7.0.0

This major version enhances the support for Multi-Conversations and introduces Multi-Party where conversations can now host multiple users messaging each other. Read about Multi-Party Conversations on our guide.

Many APIs have also been reworked to offer better type and nullability safety, improving the overall usability of the SDK.

Behavior Changes

Initializing the SDK

The new version of the SDK requires an `Integration ID` instead of the `App ID` to be initialized. You can find your `Integration ID` in the Sunshine Conversations dashboard when connecting an `Android SDK` integration as a `Customer Channel` to your Sunshine Conversations app.

Another new aspect of the initialization step is that `SmoochCallback` is now typed and annotated `@NonNull` for better type and nullability safety.

```
// Before
public class YourApplication extends Application {

    @Override
    public void onCreate() {
        super.onCreate();

        Settings settings = new Settings("your_app_id");
        Smooch.init(this, settings, new SmoochCallback() {
            @Override
            public void run(Response response) {
                // Handle the response...
                InitializationStatus status = (InitializationStatus) re
                if (status == InitializationStatus.Success) {
                    // Smooch is ready to use
                }
            }
        });
    }
}
```

```
// After
public class YourApplication extends Application {

    @Override
    public void onCreate() {
        super.onCreate();

        Settings settings = new Settings("your_integration_id");
        Smooch.init(this, settings, new SmoochCallback<InitializationSt
            @Override
            public void run(@NonNull Response<InitializationStatus> res
                // Handle the response, no casting required...
                if (response.getData() == InitializationStatus.SUCCESS)
                    // Smooch is ready to use
                }
            }
        });
    }
}
```

Setting Delegates

The delegates are no longer nested in `Conversation` and are set directly through `Smooch`.

```
// Before
public class YourClass implements Conversation.Delegate, Conversation.M

    public void foo() {
        // With Smooch initialized...

        Conversation conversation = Smooch.getConversation();
        if (conversation != null) {
            conversation.setDelegate(this);
            conversation.setViewDelegate(this);
            conversation.setMessageModifierDelegate(this);
        }
    }
}

// After
public class YourClass implements ConversationDelegate, MessageModifier

    public void foo() {
        // With Smooch initialized...

        Smooch.setConversationDelegate(this);
        Smooch.setConversationViewDelegate(this);
        Smooch.setMessageModifierDelegate(this);
```

```
    }
}
```

Retrieving other conversations

You can now retrieve the list of conversations of the current user:

```
public class YourClass {

    public void foo() {
        // With Smooch initialized...

        Smooch.getConversationsList(new SmoochCallback<List<Conversatio
            @Override
            public void run(@NonNull Response<List<Conversation>> respo
                List<Conversation> conversations = response.getData();
            }
        });
    }
}
```

You can also retrieve any conversation given its ID:

```
public class YourClass {

    public void foo() {
        // With Smooch initialized...

        Smooch.getConversation("conversation_id", new SmoochCallback<Co
            @Override
            public void run(@NonNull Response<Conversation> response) {
                Conversation conversation = response.getData();
            }
        });
    }
}
```

Note that you still have to load a conversation to perform any actions on it:

```
public class YourClass {

    public void foo() {
        // With Smooch initialized...

        Smooch.loadConversation("conversation_to_load", new SmoochCallb
            @Override
```

```
            public void run(@NonNull Response<Conversation> response) {
                // The newly loaded conversation
                Conversation conversation = response.getData();

                // Subsequent calls to Smooch.getConversation() will al
            }
        });
    }
}
```

Showing the Conversation UI

The `ConversationActivity` was updated to use a `Builder` instead of overloaded methods.

```
// Before
public class YourActivity extends AppCompatActivity {

    public void foo() {
        // With Smooch initialized...

        ConversationActivity.show(this, getString(R.string.your_smooch_
    }
}

// After
public class YourActivity extends AppCompatActivity {

    public void foo() {
        // With Smooch initialized...

        ConversationActivity.builder()
            .withStartingText(getString(R.string.your_smooch_starting_t
            .show(this);
    }
}
```

