ABTO VoIP SIP SDK for iOS

General description

ABTO VoIP SIP SDK consists as 'phone' object with parts (interfaces): configuration, actions and event callbacks.

Configuration – allows to setup phone options, like registration credentials, display name, proxy, signaling transport, turn on/off STUN, enable/disable ICE, activate/deactivate codecs set timeouts, ringtones sound, etc;

Actions – set of methods, which allows to start outgoing call, answer/reject incoming call, etc.
Pay attention: SIP related calls are asynchronous in nature, as require to send and receive requests to/from server, so even when SDK method retunes YES (true) it means that some task was moved to internal queue and will be handled in background thread.

Event callbacks – provides ability to notify users application about some events handled/raised by SIP stack (like incoming call, successful registration).

SDK instance life cycle

1. Create phone instance (typically in 'application didFinishLaunchingWithOptions')

```
_phone = [AbtoPhoneInterface new];
[_phone initialize:self];
```

2. Get config interface, load settings, hardcode own one, finalize changes

```
AbtoPhoneConfig* config = abtoAppDelegate.sharedInstance.phone.config;
[config loadFromUserDefaults:SETTINGS_KEY];
config.regUser = name.text;
config.regPassword = password.text;
config.regDomain = domain.text;
[abtoAppDelegate.sharedInstance.phone finalizeConfiguration]
```

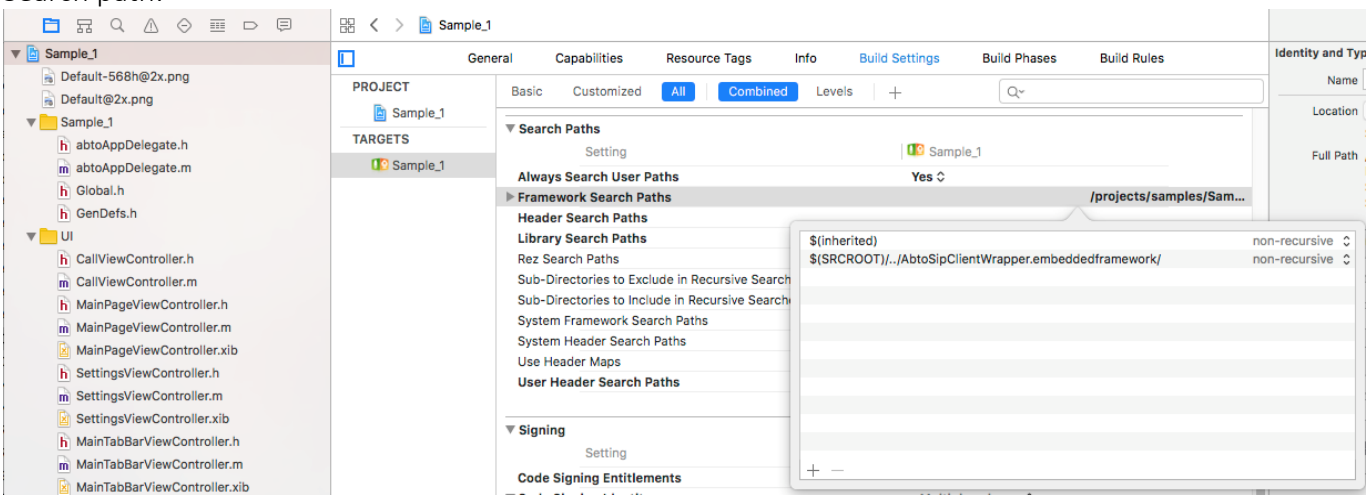
3. Make calls/handle events

4. Destroy phone instance

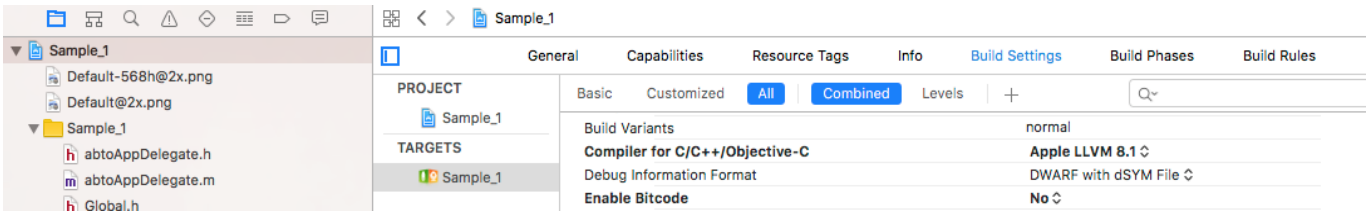
```
- (void)applicationWillTerminate:(UIApplication *)application {
    [_phone deinitialize];
}
```

