

# ZOIPER WEB API

<b>REVISION</b>	<i>Zoiper Web</i>
<b>UPDATED</b>	<i>June 2010</i>

### 1. Introduction

Zoiper Web is designed to be embedded in web pages and is compatible with the most common web browsers on the market. It allows the website visitor to make and receive VoIP calls using the SIP or IAX2 protocols. It also allows the website to control the phone by using the Zoiper Web API, using JavaScript or similar technologies. Currently Zoiper Web supports a wide variety of browsers on Microsoft Windows operating system including, but not limited to: Microsoft Internet Explorer 6 and above, Mozilla FireFox 3.0 and above, Safari for Windows and Google Chrome.

Both the Microsoft ActiveX and NPAPI (Netscape Plugin Application Programming Interface) technologies are supported, offering native plugins for every browser technology.

NB: Some of the features in this document are marked as “**not available in all versions**”. This means that they are not available with the standard Zoiper WEB version which is for sale on the Zoiper online shop. For more information regarding the availability of these features and how you can purchase them, please contact [sales@zoiper.com](mailto:sales@zoiper.com)

### 2. Installation

Zoiper Web is using an installer program to install itself on the user's PC. This program takes care for the Internet Explorers CLSID registration and for setup of the NPAPI registry keys. This way a single installation can be used to enable all the browsers on the user's PC to use Zoiper Web, without having to install it for every browser separately.

For internet explorer, a cabinet (.cab) file which encapsulates the installer is provided. This enables Internet Explorer to automatically download and run the installer for the user. (This will also install Zoiper Web for all other browsers on the user's pc). The class ID (CLSID) used to identify Zoiper Web in Internet Explorer and Microsoft Windows is {BCCA9B64-41B3-4A20-8D8B-E69FE61F1F8B}.

For Firefox users, a native .xpi installer and a standalone executable installer are available, however please note that the use of the .xpi installer is not recommended as it needs to be installed before loading the page that hosts the Zoiper Web component in order to avoid a warning popup about an unsupported plugin asking to look for it on the Firefox plugin repository. The MIME type used to identify Zoiper Web in NPAPI based browsers is “application/x-zoiper-plugin”.

### 3. Zoiper Web API interface

The API provided with Zoiper Web consists of several classes which provide the functionality of creating and managing configuration, accounts, contacts and calls. It also provides event feedback from Zoiper Web to the browser in the form of predefined callback functions in the web page that is using it.

All the properties, method and callback parameters are using the “string” type. Note that because there are boolean values that can be passed and all the values are actually strings, there are two strings values representing “true” and “false” that are used in that case.

The structure of the API is based on one main class named "Phone" which provides you with the facilities to obtain and manage “Config”, "Account", "Call" and "Contact" objects, and a predefined number of

named functions, called "Callbacks" or "Events", that must be defined in the Javascript script so that Zoiper can call them back. There is a certain "Callback" which notifies the user that the phone has finished starting up and provides a reference to a "Phone" class instance. This "Callback" is used as an entry point for the Zoiper Web usage. The "Callbacks" are not really part of the classes but they will be described along with them as they are logically related.

### 3.1 "Phone" class

The "Phone" class represents the Zoiper Web as a whole. It provides facilities for general configuration ("Config" object), creation and management of "Account", "Call" and "Contact" objects. This is the main class of Zoiper Web and it is the only object that the user can get reference to directly from the JavaScript itself (from the <object> or <embed> tags). Using the object obtained that way is not advisable because it cannot be used before the "OnZoiperReady". "Callback" is called to indicate that the "Phone" is ready to use. This "Callback" also returns as parameter the reference to the "Phone" object to be used. Obtaining the "Phone" reference has another advantage, it returns the right object reference to use, no matter if the Zoiper Web instance is loaded from <object> or <embed> tag, thus making browser or technology independent.

The properties and methods of the "Phone" class can be organized in a several groups as described below:

#### 3.1.1 Call management

- property Call - read only - this property returns a class reference to a "Call" object referring to the current active call. If there is no active call, "null" is returned.
- method Dial(number) - returns "Call" object reference - this method creates a new "Call" object and dials the number provided with the parameter "number". Any current active "Call" is put on hold and the new "Call" becomes the active call. The "Call" is made using the current active "Account". If there is no active "Account" selected, the first "Account" in the list is going to be used. If "Call" creation fails or there are no accounts at all this method will return "null".
- method Hang() - no return value - this method hangs up the active call.

