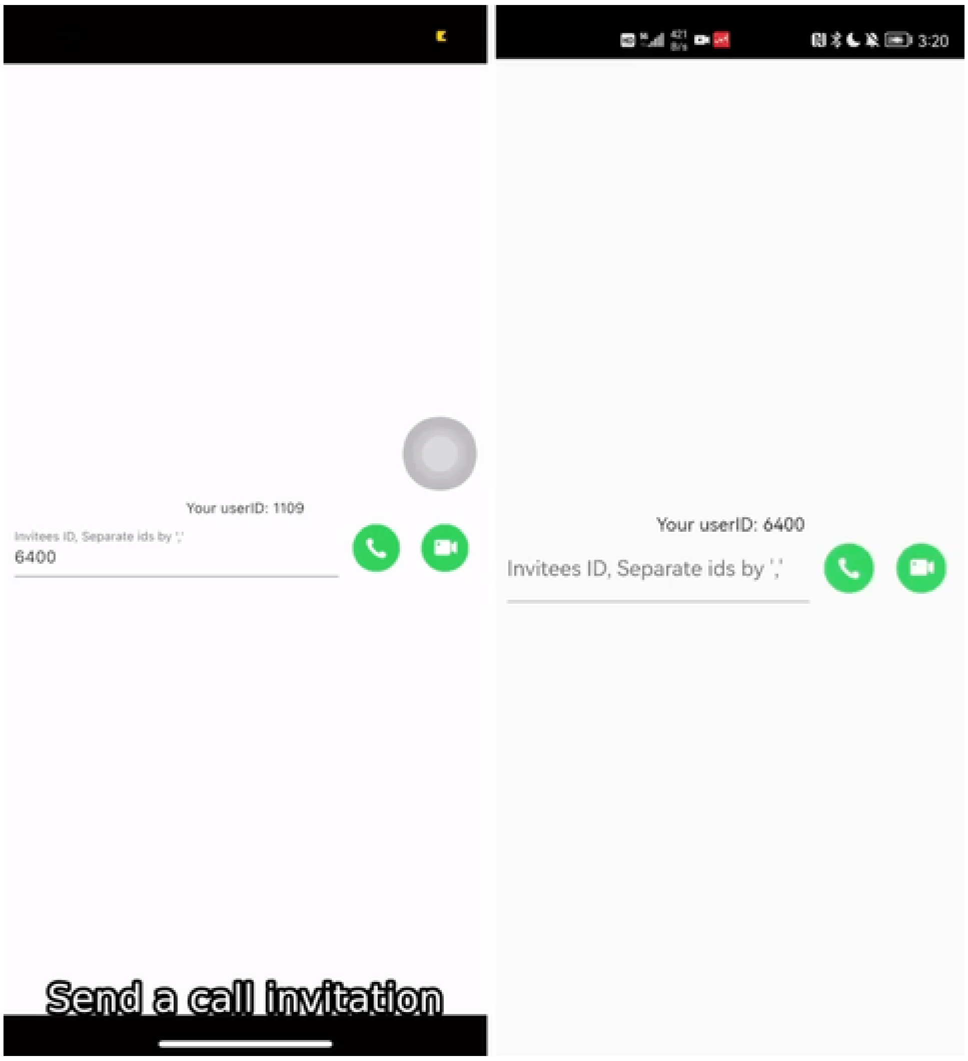
1. What is the Call Kit

`Call Kit` is a prebuilt feature-rich call component, which enables you to build `one-on-one` and `group voice/video` calls into your app with only a few lines of code. And it includes the business logic with the UI, you can add or remove features accordingly by customizing UI components.

2. Quick start with call invitation


Your userID: 1109
Invitees ID, Separate ids by ','
6400
Send a call invitation
3:20
Your userID: 6400
Invitees ID, Separate ids by ','

Preview：https://storage.zego.im/sdk-doc/Pics/ZegoUIKit/Flutter/call/invitation_calls.gif

To integrate the calling feature with invitation function into your app, the following steps are required: 1. Create a project in the ZEGOCLOUD console and obtain the `AppID` and `AppSign`; 2. Add ZegoUIKitPrebuiltCall as a dependency; 3. Import the SDK; 4. Integrate the SDK; 5. Configure your project; 6. Enable offline call invitation. You can refer this doc for all details: https://doc-preview-en.zego.im/article/14826.