New Multi-Conversation and Multi-Party events

`ConversationEvent` and `ConversationEventType` have been updated with new getters and values to support Multi-Conversation and Multi-Party events:

```
public class YourClass implements ConversationDelegate {

    public void foo() {
        // With Smooch initialized...
```

```
        Smooch.setConversationDelegate(this);
    }

    ...

    @Override
    public void onConversationEventReceived(@NonNull ConversationEvent

        // New getter to know the conversation ID which this event rela
        String conversationId = conversationEvent.getConversationId();

        // getType() was updated to return ConversationEventType instea
        switch (conversationEvent.getType()) {
            case TYPING_START:
                // Updated to include getAppUserId() and getUserId() wh
                break;
            case TYPING_STOP:
                // Updated to include getAppUserId() and getUserId() wh
                break;
            case CONVERSATION_READ:
                // Updated to include getAppUserId() and getUserId() wh

                // This date is no longer limited to business events. U
                // to distinguish between business and user events
                Date lastRead = conversationEvent.getLastRead();
                break;
            case CONVERSATION_ADDED:
                // New: current user was added to a conversation
                break;
            case CONVERSATION_REMOVED:
                // New: current user was removed from a conversation
                break;
            case PARTICIPANT_ADDED:
                // New: another user was added to a conversation the cu
                break;
            case PARTICIPANT_REMOVED:
                // New: another user was added to a conversation the cu
                break;

        }
    }
}
```

Please refer to the Javadoc for the full specification of each event.

API Changes (Breaking)

1. Class Settings

- Replaced constructor `Settings(String appId)` with `Settings(String integrationId)`
- Replaced constructor `Settings(String appId, String authCode)` with `Setting(String integrationId, String authCode)`
- Removed method `getAppId(): String`

2. Class Conversation

- Changed from `Class` to `Interface`
- Replaced nested interface `Delegate` with new interface `ConversationDelegate`
- Replaced nested interface `MessageModifierDelegate` with new interface `MessageModifierDelegate`
- Replaced nested interface `ViewDelegate` with new interface `ConversationViewDelegate`
- Replaced method `setDelegate(Delegate)` with `Smooch#setConversationDelegate(ConversationDelegate)`
- Replaced method `setMessageModifierDelegate(MessageModifierDelegate)` with `Smooch#setMessageModifierDelegate(MessageModifierDelegate)`
- Replaced method `setViewDelegate(ViewDelegate)` with `Smooch#setConversationViewDelegate(ConversationViewDelegate)`
- Removed method `getDelegate(): Delegate`
- Removed method `getMessageModifierDelegate(): MessageModifierDelegate()`
- Removed method `getViewDelegate(): ViewDelegate`
- Removed method `markAsRead(Message)`
- Removed method `setSmoochUIDelegate(Delegate)`

3. Class ConversationEvent

- Renamed `getAppMakerLastRead(): Date` to `getLastRead(): Date`

4. Enum ConversationEventType

- Renamed `TypingStart` to `TYPING_START`
- Renamed `TypingStop` to `TYPING_STOP`
- Renamed `ConversationRead` to `CONVERSATION_READ`

