VOXI API

VSys Live

VSys Anywhere

(Sample only)

Note that most commands show as "Bookmark not defined" in this document; the complete VOXI API is only available under NDA.

Introduction

VOXI, the VSys One eXchange Interface, provides an API to the business logic and data stored in VSys One. All communication with VOXI is done using HTTP on a port specified in the `VSys.ini` file.

VOXI is a multithreaded Delphi application built with the same code as VSys One. This allows changes and enhancements made to these applications to be trivially available to VOXI and the tools which call it.

Databases

External applications do not need, or in general want, to directly access the data tables use by VSys One, GMS 6 and VOXI. Unless otherwise indicated, the data stored in these tables is not documented nor intended for direct access. Where it is documented, it may be used for querying but not updating or deleting.

Simultaneous access

By default, five simultaneous requests can be processed by VOXI; any additional requests are queued while the others are processed (this setting is not configurable at this time). If VOXI is unable to process an incoming request within ten seconds it will respond with:

```
        503: SERVER TOO BUSY TO ALLOCATE A PROCESSING ENGINE FOR YOU (xx IN USE)
```

where “xx” is the number of active processors in use.

Where feasible, if an application or tool has multiple requests or actions to be performed, submit these in bulk in a single VOXI request. This uses only one command processor and the requests are handled sequentially, greatly reducing the load on VOXI.

VOXI does not provide implicit locking. Editing a person, posting hours, registering for courses: it makes no attempt to prevent multiple instances or even multiple sessions on the same VOXI instance from doing something rude and self-destructive by making simultaneous changes to the same person.

Pre-Shared Key

VOXI can require authentication via a pre-shared key (PSK) to access. This does not imply encryption of the connection between the calling application and VOXI; any encryption (SSL/TLS or others would need to be enabled using a network proxy), but does prevent callers from executing commands on VOXI without that key.

The PSK can takes the form of a string of any length. To use one, enter it in the `VSys.ini` file:

```
        [VOXI]
        PSK=ThisIsOurLongKeyWithSomeRandomDataDB34UIT34I5UGTC78589378
```

Any `<VOXI>` command must include this value or the command will be rejected.

```
<VOXI session="123" PSK="ThisIsOurLongKeyWithSomeRandomDataDB34UIT34I5UGTC78589378">
   <REQUEST command="sys.versioninfo"/>
</VOXI>
```

Configuring VOXI

VOXI itself has few configurable elements, most centered around how VOXI is communicated with and the database it uses. Most of the logic for VSys One is defined within the application, including how various web aspects are handled and displayed.

Where does VOXI look for these values?

- If `/INISECTION:ABC123` is included on the command line, VOXI will use the `[ABC123]` section of the `VSys.ini file`, regardless of the /SERVICENAME parameter.
- If `/SERVICENAME:ABC` is included on the command line, VOXI will look to the registry value `\HKLM\CurrentControlSet\Services\ABC\IniSection`.

Otherwise VSys will look in the `[VOXI]` section of the `VSys.ini` file.

Port, monitor port, location geocoding, localization disabling, command logging and site are generally configured by the VSys Live installer. Other settings can be placed within the "Additional VOXI settings" for a specific site for automatic inclusion in the appropriate section of the `VSys.ini` file during installation.

Port

VOXI's listening port is defined in the `VSys.ini` file appropriate section. Absent a value here, VOXI will listen on port 80.

```
Port=99
```

Site

```
Site=MX1KA8NDVHC0YZM0
```

Tells VOXI to start up using the settings for the given defined VSys Live site. If the site doesn't exist or is not defined to use alternate behavior (its own website settings) then VOXI will use the default global settings instead.

Command lockouts

For cases where VOXI should only be permitted to respond to certain commands, or where specific commands must be inhibited, VOXI supports command lockouts. In the `VSys.ini` file, in appropriate section, enter these commands in explicit or wildcard format.

```
validCommands=org.*,sys.*
invalidCommands=sys.versioninfo
```

In this example, any command which begins with `org.` or `sys.` will be valid with the exception of `sys.versioninfo`. If `validCommands` is left blank then all commands are considered to be valid, except those excluded by `invalidCommands`.