如何使用requireConfig方法

requireConfig是ZegoUIKitPrebuiltCallInvitationService().init的一个参数，类型是一个参数为`(ZegoCallInvitationData data)`的回调方法。你可以根据自己的业务需求在此方法做对应设置。比如设置底部按钮、开关设备等等。你需要像下面示例代码一样使用它：

```
ZegoUIKitPrebuiltCallInvitationService().init(
  appID: YourAppID,
  appSign: YourAppSign,
  userID: userID,
  userName: userName,
  plugins: [ZegoUIKitSignalingPlugin()],
  requireConfig: (ZegoCallInvitationData data) {
    // 处理你的业务逻辑
    // 这个方法需要返回一个 ZegoUIKitPrebuiltCallConfig 实例
    // var config = (data.invitees.length > 1)
    // ? ZegoCallType.videoCall == data.type
    //     ? ZegoUIKitPrebuiltCallConfig.groupVideoCall()
    //     : ZegoUIKitPrebuiltCallConfig.groupVoiceCall()
    // : ZegoCallType.videoCall == data.type
    //     ? ZegoUIKitPrebuiltCallConfig.oneOnOneVideoCall()
    //     : ZegoUIKitPrebuiltCallConfig.oneOnOneVoiceCall();
    // return config;
  },
);
```

Set avatar for users

在语音通话情况下，默认会把你的用户名首字母作为头像显示。如果你想更改头像，比如使用图片来显示头像，那么你可以在`requireConfig`方法中，通过`config.avatarBuilder`返回一个`NetworkImage` widget来实现。以下是示例代码：

```
requireConfig: (ZegoCallInvitationData data) {
  var config = ZegoUIKitPrebuiltCallConfig.oneOnOneVoiceCall();
  config.avatarBuilder = (BuildContext context, Size size, ZegoUIKitUser? user, Map extraInfo) {
    return user != null
        ? Container(
            decoration: BoxDecoration(
              shape: BoxShape.circle,
              image: DecorationImage(
                image: NetworkImage(
                  'https://your_server/app/avatar/${user.id}.png',
                ),
              ),
            ),
          )
        : const SizedBox();
  };
```

```
    return config;
  },
```

Add custom components to user view

Customize the foreground view

If you want to add some custom components at the top level of the view, such as, you want to add user-level icons, etc., then you can use `foregroundBuilder` in `config.audioVideoViewConfig`. This callback will be triggered after corresponding parameter changes. This callback, similar to other Flutter's Builder callbacks, requires return a custom Widget that will be placed at the top of the view. The position of the Widget can be specified by using the Flutter `Positioned`.

下面是示例代码:

```
requireConfig: (ZegoCallInvitationData data) {
    var config = ZegoUIKitPrebuiltCallConfig.groupVideoCall();

    // Modify your custom configurations here.
    config.audioVideoViewConfig =
ZegoPrebuiltAudioVideoViewConfig(foregroundBuilder: (BuildContext context,
Size size, ZegoUIKitUser? user, Map extraInfo) {
        return user != null
            ? Positioned(
                bottom: 5,
                left: 5,
                child: Container(
                  width: 30,
                  height: 30,
                  decoration: BoxDecoration(
                    shape: BoxShape.circle,
                    image: DecorationImage(
                      image: NetworkImage(

'https://your_server/app/level_icon/${getUserLevelByID(user.id)}.png',
                      ),
                    ),
                  ),
                ),
              )
            : const SizedBox();
      },
    );
    return config;
  },
```

Customize the audio view

在语音通话时默认画面会显示用户头像和一个纯色背景。如果你希望更改背景，比如替换成图片或者更改其颜色，那么你可以在`config.audioVideoViewConfig`的`backgroundBuilder`回调中返回一个有效的自定义Widget即可。这个方法在对应的参数变化后都会触发。

下面是示例代码：

```
  requireConfig: (ZegoCallInvitationData data) {
    var config = ZegoUIKitPrebuiltCallConfig.oneOnOneVoiceCall();

    // Modify your custom configurations here.
    config.audioVideoViewConfig =
ZegoPrebuiltAudioVideoViewConfig(backgroundBuilder: (BuildContext context,
Size size, ZegoUIKitUser? user, Map extraInfo) {
        return user != null
            ? ImageFiltered(
                imageFilter: ImageFilter.blur(sigmaX: 5, sigmaY: 5),
                child: Image(
                  image: NetworkImage(
                    'https://your_server/app/user_image/${user.id}.png',
                  ),
                ),
              )
            : const SizedBox();
      },
    );
    return config;
  },
```