5. Enum ActionState

- Renamed `Offered` to `OFFERED`
- Renamed `Paid` to `PAID`

6. Enum InitializationStatus
    - Renamed `Success` to `SUCCESS`
    - Renamed `Error` to `ERROR`
    - Renamed `InvalidId` to `INVALID_ID`
    - Renamed `Unknown` to `UNKNOWN`
7. Enum LoginResult
    - Renamed `Success` to `SUCCESS`
    - Renamed `Error` to `ERROR`
8. Enum LogoutResult
    - Renamed `Success` to `SUCCESS`
    - Renamed `Error` to `ERROR`
9. Enum MessageType
    - Renamed `Text` to `TEXT`
    - Renamed `Image` to `IMAGE`
    - Renamed `File` to `FILE`
    - Renamed `Carousel` to `CAROUSEL`
    - Renamed `List` to `LIST`
    - Renamed `Location` to `LOCATION`
10. Enum MessageUploadStatus
    - Renamed `Unsent` to `UNSENT`
    - Renamed `Failed` to `FAILED`
    - Renamed `Sent` to `SENT`
    - Renamed `NotUserMessage` to `NOT_USER_MESSAGE`
11. Enum PaymentStatus
    - Renamed `Unknown` to `UNKNOWN`
    - Renamed `Success` to `SUCCESS`
    - Renamed `Error` to `ERROR`

12. Enum SmoochConnectionStatus
    - Renamed `Connected` to `CONNECTED`
    - Renamed `Disconnected` to `DISCONNECTED`
    - Renamed `NotYetInitiated` to `NOT_YET_INITIATED`
13. Class ConversationActivity
    - Replaced `show(Context)` with `ConversationActivityBuilder`
    - Replaced `show(Context, int)` with `ConversationActivityBuilder`
    - Replaced `show(Context, int, String)` with `ConversationActivityBuilder`
    - Replaced `show(Context, String)` with `ConversationActivityBuilder`

API Additions

1. Class Smooch
    - New method `getConversation(String, SmoochCallback)`
    - New method `getConversationsList(SmoochCallback)`
    - New method `setConversationDelegate(ConversationDelegate)`
    - New method `setConversationViewDelegate(ConversationViewDelegate)`
    - New method `setMessageModifierDelegate(MessageModifierDelegate)`
2. Class Settings
    - New method `getIntegrationId(): String`
3. Interface Conversation
    - New method `getDisplayName(): String`
    - New method `getLastRead(): Date`
    - New method `getLastUpdatedAt(): Date`
    - New method `getParticipants(): List<Participant>`
4. Class Participant
    - New class that represents participants of a Conversation
5. Interface ConversationDelegate
    - Previously `Conversation#Delegate`

- New method `onConversationsListUpdate(List<Conversation>)`

6. Interface ConversationViewDelegate
   - Previously `Conversation#ViewDelegate`
7. Interface MessageModifierDelegate
   - Previously `Conversation#MessageModifierDelegate`
8. Enum ConversationEventType
   - New value `CONVERSATION_ADDED`
   - New value `CONVERSATION_REMOVED`
   - New value `PARTICIPANT_ADDED`
   - New value `PARTICIPANT_REMOVED`
9. Class ConversationEvent
   - New method `getConversationId(): String`
   - New method `getAppUserId(): String`
   - New method `getUserId(): String`
10. Interface SmoochCallback
    - Added type parameter
      - The `data` returned by the callback no longer needs to be cast for better type safety
      - Existing callback usages will show a warning saying the callback is unchecked
      - The compiler should suggest the type that is missing
11. Class ConversationActivityBuilder
    - New builder for configuring `ConversationActivity`
    - New method `intent(Context): Intent` to retrieve the intent and start the activity

Other Changes

- Updated `compileSdkVersion` to `29`
- Updated `targetSdkVersion` to `29`

- Updated `com.google.gms:google-services` to `4.3.3`
- Updated `com.google.code.gson:gson` to `2.8.6`
- Updated `com.squareup.okhttp3:okhttp` to `3.12.6`
- Added `com.squareup.retrofit2:retrofit:2.6.2`
- Added `com.squareup.retrofit2:converter-gson:2.6.2`
- Added `com.google.dagger:dagger:2.25.2`
- Removed `com.google.firebase:firebase-core` dependency
- SDK dependencies are no longer declared as `api`
- Updated SDK proguard rules

SDK API

Please refer to the Javadoc for the full specification of each class included in the SDK.