Recommended project settings

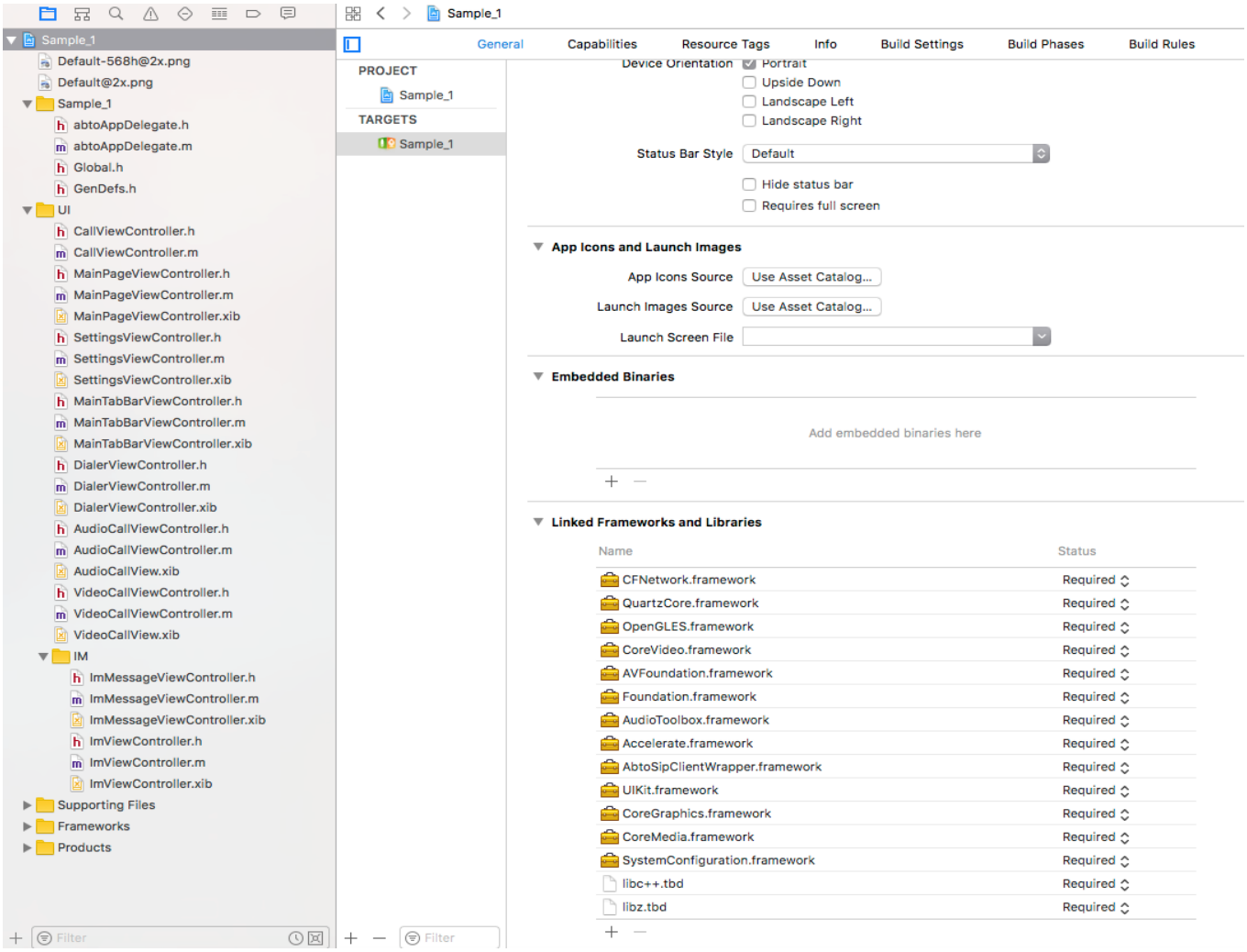
Search path:



Bit code:



Frameworks:



Phone Configuration interface

When is required to set initial or modify existing “phone’s” configuration use following code:

```
//retrieve configuration interface
AbtoPhoneConfig* config = phone.config;

//Modify properties
config.regUser = name.text;
config.regPassword = password.text;
config.regDomain = domain.text;
config.ringToneUrl = @"abto_ringtone.mp3";
…
//Store new settings
[config saveToUserDefaults:SETTINGS_KEY];

//Verify and apply all changes
[phone finalizeConfiguration]
```

AbtoPhoneConfig class

Properties of AbtoPhoneConfig

enableSrtp

Enables/disables SRTP (use SRTP protocol to encrypt audio/video streams).

Syntax

```
@property BOOL enableSrtp
```

Default

NO

Example

```
// set SRTP to disabled state
config.enableSrtp = NO;


// query if SRTP is enabled
if (config.enableSrtp)…
```

allowSrtp256

Enables/disables support for 256bit keys in SRTP.

Syntax

```
@property BOOL allowSrtp256
```

Default

NO

Example

```
// allow 256bit cypher
config.allowSrtp256 = YES;

// check is 256 bit SRTP keys allowed
if (config.allowSrtp256)
```

enableZrtp

Enables/disables ZRTP (use ZRTP protocol to encrypt audio/video streams).

Syntax

```
@property BOOL enableZrtp
```

Default

NO

Example

```
// set ZRTP to disabled state
config.enableZrtp = NO;


// check whether ZRTP is enabled
if (config.enableSrtp)
```

tlsCaList

Set path to file with CA certificate(s), which SDK will use to verify servers certificate on establishing TLS connection.

Syntax

```
@property NSString *tlsCaList
```

Default

```
@""
```

Example

```
// set path to TLS CA cert
config.tlsCaList = [NSString stringWithFormat:@"%@/%@",
NSBundle.mainBundle.resourcePath, @"ca.crt"]];
// query for TLS CA path presence
if (!config.tlsCaList.length)
```

enableProxy

Enables/disables using server address, set in '**proxy**' property.

Syntax

```
@property BOOL enableProxy
```

Default

```
NO
```

Example

```
// switch off proxy usage
config.enableProxy = NO;

// assign proxy var with proxy state usage
proxy = config.enableProxy;
```

proxy

Set/Unset SIP Proxy address (if set – SDK routes all request to proxy address instead of RegDomain).

Syntax

```
@property NSString *proxy
```

Default

```
@""
```

Example

```
// set SIP proxy
config.regDomain = @"superserver";
config.proxy = @"172.30.30.150";//registration domain exists, but can't
be resolved to IP address via DNS. Using 'proxy' allows set destination,
where to send SIP requests.
```

enableStun

Enable/disable using server address set in '**stun**' property.

Syntax

```
@property BOOL enableStun
```

Default

```
NO
```

Example

```
// use STUN
config.enableStun = YES;
config.stun = @"stun.l.google.com:19302”;
```

stun

Set/Unset address of STUN server (if set – SDK resolves external `address:port` for SIP/RTP/RTCP streams using this server).

Syntax

```
@property NSString *stun
```

Default

```
@""
```

Example

```
// set STUN value
config.stun = @"stun.sipgate.net";
// retrieve STUN value
currentStun = config.stun;
```

enableIce

Enable/Disable ICE support.

This option requires to set STUN server – '**stun**'.

Syntax

```
@property BOOL enableIce
```

Default

```
NO
```

Example

```
// disable ICE
config.enableIce = NO;
// query if ICE is enabled
if (config.enableIce)
```

enableRingTone

Enable/Disable playing ringtone sound, set in '**ringToneUrl**', when received incoming call.

Syntax

```
@property BOOL enableRingTone
```

Default

```
YES
```

Example

```
// enable ringtone playback handling by SDK
config.enableRingTone = YES;
config.ringToneUrl = @"ipod-library://item/item.mp3?id=685...29";


// query for enabled SDK ringtone handling
if (config.enableRingTone)
```

ringToneUrl

Set/Unset URL to ringtone sound.
Can be set as related path (SDK will use current bundle path plus property value).