#### 3.1.2 Account management

- method UseAccount(name) - returns the "Account" object reference - this method selects the current active "Account" according to the parameter "name". If an "Account" with that "name" does not exist, "null" is returned.
- method GetAccount(name) - returns the "Account" object reference - this method is used to get the account object reference by its "name". If an account with that "name" does not exist, "null" is returned.

- method AddAccount(name,protocol) - returns "Account" object reference - this method creates a new "Account" or gets an existing one with that "name". If the account is created, it uses the parameter VoIP "protocol". The "protocol" parameter can have the following values "SIP" or "IAX". If there is an error while creating the "account", "null" is returned. The "name" of the "Account" is used as unique identifier.
- method DelAccount(name) - no return value - this method deletes an "Account" with the parameter "name", if it exists.
- method Login(username,password) - no return value - this method is used to login Zoiper Web to the Zoiper service using the parameters "username" and "password".
- method Logout() - no return value - this method is used to logout Zoiper Web from the Zoiper service.

### 3.1.3 Contacts management

- method GetContact(phone) - returns a "Contact" object reference - this method is used to find an existing "Contact" object by its primary phone number given by the parameter "phone". If a "Contact" with that name does not exist, "null" is returned.
- method AddContact(phone) - returns the "Contact" object reference - this method is used to create a new "Contact" or retrieve an existing one with the "phone" number as a parameter.
- method DelContact(phone) - no return value - this method is used to delete an existing "Contact" by its "name".
- method SavePrivacy() - no return value - this method saves the privacy rules to the data folder on the web visitors pc. Zoiper Web will not save them locally if you do not use this method. If privacy rules exist, they will be loaded upon phone startup.

### 3.1.4 Configuration management

- property Version – read only – this property is used to obtain the version string of the currently running Zoiper Web. It can be used to inform the user that there is a newer version and help him download and install it.
- property SaveHistory – read/write – this property is used to make Zoiper Web save its history upon receiving new history events. The history file is named “History.txt”. It is located in subfolder named “Zoiper Web” placed in the current user’s application data folder.
- property SaveContacts – read/write – this property is used to make Zoiper Web save its contact entries upon creation and editing. The contacts file is named “Contacts.xml”. It is located in subfolder named “Zoiper Web”, placed in the current user’s application data folder. Note that you can select if the contacts that are created through the API will be saved locally or not.

- method `SaveOptions()` - no return value - this method saves the Zoiper Web configuration to the data folder on the web visitors pc. Zoiper Web will not save them locally if you do not use this method. If options exist, they will be loaded upon phone startup.
- method `GetConfig()` - returns "Config" object reference – this method is used to obtain the reference to the object of class "Config", which is responsible for the general configuration of the phone. There is only one instance of that class that can be obtained only from this method.
- method `ShowAudioWizard()` - no return value - this method shows the audio wizard dialog.
- callback `OnZoiperReady(phone)` - this callback function is called by Zoiper Web when it is done loading and ready for use. It provides an object reference to the "Phone" class as a parameter. This function will be called before any other callback. Don't use any methods or properties before this callback is called.

### 3.1.5 Miscellaneous properties and methods

- method `Showlog()` – no return value – this method will show a log window containing text information about the actions of Zoiper WEB. It could also provide debug information if "Config.EnableDebugLog" is enabled.
- method `CheckCertificate(url)` – no return value – this method is used to validate the Zoiper Web certificate from a different URL than the one used by default. The default certificate search path is in the root folder of the site Zoiper Web is loaded from and the file is called "activex.certificate". Use this method only if it is impossible to place the file in your root folder.

### 3.2 "Config" class

The "Config" class encapsulates the Zoiper Web general configuration options. An instance of that class can only be obtained from the "Phone" class by using the "GetConfig" method.

- property `PopupMenuOnIncomingCall` - read/write - this property is used to enable and disable the showing of a popup menu when there is a new incoming call. This property can have the values "true" or "false".
- property `OnTransferRequest` - read/write - this property is used to select the behavior of Zoiper Web when there is a transfer request from the remote peer. This property can have the following three values : "accept" - always accept the requests automatically; "reject" - always reject the requests automatically; "ask" - ask the user what action has to be taken.
- property `UseEchoCancelation` - read/write - this property is used to enable and disable echo cancellation and can have the value "true" or "false".