Maximum active processing engines

The maximum number of internal processing engines is defined in the `VSys.ini` file. Absent a value here, VOXI will use a maximum of five processing engines. Values less than1 will be assumed to be `1`; greater than `100` will be assumed to be `100`.

```
maxActiveProcessEngines=10
```

Trace purging

VOXI will purge the `voxilogs` table on a scheduled basis to keep its size down. Any log records which exceed the number of hours specified here will be removed. Absent a valid value here, one hour will be used.

```
TracePurgeWindow=72
```

Engine retirement

VOXI will "retire" processing engines (think Blade Runner) after they've reached a certain age or lifetime activity. Configure the retirement behavior with two settings:

```
maxEngineLife=10
maxEngineOperations=5
```

`MaxEngineLife` is the maximum permitted lifetime, in minutes, of an engine; `MaxEngineOperations` is the maximum permitted number of requests that an engine may have processed. (Any engine which experiences an internal failure/exception is always immediately retired but not freed; it continues to remain in memory to avoid its damaging other engines.)

Overhead logging

    overheadLogging=1

Enables the posting of diagnostic timing records to the `voxilogs` table. This is an expensive operation (time) as each request may generate dozens of `voxilogs` records, uses up a substantial amount of disk space, and can severely bog down the underlying database server with queries.

Test mode

    testmode=1

Used for diagnostics with the VOXI Hammer: causes VOXI to attempt to return the same result for a request as it did in the past, even if that information is out of date, and limits the precision of various returned values. Not intended for use in a production environment.

Disk logging

    diskLogging=1

By default VOXI logs all commands and their results to the `voxilogs` table. It can also log complete requests and the results from those and will do so if `diskLogging=2`. Set `diskLogging=-1` to prevent all such logs; this is helpful for performance reasons but limits diagnostic abilities using the VOXI logs monitoring tool in VSys One.

Geocoding

    NoGeocoding=1

The various location query commands will, if the location hasn't been geocoded, attempt to do so using the Google Maps API when locations are returned. Setting this value prevents that; this is usually only required in heavily locked-down environments in which VOXI is unable to get to the Google Maps API and. If the command `loc.list` and its variants seem inordinately slow, try this setting.

Localization

    NoLocalization=1

Disables the localization tools, has the same effect as not putting the `VSys.bld` file in the same folder as `VOXI.exe`.

Caching

    CachedObjectsLifetime=120

When an image is extracted from a news item or message and provided to the caller via VOXI's `/cache/ GET` interface, this value defines how long the image should remain available via that interface. From this many minutes after the last time the image was placed into that cache by VOXI (as a result of its being extracted from a news item, etc.), the image will expire. Calls to `/cache/` do not reset the lifetime counter here, but later extractions of the image do reset the counter. Default value is sixty minutes; minimum configurable value is five minutes.

Background operations

    BackgroundOps=30000

This overrides the delay, in milliseconds, between runs of the background maintenance thread. This thread validates processors, reaps expired processors, checks for forced settings refresh, runs background operations commands, and sets up pre-initialized advanced processors. Default value is `30000` (30 seconds).

    BackgroundOpsCommand=60000

This sets the delay, in milliseconds, of the background operations command. That command runs the kiosk auto-checkout, and knowledgebase rebuilder tools. This command is run by the background maintenance thread and thusly cannot run any more often than that thread does. Default value is `30000` (30 seconds).

Monitoring

VOXI listens by default on a second port – 8080 – to which monitor requests can be sent. Any HTTP `GET` or `POST` request returns the basic status of VOXI:

```
<VOXIMONITOR CommandVersion="0.0.0.5" Version="0.1.0.181">
   <STATUS ActiveRequests="0" Now="2011/07/16 00:31:43.971">
      <PORT>80</PORT>
      <DATABASE Name="temp2" ServerName="NexusDB" Address="Ishmael"/>
      <ENGINES Active="0" Available="0"/>
   </STATUS>
</VOXIMONITOR>
```

Requests on this port do not use any database connections nor any processing engines, and so are more likely to be responded to if VOXI itself has encountered problems.

The port on which VOXI listens for monitoring requests is set with:

```
monitorPort=8888
```

The current status can also be retrieved with an HTTP `GET` request to `/status/` on the current primary port. Note that calls to `/status/` require a processing engine to complete; if VOXI is having problems allocating or using processing engines, a request on the monitoring port is more likely to succeed.