Configure layouts

如果你需要更改画面布局，你可以通过`config.layout`来更改。目前支持`picture-in-picture`和`gallery`布局。

Picture-in-picture layout

picture-in-picture的效果就是全屏显示一个大的画面，然后其他人的画面悬浮在该大画面上显示成小画面。picture-in-picture支持两个参数：`isSmallViewDraggable`表示布局中的小View是否可以被拖动，默认是允许的；`switchLargeOrSmallViewByClick`表示是否能通过点击小View来切换大小View的画面，默认是允许的。下面是如何使用picture-in-picture布局的示例代码：

```
  requireConfig: (ZegoCallInvitationData data) {
    var config = ZegoUIKitPrebuiltCallConfig.oneOnOneVideoCall();

    // Modify your custom configurations here.
    config.layout = ZegoLayout.pictureInPicture(
      isSmallViewDraggable: true,
      switchLargeOrSmallViewByClick: true,
    );
    return config;
  },
```

Gallery layout

gallery布局的效果是每个人的画面会平局平铺在手机屏幕上，最多支持显示8个人的画面，超过8人后其他人会在一个画面中显示“Others”来表示。gallery布局支持`addBorderRadiusAndSpacingBetweenView`参数，表示是否在View之间添加圆角和间隙，这个值默认为true。以下是使用gallery布局的示例代码：

```
  requireConfig: (ZegoCallInvitationData data) {
    var config = ZegoUIKitPrebuiltCallConfig.groupVideoCall();

    // Modify your custom configurations here.
    config.layout = ZegoLayout.gallery(
      addBorderRadiusAndSpacingBetweenView: false,
    );
    return config;
  },
```

Implement an audio-only(aka voice-only) call

When starting a call, the Call Kit (ZegoUIKitPrebuiltCall) turns on the camera, and microphone, and uses the speaker as the audio output device by default.

To change this default configuration, for example, turn off the camera when you start a call or don't use the speaker (If the speaker is not used, the system's default audio output device, such as ear speaker, headset, Bluetooth, etc., will be used.), you can modify the following configurations:

1. `turnOnCameraWhenJoining`: Whether to turn on the camera when the call starts. true: turn on (by default). false: turn off.
2. `turnOnMicrophoneWhenJoining`: Whether to turn on the microphone when the call starts. true: turn on (by default). false: turn off.
3. `useSpeakerWhenJoining`: Whether to use the speaker when the call starts. true: use the speaker (by default). false: use the system's default audio output device, such as an ear speaker, headset, Bluetooth, etc.

Here is the reference code of how to implement an audio-only call:

```
  requireConfig: (ZegoCallInvitationData data) {
    var config = ZegoUIKitPrebuiltCallConfig.oneOnOneVoiceCall();

    // Modify your custom configurations here.
    config
      ..turnOnCameraWhenJoining = false
      ..turnOnMicrophoneWhenJoining = false
      ..useSpeakerWhenJoining = true;
    return config;
  },
```

Customize the menu bar

Add buttons on the menu bar

如果你想修改底部栏的按钮排序，或者删除上面的上面的按钮、往上面添加你自己的按钮，你都可以通过`config.topMenuBarConfig`参数来做修改。你可以往`config.topMenuBarConfig.buttons`设置我们提供的几个默认按钮枚举值，设置了就会显示。然后可以通过`config.topMenuBarConfig.extendButtons`参数来添加你自己的按钮。最后你可以通过设置`config.topMenuBarConfig.maxCount`参数来控制底部工具栏最多显示几个按钮，默认值是5，有效范围是[1-5]个，当按钮数量超过`maxCount`值，就会在最右边显示有三个点的按钮(⁝)，点击该按钮会弹出一个view以显示其他没有在底部工具栏显示的按钮。

以下是如何更改底部工具栏按钮的示例代码：

```
requireConfig: (ZegoCallInvitationData data) {
    var config = ZegoUIKitPrebuiltCallConfig.oneOnOneVoiceCall();

    // Modify your custom configurations here.
    config.bottomMenuBarConfig = ZegoBottomMenuBarConfig(
      maxCount: 5,
      extendButtons: [
        ElevatedButton(
            style: ElevatedButton.styleFrom(
              fixedSize: const Size(60, 60),
              shape: const CircleBorder(),
              primary: controlBarButtonCheckedBackgroundColor,
            ),
            onPressed: () {},
            child: Icon(Icons.file_copy),
        ),
      ],
      buttons: [
        ZegoMenuBarButtonName.toggleCameraButton,
        ZegoMenuBarButtonName.toggleMicrophoneButton,
        ZegoMenuBarButtonName.switchAudioOutputButton,
        ZegoMenuBarButtonName.hangUpButton,
        ZegoMenuBarButtonName.switchCameraButton,
      ],
    );
    return config;
  },
```