- property DebugLogPath - read/write - this property is used to set the path where a debug log file will be created if the EnableDebugLog is set to “true”
  
- property EnableDebugLog - read/write - this property is used to enable and disable debug logging in the Zoiper Web softphone.
  
- property Restrictions - read/write - this property is used to apply restrictions to Zoiper Web interface. The restrictions entries are comma separated values and can be used in any combination. The valid values for the restrictions are as follows:
  - options\_sip\_accounts – restricts access to the options for SIP accounts as a whole.
  - options\_iax\_accounts – restricts access to the options for IAX accounts as a whole.
  - options\_sip\_accounts\_overview – restricts access to the options SIP accounts overview.
  - options\_iax\_accounts\_overview – restricts access to the options IAX accounts overview.
  - options\_sip\_new\_account – restricts the creation of new SIP accounts.
  - options\_iax\_new\_account – restricts the creation of new IAX accounts.
  - options\_audio\_general – restricts access to the audio general options panel.
  - options\_audio\_devices – restricts access to the audio devices options panel.
  - options\_audio\_codecs – restricts access to the audio codecs options panel.
  - options\_general – restricts access to the general options panel.
  - options\_call\_events – restricts access to the general call events options panel.
  - options\_integration – restricts access to the general integration options panel (not available in all versions).
  - options\_provision – restricts access to the provisioning options panel (not available in all versions).
  - options\_sip – restricts access to the SIP protocol options panel.
  - options\_iax – restricts access to the IAX protocol options panel.
  - options\_rtp – restricts access to the RTP protocol options panel.
  - options\_stun – restricts access to the STUN protocol options panel.
  - options\_diagnostics – restricts the diagnostic options panel.
  - options\_network – restricts access to the network options panel.
  - options\_fax – restricts access to the fax options panel.
  - options\_video – restricts access to the video options panel (not available in all versions).
  - options\_chat – restricts access to the chat options panel (not available in all versions).
  - options\_contact\_server – restricts access to the contact server options panel (not available in all versions).
  - options\_show\_advanced – restricts access to the “Show advanced options” check box in the options window thus disabling all the advanced options access.
  - options – restricts the showing of the options window.
  - dial – restricts the ability to dial.
  - hangup – restricts the ability to hang up.
  - transfer – restricts the ability to transfer.
  - hold – restricts the ability to hold.
  - record – restricts the ability to record calls (not available in all versions).
  - conference – restricts the ability to start a conference call (not available in all versions).

- . auto\_answer – restricts the ability to auto answer calls (not available in all versions).
- . fax – restricts the ability to send faxes (not available in all versions).
- . video - restricts the ability to start video calls (not available in all versions).
- . history – restricts the ability to view the history.
- . contacts – restricts the ability to view the address book / contacts.
- . dialpad – restricts the access to the dial pad.
- . chat – restricts the access to the chat window (not available in all versions).
- . chat\_history - restricts the access to the chat history (not available in all versions).
- . save\_contact – restricts access to the save contact button (not available in all versions).
- . find\_contact – restricts access to the find contact button (not available in all versions).
- . send\_message – restricts access to the send message button (not available in all versions).
- . account\_register – restricts access to the account register button.
- . account\_unregister – restricts access to the account unregister button.
- . wizard\_audio – restricts access to the audio wizard button.

- property ServerSideAutoAnswer - read/write - this property is used to enable and disable the ability of the phone to favor SIP “Call-Info” auto answer headers (not available in all versions).

- property AutoAnswer - read/write - this property is used to enable and disable the auto answer feature of the phone (not available in all versions).

- property AutoAnswerSeconds - read/write - this property is used to set the time that should pass before the call is going to be auto answered. Note that this property must be set before “AutoAnswer” property is set to “True” (not available in all versions).

- property AutoAnswerPlaySound - read/write - this property is used to enable and disable the sound that is played upon auto answering (not available in all versions).

- property AudioInputVolume - read/write - this property is used to control the input (microphone) volume. It takes values between “1” and “100”.

- property AudioOutputVolume - read/write - this property is used to control the output (speaker/headphones) volume. It takes values between “1” and “100”.