Syntax

```
@property NSString *ringToneUrl
```

Default

```
@"abto_ringtone.mp3"
```

Example

```
// set iTunes media file as ringtone
config.ringToneUrl = @"ipod-library://item/item.mp3?id=6858129";

// check if it's default SDK ringtone
if (![@"abto_ringtone.mp3" isEqual:config.ringToneUrl])
```

enableRingBackTone

Enable/disable playing ring back tone.
When this option enabled and received "180 Ringing" from remote side – SDK plays local sound, using resource set in '**ringBackToneUrl**'.

Syntax

```
@property BOOL enableRingBackTone
```

Default

```
YES
```

Example

```
// enable ringback tone playback handling by SDK

config.enableRingBackTone = YES;

// query for enabled SDK ringback tone handling
```

```
if (config.enableRingBackTone)
```

ringBackToneUrl

Set/Unset URL to ringbacktone sound.
Can be set as related path (SDK will use current bundle path plus property value).

Syntax

```
@property NSString *ringBackToneUrl
```

Default

```
@"abto_ringbacktone.wav"
```

Example

```
// set iTunes media file as ringback tone
config.ringBackToneUrl = @"ipod-library://item/item.mp3?id=6129";
// check if it's default SDK ringback tone
if (![@"abto_ringbacktone.wav" isEqual:config.ringBackToneUrl])
```

enableAutorotateCaptureDevice

Enable/disable video rotation to always match portrait mode.

Syntax

```
@property BOOL enableAutorotateCaptureDevice
```

Default

```
NO
```

Example

```
// enable video rotation so that it always in portrait mode
config.enableAutorotateCaptureDevice = YES;
// query video rotation state
if (config.enableAutorotateCaptureDevice)
```

ua

Set string, which SDK puts in SIP header 'User-Agent'.

Syntax

```
@property NSString *ua
```

Default

```
@"ABTO VoIP SDK"
```

Example

```
// set UA text
config.ua = @"VOIP";
// retrieve UA name
currentUa = config.ua;
```

localIp

Override IP address which SDK puts in SIP header 'Contact'.

Syntax

```
@property NSString *localIp
```

Default

```
@""
```

Example

```
// set stack IP used for registration
config.localIp = @"192.168.0.2";
```

displayName

Set display name (caller id) which SDK will put in SIP header 'From'.

Syntax

```
@property NSString *displayName
```

Default

```
@""
```

Example

```
config.displayName = @"super_user";
config.regUser = @"100";
config.regDomain = @"172.30.30.150";
//will produce:
From: "super_user"<sip:100@172.30.30.150>
Many apps/devices will display 'super_user' as name of caller on incoming call instead of '100'.
```

regUser

Set SIP user name (extension), which SDK uses in SIP header 'From'.
SDK also uses this value as user name for authentication, when '**regAuthId**' is empty.

Syntax

```
@property NSString *regUser
```

Default

```
@""
```

Example

```
// SIP user
config.regUser = @"user";
```

regPassword

Set password for SIP authentication.

Syntax

```
@property NSString *regPassword
```

Default

```
@""
```

Example

```
// SIP password
config.regPassword = @"password";
```

regDomain

Set SIP domain name (server address) for registration.

Syntax

```
@property NSString *regDomain
```

Default

```
@""
```

Example

```
// set SIP host in domain form
config.regDomain = @"mysiphost.com";


// retrieve SIP host used in requests
domain = config.regDomain;
```

regAuthId

Set user name, which SDK has to use for authentication.
Typically, when client sends SIP REGISTER/INVITE requests sip servers are expecting digest access authentication. SDK uses value of this property as username.

Syntax

```
@property NSString *regAuthId
```

Default

```
@""
```

Example

```
// set Authentication ID for SIP
config.regAuthId = @"911";

// retrieve SIP auth name
auth = config.regAuthId;
```

regExpirationTime

Set interval in seconds, how often SDK has update registration on server.
Allowed range: [15 ... 3600].
Additionally SDK allows to set 0, which means disable registration (doesn't send SIP REGISTER request to server).

Syntax

```
@property int regExpirationTime
```

Default

```
300
```

Example

```
// send REGISTER every 60 seconds
config.regExpirationTime = 60;

// check if SIP REGISTER is disabled
if (config.regExpirationTime == 0)
```

localPort

Set local port number, which SDK will use for sending/receiving for SIP requests.
Allowed range: [1000 … 65535].
Additionally SDK allows to set 0, which means to use random port number.

Syntax

```
@property int localPort
```

Default

```
0
```

Example

```
config.localPort = 5060;
config.regDomain = @"172.30.30.150:7000";
//sends sip request from local port 5060 to remote server port 5070
```

registerTimeout

Set value of registration timeout (how long SDK has to wait response on SIP REGISTER request, before raise 'onRegistrationFailed' event).
Value in milliseconds. Allowed range [1000:65535];

Syntax

```
@property int registerTimeout
```

Default

```
0 corresponds to 32000ms.
```

Example

```
// set register timeout to 5 seconds
config.registerTimeout = 5000;
```

hangupTimeout

Set value for hangup timeout (how long SDK has to wait response on SIP BYE/CANCEL request before raise 'onCallDisconnected' event).
Value in milliseconds; Allowed range [1000:65535];

Syntax

```
@property int hangupTimeout
```

Default

0 corresponds to 32000ms

Example

```
// set hangup timeout to 3 seconds
config.hangupTimeout = 3000;
```

inviteTimeout

Set value for startCall timeout (how long SDK has to wait response on SIP INVITE requests, before raise 'onCallDisconnected' event).
Value in milliseconds; Allowed range [1000:65535];

Syntax

```
@property int registerTimeout
```

Default

```
0 corresponds to 32000ms
```

Example

```
// set startCall timeout to 5 seconds
config.inviteTimeout = 5000;
```

contactDetails

Allows insert additional text in SIP 'Contact' header

Syntax

```
@property NSString *contactDetails
```

Default

```
@""
```

Example

```
// append extra "expires=300" to SIP 'Contact' header value
config.contactDetails = @"expires=300";

//produces:
//Contact: <sip:username@123.456.7.8>;expires=3600;...
```

contactDetailsUri

Allows insert additional text in SIP 'Contact' header

Syntax

```
@property NSString *contactDetailsUri
```

Default

```
@""
```

Example

```
//append extra “pn-type=acme;” to SIP Contact URL value
config.contactDetailsUri = @"pn-type=acme";

//produces:
//Contact: <sip:username@123.456.7.8;pn-type=acme>...
```

Methods of AbtoPhoneConfig class

-initWithConfig:

Copy properties from another instance.