Source filtering

By default VOXI accepts command and monitor requests from any remote IP address. To filter the addresses from which VOXI will accept commands, use:

```
IPAcceptCommands=10.2.1.0/24,192.168.1,101
IPRejectCommands=10.2.1.21
IPAcceptGet=10.2.1.0/24
IPRejectGet=
IPAcceptMonitor=10.2.1.0/24
IPRejectMonitor=10.2.1.15.10.2.1.213
```

For any of these four settings VOXI accepts a comma-delimited list of IP addresses with an optional netmask for each. Commands or monitor requests rejected due to the requester's address are returned a `403 Forbidden` and the fact that the command/request was made is logged but the details of the command/request are not.

Notes:

- If `IPAcceptCommands` is blank, all addresses are valid for commands except those defined in `IPRejectCommands`. This affects HTTP POST commands made to the standard port.
- If `IPAcceptGet` is blank, all addresses are valid for HTTP GET commands except those defined in `IPRejectGET`. This affects HTTP GET commands made to the standard port.
- If `IPAcceptMonitor` is blank, all addresses are valid for monitor requests except those defined in `IPRejectMonitor`. This affects all GET and POST commands made to the monitoring port (by default 8080).
- A command which comes from Apache (or other application) will, to VOXI's logic, have originated from the machine on which that application is running. Do not allow Apache or other applications to forward arbitrary commands to VOXI!

VOXI GET Requests

Certain limited HTTP `GET` requests are supported by VOXI.

/status/

Returns the same XML value as a request of any type on the monitoring port. Can limit its output to specific aspects of the status by including one or more of the following options, e.g.

```
/status/running
/status/problems
/status/engines,memory
```

Option	Description
`running`	All running processing engines.
`abandoned`	All abandoned processing engines; an engine is marked as “abandoned” if its sent a command and does not respond in a reasonable period of time. This permits the caller to continue on with life without waiting forever. If the command does eventually complete the engine will be moved to “available”.
`available`	Engines available for new commands.
`problems`	Possible problems: engines which have been running for a long time on a single command, engines whose status is indeterminate.
`engines`	Returns engines of all statuses: *running*, *available*, *abandoned*, and *problems*.
`errors`	A log of the last 1,000 errors.
`connections`	Count of active connections.
`memory`	Current memory utilization. Note that a very low efficiency is not an inefficient application: if memory is returned in a fragmented manner, it can’t be released to the operating system, but it is re-used when needed again.
`config`	Basic configuration parameters used by VOXI.

/image/favicon/
/favicon.ico

Both of these return the favicon graphic, if present, (same data as `org.favicon`) in icon format with no encoding.

/cache/xxx/

The cache as used here is primarily for images embedded within messages and news items. A request for a document here returns that cached document if present.

/downloadfile/xxx

Used for “File download”-type blocks to access a specific file given the file’s 16-digit ID code.

/reportresult/xxx

Used for “My report”-type blocks to access a specific report’s content given the file’s 16-digit ID code.

/image/xyz

Searches for and returns the VSys Live named image with the code or description “xyz”.

VOXI Requests and Results

All VOXI calls and responses are in XML format. These calls are submitted as HTTP `POST` commands.

Requests

VOXI calls include one or more commands, and each command is processed sequentially. The overall call includes a required session identifier which can be text data of any type/length. Each command can optionally include a `SEQ` attribute which, if present, will be included in the response for that command.

Parameters to the VOXI element

Name	Required	Type	Description
`Session`	Optional	Text	
`IP`	Optional	Text	Dotted quad IP address (IP v4 only); if provided, must be the IP address of the user session and is used to enable/disable features on an IP address-by-IP address basis.
`ForwardedIP`	Optional	Text	If user sessions are done through a proxy, this should be the value of the `X-Forwarded-For` HTTP header or any other mechanism for determining the IP address of the actual user. Where by `ForwardedIP` and `IP` are provided, `ForwardedIP` will be used for determining feature enabling.
`URL`	Optional	Text	URL used to access the web application, if applicable; used for enabling/disabling features on a host-by-host basis.
`App`	Optional	Text	Used to indicate the system calling VOXI; intended for enabling access and/or features to different web applications.
`AppVersion`	Optional	Text	Version number in `1.2.3.4` format for `App` above.
`Language`	Optional	Text	Default language code for all commands in this request. Ignored if localization is disabled or the `VSys.bld` file is not present. Using this parameter is preferred to using the command `localization.setlanguage` if all commands in this request will use the same language: VOXI will, if possible, find an existing processing engine already set to this language and re-use it. This saves the overhead of language switching.
`UserAgent`	Optional	Text	`User-Agent` string from the calling browser; used by VOXI to determine if the browser represents a “mobile” device.
`ImageString`	~~Optional~~	~~Text~~	~~Removed in version 0.3.0.0+~~