- property MediaDSCP - read/write - this property is used to control the DSCP tagging for the outbound media streams. It can take one of the following values "NONE", "CS0", "CS1", "CS2", "CS3", "CS4", "CS5", "CS6", "CS7", "AF11", "AF12", "AF13", "AF21", "AF22", "AF23", "AF31", "AF32", "AF33", "AF41", "AF42", "AF43" and "EF". Note that Windows might not favor this setting unless the following registry key is not set to “0” :

HKEY\_LOCAL\_MACHINE\System\CurrentControlSet\Services\Tcpip\Parameters\DisableUserTOSSetting

- property RTPPortBase - read/write - this property is used to control the port base used for creating RTP sessions. It should have an even value as by RFC recommendations even ports are used for RTP connections and odd ports are used for RTCP connections. Both connections are always present. If an odd value is configured given the next even value will be used as a port base instead. Values between "1" and "65500" are accepted but using ports below 1024 is not recommended as port collision can occur very easily in those cases.
- property RTPUseRandomPort - read/write - this property is used to enable and disable the usage of the random RTP port base. The random port base will be randomly selected between 32000 and 65500. This property can have the value "true" or "false".
- property SIPTLSCertificateFile - read/write - this property is used to specify an additional TLS certificate file path. The file should be in PEM format and could hold several certificates (not available in all versions).
- property Forwarding - read/write - this property is used to enable and disable the automatic call forwarding. If there is no value set for the "ForwardingExtension" property no forwarding will take place. This property can have the value "true" or "false" (not available in all versions).
- property ForwardingSeconds - read/write - this property is used to set the time in seconds after which the call will be forwarded to the number set in the "ForwardingExtension" property. This property can have the values between "0" and "60000" (not available in all versions).
- property ForwardingExtension - read/write - this property is used to set the number to which the call will be forwarded (not available in all versions).
- property RecordCalls - read/write - this property is used to enable and disable the automatic call recording. When this property is enabled all calls will be recorded. The recordings will be located in the path specified by the "RecordPath" property and will be named with prefix "recorded\_conversation\_" and date and time of the call. Recordings are saved in ".wav" file format. This property can have the value "true" or "false" (not available in all versions).
- property RecordPath - read/write - this property is used to set the path that is going to be used for storing the call recordings (not available in all versions).

### 3.3 "Account" class

The "Account" class represents a Zoiper Web connection to a given VoIP service, its properties and basic functions. Zoiper Web can't make calls without an "Account" class instance. Note that some VoIP services will allow making calls without registering first while others don't. It depends on the service. Changing the properties will take effect only after the "Apply" method is called.

- property Name - read only - this property is used to read the "Account" name. It acts as a unique identifier for the "Account".
- property Domain - write only - this property is used to set the SIP domain and port for the "Account". To set the port you have to use the form <domain>:<port>
- property OutboundProxy - write only - this property is used to set the SIP outbound proxy address and port for the "Account". To set the port you have to use the form <domain>:<port>
- property Host - write only - this property is used to set the IAX2 host and port for the "Account". To set the port you have to use the form <domain>:<port>
- property Context - write only - this property is used to set the IAX2 context for the "Account".
- property CalledID - write only - this property sets the caller identification name for the "Account" for both SIP and IAX2.
- property Phone - write only - this property sets the caller identification number for the "Account" for IAX2.
- property UserName - write only - this property sets the user name used by the VoIP service to identify the "Account".
- property AuthenticationUserName – write only – this property sets the user name used by the VoIP service for the authentication of the “Account”. Note that in most cases “UserName” and “AuthenticationUserName” are the same and there is no need to set this property explicitly.
- property Password - write only - this property sets the password used by the VoIP service to authenticate the "Account".
- property STUNHost - write only - this property sets the SIP STUN server host address for this "Account". If an empty string is set the STUN usage for this "Account" is disabled. The STUN usage is disabled by default.
- property STUNPort - write only - a property used to set the port for a SIP STUN server.
- property RegistrationExpiry - write only - this property is used to set the registration expiry time for the “Account” in seconds. For SIP accounts the default value is “3600” seconds and for IAX2 accounts it is “60” seconds. Note that this is a suggestion zoiper will make to the VoIP service and can be overridden by the VoIP service itself.

- property `DisableRingbackTones` - write only - this property is used to enable and disable the playback of ringback tones when Zoiper Web is waiting for the remote peer to answer using this “Account”. This property can have the value “true” or “false”.
  