Syntax

```
- (id)initWithConfig:(AbtoPhoneConfig *)config
```

Parameters

Return Value

Returns initialized AbtoPhoneConfig instance pointer or NULL otherwise

Example

```
//Instantiate AbtoPhoneConfig and initialize it with values from oldConfig
AbtoPhoneConfig *config = [[AbtoPhoneConfig alloc]
initWithConfig:oldConfig];
```

-setFromConfig:

Copy properties from another instance.

Syntax

```
- (void)setFromConfig:(AbtoPhoneConfig *)config
```

Parameters

config(AbtoPhoneConfig *) - config to copy values from

Return Value

No return value

Example

```
// set AbtoPhoneConfig to values from oldConfig
[config setFromConfig:oldConfig];
```

-saveToUserDefaults:

Save values to UserDefaults with provided key

Syntax

```
- (BOOL)saveToUserDefaults:(NSString *)key
```

Parameters

key(NSString *) - key of NSUserDefaults where to serialize values

Return Value

boolean YES for success of save operation, NO otherwise;

Example

```
//save config to "settings" key and check operation success
if([config saveToUserDefaults:@"settings"])
```

-loadFromUserDefaults:

Load values from UserDefaults with provided key

Syntax

```
- (BOOL)loadFromUserDefaults:(NSString *)key
```

Parameters

key(NSString *) - key of NSUserDefaults from which de-serialize values **Return Value**

boolean YES for success of load operation, NO otherwise

Example

```
// load config from "settings" key and check operation success
if ([config loadFromUserDefaults:@"settings"])
```

+loadFromUserDefaults:

Creates new config instance and loads values from UserDefaults with provided key.

Syntax

```
+ (id)loadFromUserDefaults:(NSString *)key
```

Parameters

key(NSString *) - key of NSUserDefaults from which de-serialize values

Return Value

Returns new AbtoPhoneConfig instance pointer or NULL otherwise;

Example

```
// instantiate and load config from "settings" key
AbtoPhoneConfig *config = [AbtoPhoneConfig
loadFromUserDefaults:@"settings"];
```

-setCodecPriority:priority:

Allows configure codecs order or disable selected codecs.

Codec with highest priority is displayed on top of SDP codecs list.

Syntax

```
- (void)setCodecPriority:(PhoneAudioVideoCodec)idx
priority:(NSInteger)priority
```

Parameters

idx(PhoneAudioVideoCodec) - codec index, refer to PhoneAudioVideoCodec

priority(NSInteger) - codec priority value.

Should be in range [0:255]. To disable codec – use 0 value;

Return Value

No return value

Example

```
//Set OPUS audio codec priority to 200
[config setCodecPriority:PhoneAudioVideoCodecOpus priority:200];

//Disable PCMU audio codec
[config setCodecPriority:PhoneAudioVideoCodecPcmu priority:0];
```

-getCodecPriority:

Get particular codec priority

Syntax

```
- (NSInteger)codecPriority:(PhoneAudioVideoCodec)idx
```

Parameters

idx(PhoneAudioVideoCodec) – codec index, refer to PhoneAudioVideoCodec

Return Value

Current priority value for specified codec

Example

```
//Get current SPEEX audio codec priority value
NSInteger value = [config getCodecPriority:PhoneAudioVideoCodecSpeex];
```

+codecName:

Get codec name by its index.

Syntax

```
+ (NSString *)codecName:(PhoneAudioVideoCodec)idx
```

Parameters

idx(PhoneAudioVideoCodec) – codec index, refer to PhoneAudioVideoCodec

Return Value

String with codec name

Example

```
// logs name that match GSM codec
NSLog(@"GMS=%@",[AbtoPhoneConfig codecName:PhoneAudioVideoCodecGsm]
```

+codecType:

Get codec type by its index.

Syntax

```
+ (PhoneCodecType)codecType:(PhoneAudioVideoCodec)idx
```

Parameters

idx(PhoneAudioVideoCodec) – codec index, refer to PhoneAudioVideoCodec

Return Value

Enumerator that indicates type of codec – audio, video or unsupported.

Example

```
//Iterate over codecs and check their type
```

```
for (NSInteger codec = PhoneAudioVideoCodecNone + 1; codec <
PhoneAudioVideoCodecCount; codec++)
{
    switch ([AbtoPhoneConfig codecType:codec]) {
        case PhoneCodecTypeAudio: break;   // is audio codec
        case PhoneCodecTypeVideo: break;   // is video codec
        default:                           // unsupported codec
    }
}
```

AbtoPhoneInterface class

Properties of AbtoPhoneInterface

libVersion

Returns version of this SDK build.

Syntax

```
@property(readonly) NSString *libVersion
```

Example

```
NSLog(@"ABTO version ", phone.libVersion);
```

Methods of AbtoPhoneInterface class

-initialize:

Initializes phone instance. Should be called before any other method.

Syntax

```
- (BOOL)initialize:(id <AbtoPhoneInterfaceObserver>)observer
```

Parameters

```
observer( AbtoPhoneInterfaceObserver *) - observer delegate to listen for
events
```

Return Value

```
boolean value YES - indicates success initialization, NO otherwise;
```

Example

```
// phone initialization with enabled background
if ([phone initialize:self])
```

-initialize:withBackground:

Same as '-initialize', but additionally allows set background mode behavior.

SDK has code which allows to work in background and prevent app from sleep, when user presses Home button (pay attention: background mode drains extra battery).

Apple recommends to use PUSH notifications for handling incoming SIP messages. In this case app sleeps in background and system wakes it up (or starts), when received push notification and background mode implementation has to be disabled using extra argument.