Sample request:

```
<VOXI session="ABCD123" IP="72.0.154.3" URL="http://volunteers.yourorg.org/" UserAgent="Mozilla/5.0 (Windows NT 5.1;
rv:31.0) Gecko/20100101 Firefox/31.0">
   <REQUEST command="sys.versioninfo” seq=“123”/>
   <REQUEST command="sys.quote”/>
</VOXI>
```

Sample response:

```
<VOXI CommandVersion="0.0.0.4" Version="0.1.0.173">
   <RESULTS>
      <RESULT Command="SYS.VERSIONINFO" Seq="123">
         <VERSION>0.1.0.173</VERSION>
         <NOW>20110618 113640.812</NOW>
         <APPNAME>VSys One eXchange Interface</APPNAME>
         <CACHEABLE Duration="120"/>
         <META Elapsed="6.39649300571364E-5" PerSecond="15633.6"/>
      </RESULT>
      <RESULT Command="SYS.QUOTE" Seq="">
         <QUOTE Speaker="Donald Knuth" Context="Preface to Fundamental Algorithms (1968)" Category="" Code="107">The
         process of preparing programs for a digital computer is especially attractive, not only because it can be
         economically and scientifically rewarding, but also because it can be an aesthetic experience much like
         composing poetry or music.</QUOTE>
         <META Elapsed="0.0187132241267697" PerSecond="53.4"/>
      </RESULT>
      <META Elapsed="0.0196122097441527" PerSecond="51"/>
   </RESULTS>
</VOXI>
```

Both VOXI and REQUEST elements support the optional *`NoCache`* parameter. When set, VOXI will not attempt to process the request from cache; it will still cache the results of the request if appropriate. Example:

```
<VOXI session="ABCD123">
   <REQUEST command="sys.versioninfo" seq="123" NoCache="1"/>
   <REQUEST command="sys.quote"/>
</VOXI>
```

In the above example, `sys.versioninfo` will not be processed from cache.

```
<VOXI session="ABCD123" NoCache="1">
   <REQUEST command="sys.versioninfo" seq="123"/>
   <REQUEST command="sys.quote"/>
</VOXI>
```

In the above example neither `sys.versioninfo` nor `sys.quote` will be processed from cache.

Results

Each *`RESULT`* element's format may vary depending on the command it's associated with. Common attributes and values include:

`CACHEABLE` (element)
If present, means that the provided result may be cached for up to the indicated number of seconds. Caching is done by the caller, not by VOXI itself, and is optional but recommended to reduce the server's workload.

`OK` (element)
If present, indicates that the command was successful. May have additional attributes, may have a *`PROMPT`* element within it.

`ERROR` (element)
If present, indicates that the command failed. The *`Error`* attribute on this element will contain additional information about what went wrong.

Metadata

General

`org.favicon`

Description
Requests the “Header graphic” or “Favicon graphic” values, respectively.

Parameters

Name	Required	Type	Description
`MaxHeight`	Optional	Integer	If provided, requested graphic will be shrunk to fit within the dimensions provided
`MaxWidth`	Optional	Integer	here; if not, it may be shrunk according to system-preferred sizing.
`MaxSize`	Optional	Integer	If provided and the resulting image is greater than this number of bytes in size, VOXI will attempt to re-JPEG encode it at decreasing quality values until either the target size is reached or a lower quality bound of 40 is reached.

Returns
`RESULT` element whose text is the value requested encoded with Base64. (Note that the example below is intentionally truncated for space considerations here.)

```
<RESULT>iVBORw0KGgoAAAANSUhEUgAAACAAAAAgCAYAAABzenr0A…</RESULT>
```

`org.calendar`

See `calendar.list`.

`org.features`

Description
Requests the status of the enabling of various features in VSys One in order to display or hide these features in the web interface.

Parameters

Name	Required	Type	Description
`Entrant`	Optional	Text	Person’s 16-digit code.