- property `UseRPort` - write only - this property is used to enable and disable the use of RPort for this “Account” and can have the value “true” or “false”.
  
- property `UseRPortMedia` - write only - this property is used to enable and disable the use of RPort for media for this “Account” and can have the value “true” or “false”.
  
- property `ForceRFC3264` - write only - this property is used to enable and disable the forced use of RFC-3264 for this “Account” and can have the value “true” or “false”.
  
- property `PresenceModel` - write only - this property is used to set the presence model used by this “Account”. It can have the following values : “server\_based” – in this case the server based presence model will be used; “peer-to-peer” – in this case the peer to peer will be used.
  
- property `DTMFType` - write only - this property is used to set the DTMF type used by this “Account”. It can have the following values :
  - “media\_outband” - in case of SIP “Account”, RFC-2833 is going to be used and in case of IAX2 “Account”, the normal out-of-band DTMF signaling will be used;
  - “signalling\_outband”- for SIP is going to use the SIP INFO method for sending DTMF signals and for IAX2 the same type as “media\_outband” is used;
  - “media\_inband” - for both SIP and IAX2 accounts inband DTMF signals will be generated;
  - “disabled” - is going to disable the sending of DTMF signals.
  
- property `SIPTransportType` - write only - this property is used to set the transport type for the SIP protocol. It can have the following values :
  - “udp” - UDP transport protocol will be used. This is the default value;
  - “tcp”- TCP transport protocol will be used;
  - “tls” - TLS protocol over TCP protocol will be used; TLS protocol is used to encrypt the SIP data using the certificates already installed on the machine. Additional certificates can be added using the “Config.SIPTLSCertificateFile” property.  
(not available in all versions).
  
- property `RTPEncryption` - write only - this property is used to set the RTP encryption method. It can have the following values :
  - “none” - No encryption will take place. This is the default value;
  - “sdes”- SDES encryption will be applied to the RTP streams using this “Account”. Note that this encryption method can be used only with “SIPTransportType” set to “tls” because the encryption key should be transferred between both sides in a secure way.  
(not available in all versions).

- method `Apply()` - no return value - this method applies any changes made to the "Account" object properties.
- method `Register()` - no return value - this method tries to register the "Account" with the VoIP service specified in its properties. Note that this method must be called after the "Apply" method. The result of the register attempt will be returned in the "Callback's" "OnZoiperAccountRegister" or "OnZoiperAccountRegisterFail".
- method `Unregister()` - no return value - this method tries to unregister the "Account" with the VoIP service specified in its properties. Note that this method must be called after the "Apply" method. The result of the unregister attempt will be returned in the "Callback's" "OnZoiperAccountUnregister" or "OnZoiperAccountRegisterFail".
- method `AddCodec(codec)` - no return value - this method is used to add a "codec" that can be used by this "Account". The "codec" parameter can have the following values: "GSM", "a-law", "u-law", "Speex", "iLBC 30" and "iLBC 20". The first call to this method will enable the custom codec list for that "Account". If this method is not called, the "Account" will use the global codec list, which by default has all the available codecs enabled.
- method `ClearCodecs()` - no return value – this method is used to disable the custom codec list for that "Account".
- callback `OnZoiperAccountRegister(account)` - this callback function is called by Zoiper Web upon a successful registration of an "Account". Object reference to the "Account" is specified as a parameter.
- callback `OnZoiperAccountUnregister(account)` - this callback function is called by Zoiper Web upon a successful unregistration of an "Account". Object reference to the "Account" is specified as a parameter.
- callback `OnZoiperAccountRegisterFail(account)` - this callback function is called by Zoiper Web upon a failed registration or unregistration of an "Account". Object reference to the "Account" is specified as a parameter.

### 3.4 "Call" class

The "Call" class represents a call instance in Zoiper Web, its properties and methods. The "Call" is always done using an "Account" but there are cases of incoming calls which can't be matched to an existing "Account". In this case the "Account" property holds a temporary "Account" created just for that "Call".

- property `Phone` - read only - this property holds the caller identification name if this is an incoming call or the number dialed if this is a outgoing call.