Syntax

```
- (BOOL)initialize:(id <AbtoPhoneInterfaceObserver>)observer
withBackground:(BOOL)state
```

Parameters

`observer(AbtoPhoneInterfaceObserver *)` observer delegate to listen for events;

`withBackground(BOOL)` flag that enables/disabled background mode implementation;

Return Value

boolean value YES - indicates success initialization, NO otherwise;

Example

```
// phone initialization with disabled background mode
if ([phone initialize:self withBackground:NO])
```

-deinitialize

Deinitializes phone instance.

Syntax

```
- (void)deinitialize
```

Parameters
Return Value

No return value

Example

```
// deinitialize AbtoPhoneInterface
[phone deinitialize];
```

-finalizeConfiguration

Method verifies and applies changes, made via config interface.

Pay attention, that method is asynchronous and when it returns sip stack may be not ready yet.

Syntax

```
- (BOOL)finalizeConfiguration
```

Parameters
Return Value

boolean YES indicates success of configuration operation, NO otherwise;

Example

```
// apply phone settings and start stack
if ([phone finalizeConfiguration])
```

-config

Retrieve instance of AbtoPhoneConfig class.

Syntax

```
- (AbtoPhoneConfig *)config;
```

Parameters
Return Value

```
Instance of AbtoPhoneInterface;
```

Example

```
// retrieve phone stack config
AbtoPhoneConfig *config = [phone config];
```

-keepAwake:

Method obsolete and does nothing.

-unregister

Start unregister operation.

Pay attention- method is asynchronous and only initiates operation. When it returns phone is not unregistered yet. After dialog with server and successful un-registration SDK raises callback 'onUnRegistered'.

Syntax

```
- (BOOL)unregister
```

Parameters
Return Value

```
boolean YES indicates that operation started successfully, NO otherwise
```

Example

```
// do something on successful start of phone unreg
if ([phone unregister])
```

-startCall:withVideo:

Make outgoing audio/video call.

Pay attention- method is asynchronous and only verifies received arguments and initiates outgoing call operation.

Syntax

```
- (NSInteger)startCall:(NSString *)destination withVideo:(BOOL)video
```

Parameters

```
destination(NSString *) – number of remote side or SIP URI(sip:user@domain);
video(BOOL) - flag that indicate type of call – audio(NO), video(YES);
```

Return Value

```
Integer value that is call ID generated by SDK and may be used for later
call manipulations or events handling;
```

Example

```
//Try to start audio call and test if it failed
NSInteger callId = [phone startCall:@"911" withVideo:NO];
if ( callId == kInvalidCallId)
```

-answerCall:status:withVideo:

Answer incoming call.

Syntax

```
- (BOOL)answerCall:(NSInteger)callId status:(int)status
withVideo:(BOOL)video
```

Parameters

callId (NSInteger) – value received in 'onIncomingCall' delegate;

video(BOOL) - type of call: audio(NO), video(YES);

Return Value

boolean YES indicates success, NO otherwise;

Example

```
if([phone answerCalll:callId status:200 withVideo:NO])…
```

-hangUpCall:status:

Method rejects incoming call or cancels/hangups outgoing.

Syntax

```
- (BOOL)hangUpCall:(NSInteger)callId status:(int)status
```

Parameters

callId(NSInteger) - value received in 'onIncomingCall' delegate or returned by 'startCall' method;

status(int) - SIP status code to send;

Return Value

boolean value that indicates that stack accepts action

Example

```
// end call with 487 status – Request Terminated
if ([phone hangUpCall:callId status:487])
```

-holdRetrieveCall:

Hold/retrieve call. Works as toggle method. Assumes by default that call is in unhold state.

Syntax

```
- (BOOL)holdRetriveCall:(NSInteger)callId
```

Parameters

callId(NSInteger) - value received in 'onIncomingCall' delegate or returned by 'startCall' method;

Return Value

boolean value that indicates that stack accepts action;

Example

```
[phone holdRetrieveCalll:callId]
```

-setCall:speakerLevel:

Adjusts speaker level for call.

Syntax

```
- (BOOL)setCall:(NSInteger)callId speakerLevel:(float)level
```

Parameters

callId(NSInteger) - value received in 'onIncomingCall' delegate or returned by 'startCall' method;

level(float) - value that specifies sound level. Should be in range [0; 32].

Return Value

boolean value that indicates that stack accepts action;

Example

```
// resets call speaker level
if ([phone setCall:callId speakerLevel:1.0])
```

-setCall:microphoneLevel:

Adjusts microphone level for call.

Syntax

```
- (BOOL)setCall:(NSInteger)callId microphoneLevel:(float)level
```

Parameters

callId(NSInteger) - value received in 'onIncomingCall' delegate or returned by 'startCall' method;

level(float) - value that specifies sound level. Should be in range [0; 32].

Return Value

boolean value that indicates that stack accepts action;

Example

```
// resets call microphone level
if ([phone setCall:callId microphoneLevel:1.0])
```

-muteMicrophone:on:

Mute/unmute microphone.

Syntax

```
- (BOOL)muteMicrophone:(NSInteger)callId on:(BOOL)on
```

Parameters

callId(NSInteger) - value received in 'onIncomingCall' delegate or returned by 'startCall' method;

on(BOOL) - unmute(NO), mute(YES)

Return Value

boolean value that indicates that stack accepts action;

Example

```
// mute call
if ([phone muteMicrophone:callId on:YES])
```

-sendTone:tone:

Send DTMF to remote side.

Note: SDK sends tones as RTP signaling packets, which requires codec "telephone-event/8000". On SDK side this code is always enabled. In case when during call negotiation codec was disabled – SDK plays tones sound to remote side.

Syntax

```
- (BOOL)sendTone:(NSInteger)callId tone:(unichar)tone
```

Parameters

callId(NSInteger) - value received in 'onIncomingCall' delegate or returned by 'startCall' method;

tone(unichar) - ASCII value of DTMF.
One of: '0', '1', '2', '3', '4', '5', '6', '7', '8', '9', '*', '#', 'A', 'B', 'C', 'D'.

Return Value

boolean value that indicates that stack accepts action;

Example

```
// send '*' DTMF
if ([phone sendTone:callId tone:'*'])
```

-sendToneViaInfo:tone:

Send DTMF to remote side in SIP INFO message.