Customize the hidden behavior of the menu bar

如果有需要，你还可以通过`config.topMenuBarConfig.hideByClick`和`config.topMenuBarConfig.hideAutomatically`参数来控制底部工具栏的隐藏和显示的行为。`hideByClick`参数可以控制工具栏是否可以通过用户点击屏幕隐藏，默认是`true`。`hideAutomatically`参数控制工具栏是否在没有用户操作5秒后自动隐藏，默认是`true`.

Set a hangup confirmation dialog

如果你想在结束通话前（比如用户按下结束通话按钮或者Android系统的返回键）可以给用户弹框确认是否结束通话，你可以通过`config.hangUpConfirmInfo`参数来设置。 下面是示例代码：

```
  requireConfig: (ZegoCallInvitationData data) {
    var config = ZegoUIKitPrebuiltCallConfig.oneOnOneVoiceCall();

    // Modify your custom configurations here.
    config.hangUpConfirmDialogInfo = ZegoHangUpConfirmDialogInfo(
      title: "Hangup confirm",
      message: "Do you want to hangup?",
      cancelButtonName: "Cancel",
      confirmButtonName: "Confirm",
    );
    return config;
  },
```

如果你希望在通话前不仅仅弹一个简单的对话框，而是需要做更多复杂的业务逻辑，比如向后台更新一些记录，那么你可以通过`config.onHangUpConfirmation`参数来设置。这个参数要求你提供一个方法，该方法需要返回一个异步结果。下面是示例代码:

```
  requireConfig: (ZegoCallInvitationData data) {
    var config = ZegoUIKitPrebuiltCallConfig.oneOnOneVoiceCall();

    // Modify your custom configurations here.
    config.onHangUpConfirmation = (BuildContext context) async {
      // 向服务器请求更新操作等业务逻辑代码
      return await showDialog(
        context: context,
        barrierDismissible: false,
        builder: (BuildContext context) {
          return AlertDialog(
            title: const Text("This is your custom dialog"),
            content: const Text("You can customize this dialog however you
like"),
            actions: [
              ElevatedButton(
                child: const Text("Cancel"),
                onPressed: () => Navigator.of(context).pop(false),
              ),
              ElevatedButton(
                child: const Text("Exit"),
                onPressed: () => Navigator.of(context).pop(true),
              ),
            ],
          );
        },
      );
    };
    return config;
  },
```

如果你想在通话结束后再弹框或者跳转到其他页面显示本次通话时长和计费信息等，你可以通过设置`config.hangup`这个参数来监听通话结束的事件，然后在里做你的业务逻辑，展示信息等操作。

Call invitation config

Customize the call ringtone

Ringtone for call invitation

App在前台运行时收到呼叫邀请会弹框并响铃，如果你想更改App在前台时的响铃，可以通过`ZegoUIKitPrebuiltCallInvitationService().init`方法的`ringtoneConfig`参数配置。下面是示例代码：

```
ZegoUIKitPrebuiltCallInvitationService().init(
  appID: YourAppID,
  appSign: YourAppSign,
  userID: userID,
  userName: userName,
  plugins: [ZegoUIKitSignalingPlugin()],
  // Modify your custom configurations here.
  ringtoneConfig: const ZegoRingtoneConfig(
    incomingCallPath: "assets/ringtone/incomingCallRingtone.mp3",
    outgoingCallPath: "assets/ringtone/outgoingCallRingtone.mp3",
  ),
);
```

Ringtone for offline call invitation

App在后台或者被杀死后，收到呼叫邀请时的通知响铃是通过ZEGOCLOUD控制台(https://console.zegocloud.com/)配置的。在控制台`Projects Management>In-app Chat`添加一个`Push Resource ID`，在添加时如果选的是Normal类型则可以指定响铃的音频文件。在配置好后，将该`Push Resource ID`设置到`ZegoSendCallInvitationButton`的`resourceID`参数。以下是示例代码：

```
ZegoSendCallInvitationButton(
   isVideoCall: true,
   resourceID: "zegouikit_call",
   invitees: [
      ...
    ],
)
```