- property DNID - read only – in case of an incoming call this property will hold the number that the other end has dialed to make the call.(Dialed Number Identifier)
- property Account - read only - this property holds a reference to an "Account" object used for this "Call".
- property Contact - read only - this property hold a reference to a "Contact" object used for this "Call". When a phone is dialed, Zoiper Web tries to find a matching phone number in the contact list and assigns the "Call" to the found "Contact". If there is no matching "Contact", a temporary one is created for that "Call".
- property Duration - read only - this property returns the "Call" duration in seconds. If this is an incoming "Call" and it is not picked up yet, it will show the duration of the ringing. After "Call" pickup this property is reset to zero.
- property InConference – read / write - this property is used to put or remove the “Call” in and from conference. Zoiper Web supports only one conference at given time. This property can have the value “true” or “false”. Set this property to “true” to make the “Call” join or set it to “false” to make the “Call” leave the conference. Reading the property will return information if the “Call” is in conference (not available in all versions).
- method Hang() - no return value - this method is used to hang up the "Call". After calling this method no other references to this "Call" object should be made as it is destroyed.
- method Hold() - no return value - this method is used to hold the "Call". If the "Call" is the active call in the "Phone" object, the active call will be set to "null".
- method UnHold() - no return value - this method is used to unhold the "Call". If there is another "Call" object in the "Phone" object it will be put on hold automatically and this "Call" will become the active call. This method does the same as the “SetActive” method.
- method SetActive() - no return value - this method is used to set the specified “Call” to be the active call thus holding any other “Call” that might be active at this moment. It will also unhold this “Call” and will result in visual line change. This method does the same as the “UnHold” method.
- method Accept() - no return value - this method is used to accept the "Call" if it is incoming.
- method Reject() - no return value - this method is used to reject the "Call" if it is incoming.
- method Transfer(number) - no return value - this method is used to make an unattended transfer to another "number" given as a parameter. Note that you can't transfer a "Call" using a certain VoIP service towards another VoIP service.

- method SendDTMF (dtmf\_sequence) - no return value - this method is used to send a DTMF sequence. The parameter "dtmf\_sequence" could contain a single character or a sequence of characters which are converted to DTMF signals according to the "account", used for this "call", property "DTMFType". The valid characters used by this method are "0", "1", "2", "3", "4", "5", "6", "7", "8", "9", "a", "b", "c", "d", "\*" and "#". Note that using "DTMFType" "signalling\_outband" could result in slower sending of the DTMF signals because confirmation must be received from the remote end.
- callback OnZoiperCallFail(call) - this callback function is called by Zoiper Web when a "Call" failed for some reason. Object reference to the "Call" is specified as a parameter.
- callback OnZoiperCallRing(call) - this callback function is called by Zoiper Web when a "Call" has finished the dialing stage and the remote peer is ringing. Object reference to the "Call" is specified as a parameter.
- callback OnZoiperCallHang(call) - this callback function is called by Zoiper Web when a "Call" is hung up. Object reference to the "Call" is specified as a parameter.
- callback OnZoiperCallHold(call) - this callback function is called by Zoiper Web when the remote peer puts the "Call" on hold. Object reference to the "Call" is specified as a parameter.
- callback OnZoiperCallUnhold(call) - this callback function is called by Zoiper Web when the remote peer resumes (unholds) the "Call". Object reference to the "Call" is specified as a parameter.
- callback OnZoiperCallAccept(call) - this callback function is called by Zoiper Web when the remote peer has accepted the "Call". The object reference to the "Call" is specified as a parameter.
- callback OnZoiperCallReject(call) - this callback function is called by Zoiper Web when the remote peer has rejected the "Call". Object reference to the "Call" is specified as a parameter.
- callback OnZoiperCallIncoming(call) - this callback function is called by Zoiper Web when there is a new incoming "Call". Object reference to the "Call" is specified as a parameter.

### 3.5 "Contact" class

The "Contact" class represents a single contact in the Zoiper Web address book. It is used to store information about the "Contact" properties.

- property Name - read only - this property is used to get the "Contact" display name or, if the display name is not set, it will return a construct from the first name, middle name and last name in that order.
- property Phone - read only - this property is used to obtain the "Contact's" primary phone.