Syntax

```
- (BOOL)sendToneViaInfo:(NSInteger)callId tone:(unichar)tone
```

Parameters

callId(NSInteger) - value received in 'onIncomingCall' delegate or returned by 'startCall' method;

tone(unichar) - ASCII value of DTMF.
Should one of: '0', '1', '2', '3', '4', '5', '6', '7', '8', '9', '*', '#', 'A', 'B', 'C', 'D'.

Return Value

boolean value that indicates that stack accepts action;

Example

```
// sends 9 DTMF to call via INFO SIP message
if ([phone sendToneViaInfo:callId tone:'9'])
```

-setBluetoothOn:

Redirect sound to Bluetooth headset.

Syntax

```
- (BOOL)setBluetoothOn:(BOOL)on
```

Parameters

on(BOOL) - set redirect state other(NO), bluetooth(YES)

Return Value

boolean value that indicates that stack accepts action;

Example

```
// redirect sound to bluetooth speaker
if ([phone setBluetoothOn:YES])
```

-setSpeakerphoneOn:

Redirect sound to speaker.

Syntax

```
- (BOOL)setSpeakerphoneOn:(BOOL)on
```

Parameters

on(BOOL) - set redirect state other(NO), speaker(YES).

Return Value

boolean value that indicates that stack accepts action;

Example

```
// redirect sound to loud speaker
if ([phone setSpeakerphoneOn:YES])
```

-sendTextMessage:withBody:

Send SIP MESSAGE request to remote side. Method is asynchronous and simply starts operation. To detect was message sent – handle 'onTextMessageStatus' event.

Syntax

```
- (BOOL)sendTextMessage:(NSString *)to withBody:(NSString *)message
```

Parameters

to(NSString *) - destination number or SIP URI(sip:user@domain).

message(NSString *) - text message to send.

Return Value

boolean value that indicates that stack accepts action;

Example

```
// try to send text "Hello!" to number 111
if ([phone sendTextMessage:@"111" withBody:@"Hello!"])
```

-transferCall:toContact:

Start unattended (blind) transfer. To get transfer result handle 'onTransferStatus' event.

Syntax

```
- (BOOL)transferCall:(NSInteger)callId toContact:(NSString *)uri
```

Parameters

callId(NSInteger) - value received in 'onIncomingCall' delegate or returned by 'startCall' method;

uri(NSString *) - destination number or SIP URI(sip:user@domain).

Return Value

boolean value that indicates that stack accepts action

Example

```
// try to transfer current call to user 222
if ([phone transferCall:callId toContact:@"222"])
```

-setPresence:statusText:

Set own presence status with possibility to include extra text description.

Syntax

```
- (BOOL)setPresence:(PhoneBuddyStatus)status statusText:(NSString *)text
```

Parameters

status(PhoneBuddyStatus) - current user presence status defined by enumerator

text(NSString *) extra status description

Return Value

boolean value indicates that stack accepts action

Example

```
// set AWAY status with extra description "BRB in 15 min"
if ([phone setPresence:PhoneBuddyStatusAway statusText:@"BRB in 15 min"])
```

-subscribeBuddy:on:

Send SIP SUBSCRIBE request to server with selected Contact.

Syntax

```
- (BOOL)subscribeBuddy:(NSString *)uri on:(BOOL)on
```

Parameters

uri(NSString *) – remote contact user name or uri

on(BOOL) - boolean 'NO' to subscribe, 'YES' - unsubscribe

Return Value

boolean value that indicates that stack accepts action

Example

```
//subscribe presence events from number 5050
if ([phone subscribeBuddy:@"5050" on:YES])
```

-setRemoteView:

Set UIImageView where SDK will display remote video during video call. Must be set before call started.

Syntax

```
- (void)setRemoteView:(UIImageView *)view
```

Parameters

view(UIImageView *) – view used for remote video display

Return Value

No return value

Example

```
//set remoteVideo view as destination for remote video
[phone setRemoteView:self.remoteVideo];
```

-setLocalView:

Set UIImageView where SDK will display local video during video call. Must be set before call started.

Syntax

```
- (void)setLocalView:(UIImageView *)view
```

Parameters

view(UIImageView *) – view used for local video display

Return Value

No return value

Example

```
// set localVideo view as destination for local video
[phone setLocalView:self.localVideo];
```

-isVideoCall:

Verify is incoming call audio only or video (verifies in received SIP request SDP contains 'video' media).

Syntax

```
- (BOOL)isVideoCall:(NSInteger)callId
```

Parameters

callId(NSInteger) – call ID obtained from incoming call delegate or via startCall action.

Return Value

boolean value 'NO' indicates audio call, 'YES' – video.

Example

```
// check whether it is video call by querying call ID
if ([phone isVideoCall:callId])
```

-muteVideo:on:

Mute/Unmute local video during call.

Syntax

```
- (BOOL)muteVideo:(NSInteger)callId on:(BOOL)on
```

Parameters

callId(NSInteger) - call ID obtained from incoming call delegate or via startCall action

on(BOOL) - boolean 'NO' to unmute, 'YES' - mute

Return Value

boolean value that indicates that stack accepts action

Example

```
// mute video for call defined by ID
if ([phone muteVideo:callId on:YES])
```

-switchCameraToFront:on:

Switch local video source to front or back camera.

Syntax

```
- (BOOL)switchCameraToFront:(NSInteger)callId on:(BOOL)on
```

Parameters

callId(NSInteger) - call ID obtained from incoming call delegate method or via startCall action

on(BOOL) - boolean 'NO' – use back camera, 'YES' – front camera.

Return Value

boolean value that indicates that stack accepts action

Example

```
// switch to back camera as source of local video for call defined by ID
if ([phone switchCameraToFront:callId on:NO])
```

-startRecordingFor:filePath:

Start recording call audio (local and remote) to file. Currently only wav file format is supported.

Syntax

```
- (BOOL)startRecordingFor:(NSInteger)callId filePath:(NSString *)name
```

Parameters

callId(NSInteger) – call ID obtained from incoming call delegate method or via startCall action

name(NSString *) – file path to store recorded call audio. Should be full path with wav extension.