Hide the decline button

如果你想用户在接到呼叫邀请时不显示拒接按钮，那么可以将`ZegoUIKitPrebuiltCallInvitationService().init`方法的`showDeclineButton`参数设置为`false`。以下是示例代码：

```
ZegoUIKitPrebuiltCallInvitationService().init(
  appID: YourAppID,
  appSign: YourAppSign,
  userID: userID,
  userName: userName,
  plugins: [ZegoUIKitSignalingPlugin()],
  showDeclineButton: false,
);
```

Auto re-send call invitation after call timeout

有时候你在被呼叫人没有接听电话的时候可能想自动重拨电话（或者叫重新发送通话邀请）。要实现这个功能，你需要完成两个步骤：第一步是监听`ZegoUIKitPrebuiltCallInvitationEvents`的`onOutgoingCallTimeout`回调，这个回调会在通话邀请超时后触发；第二步是调用`ZegoUIKitPrebuiltCallController`的`sendCallInvitation`方法重新发送通话邀请。

下面代码展示了如何监听`onOutgoingCallTimeout`回调：

```
ZegoUIKitPrebuiltCallInvitationService().init(
    appID: YourAppID,
    appSign: YourAppSign,
    userID: userID,
    userName: userName,
    notifyWhenAppRunningInBackgroundOrQuit: false,
    plugins: [ZegoUIKitSignalingPlugin()],
    controller: callInvitationController,
    events: ZegoUIKitPrebuiltCallInvitationEvents(
      onOutgoingCallTimeout: (String callID, List<ZegoCallUser> callees,
bool isVideoCall, ) async {
        // 在这里做重新发送通话邀请的逻辑
      },
    ),
  );
```

下面代码展示了如何调用`sendCallInvitation`方法：

```
onOutgoingCallTimeout: (String callID, List<ZegoCallUser> callees, bool
isVideoCall, ) async {
  await callInvitationController.sendCallInvitation(
    callID: callID,
    invitees: callees.map((callee) => ZegoCallUser(callee.id,
'user_${callee.id}')).toList(),
    isVideoCall: isVideoCall,
  );
},
```

Modify User Interface text

如果你想更改UI上的文本，比如把接受和拒绝按钮文本替换成其他语言，那么你可以通过`ZegoUIKitPrebuiltCallInvitationService().innerText`参数来更改。以下是示例代码：

```
  ZegoUIKitPrebuiltCallInvitationService()
    ..innerText.incomingCallPageAcceptButton = "Accept"
    ..innerText.incomingCallPageDeclineButton = "Decline";
```

Listen for call invitation event callbacks

You can implement further business logic by listening for event callbacks related to the call invitation.

The following are supported event callbacks:

- Function() `onIncomingCallDeclineButtonPressed`: This callback will be triggered when the Decline button is pressed (the callee declines the call invitation).
- Function() `onIncomingCallAcceptButtonPressed`: This callback will be triggered when the Accept button is pressed (the callee accepts the call invitation).
- Function(String callID, ZegoCallUser caller, ZegoCallType callType, List< ZegoCallUser > callees) `onIncomingCallReceived`: This callback will be triggered when receiving call invitations.
- Function(String callID, ZegoCallUser caller) `onIncomingCallCanceled`: This callback will be triggered when the caller cancels the call invitation.
- Function(String callID, ZegoCallUser caller) `onIncomingCallTimeout`: The callee will receive a notification through this callback when the callee doesn't respond to the call invitation after a timeout duration.
- Function() `onOutgoingCallCancelButtonPressed`: This callback will be triggered when the Cancel button is pressed (the caller cancels the call invitation).
- Function(String callID, ZegoCallUser callee) `onOutgoingCallAccepted`: The caller will receive a notification through this callback when the callee accepts the call invitation.

- Function(String callID, ZegoCallUser callee) `onOutgoingCallRejectedCauseBusy`: The caller will receive a notification through this callback when the callee rejects the call invitation (the callee is busy).
- Function(String callID, ZegoCallUser callee) `onOutgoingCallDeclined`: The caller will receive a notification through this callback when the callee declines the call invitation actively.
- Function(String callID, List< ZegoCallUser > callees) `onOutgoingCallTimeout`: The caller will receive a notification through this callback when the call invitation didn't get responses after a timeout duration.