- property Account - read/write - this will set or get the "Account" object name associated with this "Contact".
- property Display - read/write - this property sets the "Contact's" display name.
- property FirstName - read/write - this property sets the "Contact's" first name.
- property LastName - read/write - this property sets the "Contact's" last name.
- property MiddleName - read/write - this property sets the "Contact's" middle name.
- property Country - read/write - this property sets the "Contact's" country.
- property City - read/write - this property sets the "Contact's" city.
- property WorkPhone - read/write - this property sets the "Contact's" work phone number.
- property HomePhone - read/write - this property sets the "Contact's" home phone number.
- property CellPhone - read/write - this property sets the "Contact's" cell phone number.
- property FaxNumber - read/write - this property sets the "Contact's" fax phone number.
- property SaveContact - read/write - this property is used to control if the "Contact" is going to be stored locally. Note that if the "Phone" property "SaveContacts" is set to "false" the "Contact" is not going to be saved regardless what is set to this property.
- method Apply() - no return value - this method applies any changes made to the "Contact" object properties.
- callback OnZoiperContactStatus(contact,status) - this callback function is called by Zoiper Web when the "Contact's" status has changed. The object reference to the "Contact" is specified as a parameter. The second parameter is called "status", it represents the status of the "Contact" and can be one of the following values "online", "offline", "away", "busy", "phone" and "lunch".

#### 4. Example

Here is an example on how to use Zoiper Web on the supported browsers. The explanations on the example are inline as comments. The comment references the next line or block of lines after it.

```
<html>
  <head>
    <title>Zoiper Web Example</title>
  </head>
```

```

<!-- We give the body a function to be called upon page unload for finalization of Zoiper Web -->
<body onunload="Quit()">
  <script type="text/javascript">
<!-- The variable "Zoiper" is used to keep reference about the Zoiper Web instance. It is going
to hold the "Phone" object -->
    var Zoiper;
<!-- The variable "ActiveCall" is used to keep a reference to the current active "Call" object -->
    var ActiveCall;
<!-- The function GetValue is a helper function used to get the value of a document element by
its ID -->
    function GetValue(name)
    {
        return document.getElementById(name).value;
    }
<!-- The function "Quit" is used to stop Zoiper Web, we will use it to delete the items we have created during the
example -->
function Quit()
    {
        Zoiper.DelContact("web demo");
        Zoiper.DelAccount("Sample");
<!-- On Firefox new instances of the plugin are created before old ones are destroyed
Because of this we must manually destroy the existing plugin -->
        document.getElementById('ZoiperA').innerHTML = "";
    }
<!-- The function "Hang" is a helper function and is used to hangup the "ActiveCall" that we keep reference of --
>
function Hang()
    {
        if (null != ActiveCall)
            ActiveCall.Hang();
        ActiveCall = null;
    }
<!-- The function "Dial" is used to dial a number entered by the user in the input field with ID="number". It also
assigns the newly created "Call" object to the "ActiveCall"
variable -->
function Dial()
    {
        ActiveCall = Zoiper.Dial(GetValue("number"));
    }
<!-- The function "Hold" is used to put the current active call on hold -->
function Hold()
    {

```

```

        if (null != ActiveCall)
            ActiveCall.Hold();
    }
<!-- The function "Login" is used to login the user to the Zoiper Service. It uses the username and password
provided by the user in the input fields "user" and "pass" -->
function Login()
    {
        var user = GetValue("user");
        var pass = GetValue("pass");
        Zoiper.Login(user,pass);
    }
<!-- The function "Logout" is used to logout the user from the Zoiper Service. -->
function Logout()
    {
        Zoiper.Logout();
    }
<!-- The function "Status" is used to show a status text in the element with ID="Status". It is
used as a log function to show the state of the phone and what events are triggered -->
function Status(text)
    {
        var node = document.getElementById("Status");
        node.innerHTML = text;
        // node.innerHTML = node.innerHTML + "<br/>" + text;
    }
<!-- The function "OnZoiperReady" is the entry point for Zoiper Web usage. It is called by Zoiper Web when it
is ready for use. It provides a reference to its "Phone" object which we assign to the "Zoiper" variable. We use it
to make the initial setup -->
function OnZoiperReady(phone)
    {
        Zoiper = phone;
<!-- Here we get the "Config" object instance and put it in the variable named Config-->
        var Config = Zoiper.GetConfig();
<!-- Here we set some general configuration properties. We disable the popup menu on incoming calls and we
set the debug log to be written to "D:\\" and enable it-->
        Config.PopupMenuOnIncomingCall = "false";
        Config.DebugLogPath = "d:\\";
        Config.EnableDebugLog = "true";
<!-- Here we create or get an existing "Account" object called "Sample". The "Account" is going to use the SIP
protocol -->
        Account = Zoiper.AddAccount("Sample", "sip");
<!-- Here we set the "Account" properties -->
        Account.Domain = "10.2.1.9:6060";