Return Value

boolean value that indicates that stack accepts action

Example

```
//start call audio recording to file in app main bundle with name using
template rec_<number>.wav
NSString *filename = [NSString stringWithFormat:@"%@/rec_%@.wav",
[[[NSFileManager defaultManager] URLsForDirectory:NSDocumentDirectory
```

```
inDomains:NSUserDomainMask] lastObject] path], number];
if ([phone startRecordingFor:callId filePath:filename])
```

-stopRecording

Stop audio recording.

Syntax

```
- (BOOL)stopRecording
```

Parameters

Return Value

boolean value that indicates that stack accepts action

Example

```
// try to stop call audio recording
if ([phone stopRecording])
```

-readCallMediaQuality:isVideo:

Get audio/video call quality statistic.

Syntax

```
- (AbtoPhoneMediaQuality *)readCallMediaQuality:(NSInteger)callId
isVideo:(BOOL)video
```

Parameters

callId(NSInteger) – call ID obtained from incoming call delegate or via startCall action

video(BOOL) – boolean 'NO' to get audio stream statistics, 'YES' – video.

Return Value

AbtoPhoneMediaQuality structure.

Example

```
// print current call quality statistics for audio stream
AbtoPhoneMediaQuality *info = [phone readCallMediaQuality: callId
isVideo:NO];
NSLog([NSString
stringWithFormat:@"RTT %ld : %ld : %ld\n%ld : %ld : %ld : %ld",
(long)info.minRtt, (long)info.maxRtt, (long)info.avgRtt,
(long)info.minBufferJitter, (long)info.maxBufferJitter,
(long)info.avgBufferJitter, (long)info.devBufferJitter]);
```

-isZrtpSecured:

Check is ZRTP enabled for call.

Syntax

```
- (BOOL)isZrtpSecured:(NSInteger)callId
```

Parameters

callId(NSInteger) – call ID obtained from incoming call delegate method or via startCall action.

Return Value

boolean value 'NO' indicates deactivated ZRTP, 'YES' – enabled ZRTP.

Example

```
// check if ZRTP is active for call
if ([phone isZrtpSecured:callId])
```

-setSasCall:validity:

Validate or invalidate current call SAS (*Short Authentication String*).

The communicating parties verbally cross-check a shared value displayed at both endpoints. If the values do not match, a man-in-the-middle attack is indicated.

App has invoke this method when user presses some button, which indicates correct/incorrect SAS.

Syntax

```
- (void)setSasCall:(NSInteger)callId validity:(BOOL)valid
```

Parameters

callId(NSInteger) – call ID obtained from incoming call delegate method or via startCall action.

valid(BOOL) – boolean 'YES' – SAS is valid(), 'NO' – invalid.

Return Value

No return value

Example

```
// invalidate ZRTP call SAS
[phone setSasCall:callId validity:NO];
```

-deactivateAudio

Deactivate Audio Session. Might be used for flawless CallKit integration.

Usage sequence is following:

1. Received SIP INVITE request (sip incoming call) SDK raises 'onIncomingCall' notification:
    a. App invokes this method to prevent CallKit from intercept audio session
    b. App invokes 'reportNewIncomingCall' to displaying native incoming call GUI;
2. User answers call from native GUI.
    a. App invokes methods to configure audio session
    b. App invokes 'activateAudio'
    c. App invokes 'answerCall' to answer this sip call

To see more review SDK example app 'SampleSwiftCallKit'.

Syntax

```
- (void)deactivateAudio
```

Parameters
Return Value
Example

```
// deactivate audio sub-system
[phone deactivateAudio];
```

-activateAudio

Activate Audio Session. Might be used for flawless CallKit integration

Syntax

```
- (BOOL)activateAudio
```

Parameters
Return Value

boolean value that indicates ZRTP active state – deactivated(NO) or active(YES)

Example

```
// activate audio subsystem and check result
if ([phone activateAudio])
```

+sipUriUsername:

Method parses and extracts username from SIP URI.

Syntax

```
+ (NSString *)sipUriUsername:(NSString *)uri
```

Parameters

```
uri(NSString *) –  URI source for extraction
```

Return Value

```
Extracted username string.
```

Example

```
//extracts username from SIP URI
NSString *str = @"sip:100@host";
NSString *toUser = [AbtoPhoneInterface sipUriUsername:str];//returns “100”
```

+sipUriDomain:

Method parses and extracts domain from SIP URI.

Syntax

```
+ (NSString *)sipUriDomain:(NSString *)uri
```

Parameters

```
uri(NSString *) – URI source for extraction
```

Return Value

```
Extracted domain string.
```

Example

```
//extracts username from SIP URI
NSString *str = @"sip:100@host";
NSString *toDomain = [AbtoPhoneInterface sipUriDomain:str];//returns “host”
```

+ sipUriDisplayName :

Method parses and extracts display name (caller id) from SIP URI.

Syntax

```
+ (NSString *)sipUriDisplayName:(NSString *)uri
```

Parameters

uri(NSString *) - URI source for extraction

Return Value

Extracted display name string.

Example

```
// extracts display name from SIP URI
NSString *str = @"caller <sip:100@host>";
NSString *toUser = [AbtoPhoneInterface sipUriDisplayName:str];//return
caller
```

Callbacks (events) AbtoPhoneInterface

SDK callbacks (events) exposed via AbtoPhoneInterfaceObserver delegate.
They can be divided into 6 logical parts:

1. Registration – OnRegistered, OnRegistrationFailed, OnUnRegistered, OnRemoteAlerting
2. Call – OnIncomingCall, OnCallConnected, OnCallDisconnected, OnCallAlerting, OnCallHeld, OnToneReceived, OnTransferStatus
3. IM – OnTextMessageReceived, OnTextMessageStatus
4. Presence – OnPresenceChanged
5. ZRTP – OnZrtpSas, OnZrtpSecureState, OnZrtpError
6. Network – OnNetworkStateChanged

ZRTP and Network events are optional for implementation.

-OnRegistered:

Event triggered when SDK successfully registered.

Syntax

```
- (void)onRegistered:(NSInteger)accId
```

Parameters

`accId(NSInteger) - internal account id assigned by stack`

Return Value

`No return value`

-OnRegistrationFailed:statusCode:statusText

Event triggered when received error in answer from server, or server doesn't answer during 'registerTimeout' period of time.