Here is the reference code of how to use the callback functions:

```
ZegoUIKitPrebuiltCallInvitationService().init(
  appID: YourAppID,
  appSign: YourAppSign,
  userID: userID,
  userName: userName,
  events: ZegoUIKitPrebuiltCallInvitationEvents(
    onIncomingCallDeclineButtonPressed: () {
      ///  Add your custom logic here.
    },
    onIncomingCallAcceptButtonPressed: () {
      ///  Add your custom logic here.
    },
    onIncomingCallReceived: (String callID, ZegoCallUser caller,
        ZegoCallType callType, List<ZegoCallUser> callees) {
      ///  Add your custom logic here.
    },
    onIncomingCallCanceled: (String callID, ZegoCallUser caller) {
      ///  Add your custom logic here.
    },
    onIncomingCallTimeout: (String callID, ZegoCallUser caller) {
      ///  Add your custom logic here.
    },
    onOutgoingCallCancelButtonPressed: () {
      ///  Add your custom logic here.
    },
    onOutgoingCallAccepted: (String callID, ZegoCallUser callee) {
      ///  Add your custom logic here.
    },
    onOutgoingCallRejectedCauseBusy: (String callID, ZegoCallUser callee)
{
      ///  Add your custom logic here.
    },
    onOutgoingCallDeclined: (String callID, ZegoCallUser callee) {
      ///  Add your custom logic here.
    },
    onOutgoingCallTimeout: (String callID, List<ZegoCallUser> callees) {
      ///  Add your custom logic here.
    },
  ),
  plugins: [ZegoUIKitSignalingPlugin()],
);
```

最小化通话窗口

最小化窗口效果就是在你通话过程中，可以将通话窗口变成一个可以在app内拖动的小窗口。这个时候你可以在保持通话的情况下操作app的其他功能，比如查看用户信息，查看账单等等。如果你要实现通话窗口最小化的效果，那么你有两个步骤需要完成。第一步是在`requireConfig`回调中，将`ZegoMenuBarButtonName.minimizingButton`添加到`ZegoUIKitPrebuiltCallConfig`的`topMenuBarConfig`配置项中；第二步是在`MaterialApp`的`builder`方法中将`ZegoUIKitPrebuiltCallMiniOverlayPage`组件插入`Stack`的children中返回。

下面的代码演示了如何将`ZegoMenuBarButtonName.minimizingButton`添加到`ZegoUIKitPrebuiltCallConfig`的`topMenuBarConfig`配置项中：

```
    requireConfig: (ZegoCallInvitationData data) {
      var config = (data.invitees.length > 1)
          ? ZegoCallType.videoCall == data.type
          ? ZegoUIKitPrebuiltCallConfig.groupVideoCall()
          : ZegoUIKitPrebuiltCallConfig.groupVoiceCall()
          : ZegoCallType.videoCall == data.type
          ? ZegoUIKitPrebuiltCallConfig.oneOnOneVideoCall()
          : ZegoUIKitPrebuiltCallConfig.oneOnOneVoiceCall();
      /// show minimizing button
      config.topMenuBarConfig.isVisible = true;
      config.topMenuBarConfig.buttons.insert(0,
  ZegoMenuBarButtonName.minimizingButton);

      return config;
    },
```

下面代码演示了如何将`ZegoUIKitPrebuiltCallMiniOverlayPage`组件插入`MaterialApp`的children中：

```
  ///  最小化通话窗口: Step 1/3: Declare a NavigatorState
  final navigatorKey = GlobalKey<NavigatorState>();

  class MyApp extends StatelessWidget {
    const MyApp({Key? key}) : super(key: key);
    @override
    Widget build(BuildContext context) {
      return MaterialApp(
        /// 最小化通话窗口:  Step 2/3: Assign the NavigatorState to MaterialApp
        navigatorKey: navigatorKey,
        builder: (BuildContext context, Widget? child) {
          return Stack(
            children: [
              child!,
              ///  最小化通话窗口: Step 3/3: Insert
  ZegoUIKitPrebuiltCallMiniOverlayPage into Overlay, and return the context
  of NavigatorState in contextQuery.
              ZegoUIKitPrebuiltCallMiniOverlayPage(
                contextQuery: () {
                  return navigatorKey.currentState!.context;
                },
              ),
            ],
          );
        },
      );
    }
  }

  void main() async {
    WidgetsFlutterBinding.ensureInitialized();

    ZegoUIKit().initLog().then((value) {
      runApp(const MyApp());
    });
  }
```