```

```

    Account.CallerID = "888";
    Account.UserName = "888";
    Account.Password = "888";
<!-- Here we enable the STUN usage -->
    Account.STUNHost = "stun.zoiper.com";
    Account.STUNPort = "3478";
<!-- Here we enable the custom codec list and select only the GSM codec -->
    Account.AddCodec("gsm");
<!-- Here we set the "Account" to use inband DTMF signals -->
    Account.DTMFType = "media_inband";
<!-- Here we apply the so far set properties and register the "Account" -->
    Account.Apply();
    Account.Register();
<!-- Here we select the "Sample" "Account" to be the active one -->
    Zoiper.UseAccount("Sample");
<!-- Here we create a new "Contact" object with primary number "web demo" and set its properties -->
    var Contact    = Zoiper.AddContact("web demo");
    Contact.Account = "Sample";
    Contact.Display = "web demo display";
    Contact.FirstName = "John";
    Contact.MiddleName = "F.";
    Contact.LastName = "Doe";
    Contact.Country = "Alabama";
    Contact.City = "Huntsville";
    Contact.WorkPhone = "work";
    Contact.HomePhone = "home";
    Contact.CellPhone = "cell";
    Contact.FaxNumber = "fax";
    Contact.Apply();
}
<!-- Here we have a simple implementation of the available "Callback" functions -->
function OnZoiperCallFail(call)
{
    Status(call.Phone + " failed");
}
function OnZoiperCallRing(call)
{
    Status(call.Phone + " ring");
}
function OnZoiperCallHang(call)
{
    Status(call.Phone + " hang");
}

```

```
}  
function OnZoiperCallHold(call)  
{  
  Status(call.Phone + " hold");  
}  
function OnZoiperCallUnhold(call)  
{  
  Status(call.Phone + " unhold");  
}  
function OnZoiperCallAccept(call)  
{  
  Status(call.Phone + " accept");  
}  
function OnZoiperCallReject(call)  
{  
  Status(call.Phone + " reject");  
}  
function OnZoiperCallIncoming(call)  
{  
  Status(call.Phone + " incoming");  
}  
function OnZoiperAccountRegister(account)  
{  
  Status(account.Name + " is registered");  
}  
function OnZoiperAccountUnregister(account)  
{  
  Status(account.Name + " is unregistered");  
}  
function OnZoiperAccountRegisterFail(account)  
{  
  Status(account.Name + " failed to register");  
}  
function OnZoiperContactStatus(contact,status)  
{  
  Status(contact.Name + " is " + status);  
}  
</script>
```

<!-- Here we define the HTML elements we are going to need in this example -->

```
<p id="Status">Ready</p>
```

```
<table>
```

```
<tr>
```

```

    <td>Number</td>
</tr>
<tr>
    <td><input id="number" type="text"/></td>
    <td></td>
    <td><button onclick="Dial()">Dial</button></td>
    <td><button onclick="Hold()">Hold</button></td>
    <td><button onclick="Hang()">Hang</button></td>
</tr>
<tr>
    <td>Login</td>
    <td>Password</td>
</tr>
<tr>
    <td><input id="user" type="text"/></td>
    <td><input id="pass" type="text"/></td>
    <td><button onclick="Login()">Login</button></td>
    <td><button onclick="Logout()">Logout</button></td>
</tr>
</table>

```

<!-- Here we define the <object> tag used by Internet Explorer to include Zoiper Web in the web page. Note that the version info in the "codebase" attribute is very important when a more recent Zoiper Web is available and should be upgraded -->

```

    <object id="ZoiperA" classid="clsid:BCCA9B64-41B3-4A20-8D8B-E69FE61F1F8B"
CODEBASE="http://www.zoiper.com/webphone/InstallerWeb.cab#Version=1,17,0,6802" align="left"
width="434" height="236" >

```

<!-- Here we define the <embed> tag used by the NPAPI based browsers (FireFox, etc) to include Zoiper Web in the web page.-->

```

    <embed id="ZoiperN" type="application/x-zoiper-plugin" align="left" width="434" height="236"/>
</object>
</body>
</html>

```

## 5. Contact

**support@zoiper.com**