Syntax

```
- (void)onRegistrationFailed:(NSInteger)accId statusCode:(int)statusCode
statusText:(NSString *)statusText
```

Parameters

`accId(NSInteger) - account id assigned by SDK`

`statusCode(int) - status code`

`statusText(NSString *)  status text`

Return Value

`No return value`

-OnUnRegistered:

Event triggered on successful unregistration.

Syntax

```
- (void)onUnRegistered:(NSInteger)accId
```

Parameters

`accId(NSInteger) - account id assigned by SDK`

Return Value

No return value

-OnRemoteAlerting:status:

Event generated when received “100 Trying” from server.

Syntax

```
- (void)onRemoteAlerting:(NSInteger)accId statusCode:(int)statusCode
```

Parameters

accId(NSInteger) - account id assigned by SDK statusCode(int)

statusCode(int) - status code

Return Value

No return value

-OnIncomingCall:remoteContact:

Event generated when received incoming call request from remote side.

When there is already established call SDK raises this event with callId = kInvalidCallId.

Syntax

```
- (void)onIncomingCall:(NSInteger)callId remoteContact:(NSString
*)remoteContact
```

Parameters

callId(NSInteger) - internal call id assigned by SDK

remoteContact(NSString *) - remote contact SIP URI

Return Value

No return value

-OnCallConnected:remoteContact:

Event generated when successfully established incoming/outgoing call.

Syntax

```
- (void)onCallConnected:(NSInteger)callId remoteContact:(NSString
*)remoteContact
```

Parameters

callId(NSInteger) - internal call id assigned by SDK

remoteContact(NSString *) - remote contact SIP URI

Return Value

No return value

-OnCallAlerting:statusCode:

Even generated when received SIP message like 100, 180 or 183, etc.

Syntax

```
- (void)onCallAlerting:(NSInteger)callId statusCode:(int)statusCode
```

Parameters

callId(NSInteger) - internal call id assigned by SDK

statusCode(NSInteger) - status code

Return Value

No return value

-OnCallDisconnected:remoteContact:statusCode:message:

Event generated when disconnected call to remoteContact.

Syntax

```
- (void)onCallDisconnected:(NSInteger)callId remoteContact:(NSString *)
remoteContact statusCode:(NSInteger)statusCode message:(NSString *)message
```

Parameters

callId(NSInteger) - internal call id assigned by SDK

remoteContact(NSString *) - remote contact SIP URI

statusCode(NSInteger) - status code

message(NSString *) - status test

Return Value

No return value

-OnCallHeld:state:

Event generated when call was held/resumed on remote side

Syntax

```
- (void)onCallHeld:(NSInteger)callId state:(BOOL)state
```

Parameters

callId(NSInteger) - internal call id assigned by SDK

state(BOOL) - YES – held, NO - resumed

Return Value

No return value

-OnToneReceived:tone:

Event generated when received DTMF tone.

Syntax

```
- (void)onToneReceived:(NSInteger)callId tone:(NSInteger)tone
```

Parameters

callId(NSInteger) - internal call id assigned by SDK

tone(NSInteger) - ASCII char code representing tone

Return Value

No return value

-OnNetworkStateChanged:isIpv6:

Event generated when network condition changed. It notifies about connection availability and it type IPv4/IPv6.

Syntax

```
- (void)onNetworkstateChanged:(PhoneNetworkEvent)event isIpv6:(BOOL)ipv6
```

Parameters

event(PhoneNetworkEvent) – internal account id assigned by stack

ipv6(BOOL) - either IPv4(NO) or IPv6(YES)

Return Value

No return value

-OnTextMessageReceived:to:body:

Event generated when received SIP MESSAGE request.

Syntax

```
- (void)onTextMessageReceived:(NSString *)from to:(NSString *)to
body:(NSString*) body
```

Parameters

from(NSString *) - SIP(server side) or Stack(local side) status code

to(NSString *) - SIP(server side) or Stack(local side) status code

body(NSString *) - SIP(server side) or Stack(local side) status code

Return Value

No return value

-OnTextMessageStatus:reason:status:

Event generated when received confirmation on sent text message or expired timeout.

Syntax

```
- (void)onTextMessageStatus:(NSString *)address reason:(NSString *)reason
status:(BOOL)status
```

Parameters

address(NSString *) – message delivery address

reason(NSString *) – delivery status text

state(BOOL) – 'YES' – when sent successfully, 'NO' – wasn't sent

Return Value

No return value

-OnTransferStatus:statusCode:message:

Event generated when received SIP NOTIFY message with transfer status.

Syntax

```
- (void)onTransferStatus:(NSInteger)callId statusCode:(int)statusCode
message:(NSString *)message
```

Parameters

callId(NSInteger) – internal account id assigned by stack

statusCode(int) – transfer status code

message(NSString *) – transfer status message

Return Value

No return value

-OnPresenceChanged:status:note:

Event generated triggered on changes in Presence of subscribed contact

Syntax

```
- (void)onPresenceChanged:(NSString *)uri status:(PhoneBuddyStatus)status
note:(NSString *)note
```

Parameters

uri(NSString *) – contact URI, from which was received status

status(PhoneBuddyStatus) – status, refer enum PhoneBuddyStatus

note(NSString *) – extra description of status

Return Value

No return value

-OnZrtpSas:sas:

Event generated after creating ZRTP SAS hash.

Syntax

```
- (void)onZrtpSas:(NSInteger)callId sas:(NSString *)sas
```

Parameters

callId(NSInteger) – internal call id assigned by stack

sas(NSString *) – sas for this call

Return Value

No return value

Example

// instantiate and load config from “settings” key

-OnZrtpSecureState:secured:

Event generated when call security changed.

Syntax

```
- (void)onZrtpSecureState:(NSInteger)callId secured:(BOOL)secured
```

Parameters

callId(NSInteger) – internal call id assigned by stack

secured(BOOL) – whether call is secured by ZRTP

Return Value

No return value

-OnZrtpError:error:subcode:

Event generated in case of ZRTP errors.

Syntax

```
- (void)onZrtpError:(NSInteger)callId error:(NSInteger)error
subcode:(NSInteger)subcode
```

Parameters

callId(NSInteger) – internal call id assigned by stack

error(NSInteger) –base ZRTP error code, see ZRTP error codes

subcode(NSInteger) – subcode of ZRTP error, see ZRTP error subcodes

Return Value

No return value