Example

```
<REQUEST command="org.features" entrant="EUW38W898JD7YZU5"/>
```

Returns

```
<FEATURES>
   <FEATURE Code="HoursEntry" Enabled="1"/>
   <FEATURE Code="JobSignup" Enabled="1"/>
   <FEATURE Code="JobCheckin" Enabled="1"/>
   <FEATURE Code="NonEmailLogin" Enabled="0"/>
   <FEATURE Code="TrainingSignup" Enabled="1"/>
   <FEATURE Code="NoAccountNameChange" Enabled="1"/>
   <FEATURE Code="InactivityTimeout" Value="45"/>
</FEATURES>
```

The feature `NonEmailLogin`, if enabled, means that a volunteer can sign in by using a VSys One kiosk ID, which is not an e-mail address. Therefore, of this is enabled, at login the web application should not validate the user ID as an e-mail address before passing it to VOXI for validation.

The feature `InactivityTimeout`, if non-zero, indicates in minutes how long a user’s VSys Live session may be inactive before it should be timed out.

`org.files`

Description
Requests the list of available downloadable files.

Parameters

Name	Required	Type	Description
`Entrant`	Optional	Text	Person's 16-digit code.
`FileTypes`	Optional	Text	Comma-delimited list of downloadable file types; files of these types will be returned. Note that since a downloadable file has zero or more types, VOXI does not return the type of each file since those files could have more than one type.

If `Entrant` is provided then only files which are valid for that user will be returned; if it is not provided then only files available to anonymous users are returned.

Example

```
<REQUEST command="org.files" Entrant="EUW38W898JD7YZU5"/>
```

Returns

```
<FILES>
   <FILE Description="January newsletter" Filename="C:\IMG_4122-ss.JPG" Code="EASZ2TWAA99T3FQJ" Size="61266"/>
   <FILE Description="February newsletter" Filename="C:\import specs 1.xml" Code="BGZ6OYW3U1ERQP96" Size="2504"/>
   <FILE Description="About Us" Filename="C:\t1.RTF" Code="PDALYMKKWVZB62U0" Size="47868"/>
   <FILE Description="March 2013 newsletter" Filename="C:\March 2013 newsletter.pdf" Code="0P99SYRBKNBOGET6" Size="3310"/>
</FILES>
```

Download these files from VOXI by using an HTTP GET:

```
        /downloadfile/code
```

Where code is the associated attribute of the *`FILE`* element.

sys.quote

Description

Requests a random quote from the quotes file. the user prompts displayed on various VSys Live pages as defined in the VSys One/GMS 6 configuration. (Note that only a limited subset of these is displayed in the example below.)

Parameters

Name	Required	Type	Description
`Seed`	Optional	Text	Acts as a seed to the randomizer used for selecting a quote. A given seed will always return the same quote for the same version of VOXI.

Returns

One *`QUOTE`* element containing the quote along with the speaker, context, category and unique code. All but "*`Code`*" may be blank.

```
<QUOTE Speaker="Donald Knuth" Context="Preface to Fundamental Algorithms (1968)" Category="" Code="107">The process of
preparing programs for a digital computer is especially attractive, not only because it can be economically and
scientifically rewarding, but also because it can be an aesthetic experience much like composing poetry or music.</QUOTE>
```

System

sys.versioninfo

Description

Requests basic information about VOXI.

Returns

Version, the current date/time as observed by VOXI, and the application's name.

```
<VERSION>0.1.0.173</VERSION>
<NOW>20110618 124553.809</NOW>
<APPNAME>VSys One eXchange Interface</APPNAME>
```

~~sys.commands~~

~~Description~~

~~Requests a list of all commands supported by the current version of VOXI.~~

~~Returns~~

~~One *`COMMAND`* element for each supported command. (Note that the example below is truncated for space considerations.)~~

~~`<COMMAND Code="APP.GETDATA"/>`~~
~~`<COMMAND Code="APP.GETLAYOUT"/>`~~

```
~~<COMMAND Code="APP.GETLAYOUTANDDATA"/>~~
~~<COMMAND Code="APP.GETLAYOUTS"/>~~
~~<COMMAND Code="APP.POSTDATA"/>~~
```

~~sys.commandsupported~~

~~Description~~
~~Checks the validity of a given command. Not normally used.~~

~~Parameters~~

~~Name~~	~~Required~~	~~Type~~	~~Description~~
~~Query~~	~~Required~~	~~Text~~	~~Name of the command to be checked.~~

~~Example~~
```
~~<REQUEST command="sys.commandsupported" query="sys.commands"/>~~
```

~~Returns~~
```
~~<SUPPORTED Value="1"/>~~
```

sys.optionsupported

Description
Not implemented.

