

Connection to an existing session

Since Android SDK build [1.1.0.55](#), it is possible to connect to an existing websocket session on the server to accept an incoming call when push notification is received:

1. Set keepAlive option when connection is established for the first time

```
Connection connection = new Connection();
connection.setSipLogin(mSipLoginView.getText().toString());
connection.setSipPassword(mSipPasswordView.getText().toString());
...
connection.setKeepAlive(true);
session.connect(connection);
```

2. Keep a session token after successful connection

```
@Override
public void onConnected(final Connection connection) {
    runOnUiThread(new Runnable() {
        @Override
        public void run() {
            ...
            String token = connection.getAuthToken();
            if (token != null && !token.isEmpty()) {
                mAuthTokenView.setText(token);
                mConnectTokenButton.setEnabled(true);
            }
        }
    });
}
```

Then, the session can be disconnected on mobile device when application goes to background, but the session will be kept on server during 1 hour by default.

3. When push notification is received, connect to the existing session by token

```
createSession();
Connection connection = new Connection();
connection.setAuthToken(authToken);
connection.setKeepAlive(true);
session.connect(connection);
```

4. Receive incoming call event and create answer/hangup alert dialog

```
@Override
public void onCall(final Call call) {
    call.on(callStatusEvent);
}
```

```

/**
 * Display UI alert for the new incoming call
 */
runOnUiThread(new Runnable() {
    @Override
    public void run() {
        AlertDialog.Builder builder = new
AlertDialog.Builder(PhoneMinActivity.this);

        builder.setTitle("Incoming call");

        builder.setMessage("Incoming call from '" + call.getCaller() +
""");

        builder.setPositiveButton("Answer", new
DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialogInterface, int
i) {
                PhoneMinActivity.this.call = call;

                ActivityCompat.requestPermissions(PhoneMinActivity.this,
                    new String[]{Manifest.permission.RECORD_AUDIO},
                    INCOMING_CALL_REQUEST_CODE);
            }
        });
        builder.setNegativeButton("Hangup", new
DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialogInterface, int
i) {
                call.hangup();
                incomingCallAlert = null;
            }
        });
        incomingCallAlert = builder.show();
    }
});
}

```

5. Accept the incoming call

```

case INCOMING_CALL_REQUEST_CODE: {
    if (grantResults.length == 0 ||
        grantResults[0] != PackageManager.PERMISSION_GRANTED) {
        call.hangup();
        incomingCallAlert = null;
        Log.i(TAG, "Permission has been denied by user");
    } else {
        mCallButton.setText(R.string.action_hangup);
        mCallButton.setTag(R.string.action_hangup);
        mCallButton.setEnabled(true);
        mCallStatus.setText(call.getStatus());
        call.answer();
        incomingCallAlert = null;
        Log.i(TAG, "Permission has been granted by user");
    }
}
}

```