Options

sys.allfieldoptions, sys.contactflags, sys.genders, sys.groups, sys.jobpreferences, sys.languages, sys.partnertypes, sys.peopletypes, sys.skills, sys.specialneeds, sys.sports, sys.volunteersources, sys.volunteertypes, sys.addresstypes, sys.phonetypes, sys.partnertypes, sys.partnerstatuses, sys.partnerentrystatuses, sys.partnerentryroles, sys.weblocations, sys.assignmentstatuses, sys.interviewtypes, sys.slotstatuses, sys.restrictions, sys.webroles, sys.newsapprovalstatuses, sys.calendarapprovalstatuses, sys.attachmenttypes, sys.newstags

Description
All of these return a list of the valid options of the given type. `sys.allfieldoptions` returns all options with a single call and is preferred for performance reasons if more than one option type list is needed.

Returns
One *`OPTIONS`* element for each class plus an *`OPTION`* detail element for every possible value for that class. Individual *`OPTION`* elements may return *`DETAILS`* and/or *`COMMENTS`* elements. The *`DETAILS`* elements can optionally be used for displaying additional information about an option to the end user; *`COMMENTS`* elements are generally for administrative purposes.

```
<OPTIONS Class="SKILLS">
   <OPTION Code="PC7SQEIVEJ08RNDZ" Name="Administration/Office">
      <DETAILS><![CDATA[Admin users]]></DETAILS>
      <COMMENTS><![CDATA[*** some comments ***]]></COMMENTS>
   </OPTION>
   <OPTION Code="DCFB5G003DVHW3IO" Name="Artist">
      <DETAILS><![CDATA[Artsy-fartsy!]]></DETAILS>
   </OPTION>
</OPTIONS>
```

sys.checkemailblacklist

Description
Checks one or more provided e-mail addresses against the list of system-maintained blacklisted e-mails.

Parameters
(none)

Example
```
<REQUEST command="sys.checkemailblacklist">
<EMAIL>bogus@baddomain.com</EMAIL>
```

```
<EMAIL>thisone@okay.com</EMAIL>
<EMAIL>marked_us_as_spam@yahoo.com</EMAIL>
</REQUEST>
```

Returns

An *OK* element followed by zero or more *BADEMAIL* elements.

```
<OK/>
<BADEMAIL>bogus@baddomain.com</BADEMAIL>
<BADEMAIL>marked_us_as_spam@yahoo.com</BADEMAIL
```

Logging/tracing

VOXI internally posts logging records to the “`voxilogs`” table; these logs include most requests and their results. Records posted to `voxilogs` can be searched and analyzed directly from the VSys One desktop application. External tools can post their own records here for debugging purposes.

`trace.post`

Description
Posts a single record to the `voxilogs` table.

Parameters

Name	Required	Type	Description
`Desc`	Required	Text	Basic description of what is being logged; value here will be truncated to 32 characters.
`Type`	Required	Text	`D` = debugging `E` = error `X` = logging
`Duration`	Optional	Duration	How long the command took to execute.
`Details`	Optional	Text	Unlimited length text information.

Example
```
<REQUEST command="trace.post" type="E" desc="page parsing" dur="00:00:01.239" details="page xyz requested a bogus block"/>
```

Returns
An *`OK`* or *`ERROR`* element.

VSys Live

Certain VOXI functions are specific to VSys Live and the corresponding setup tools within VSys One.

VSys Live pseudopages
VOXI will automatically generate pages for certain specially-crafted page codes. It is not necessary for VSys Live to understand how or when this is done: these pages are handled by VSys Live just as any other page is when returned by VOXI in response to `vlive.page` with one exception: they will never appear in the results of a call to `vlive.pages`.

This means that VSys Live must not call `vlive.pages` to validate the page code before attempting to display that page.

One example of a pseudopage is `app:FAQZFNO0C2ZHAFBG`. Calling

```
<REQUEST command="vlive.page" state="auth" page="app:FAQZFNO0C2ZHAFBG"/>
```

will result in VOXI returning a page which places an application input block, configured to use application "`FAQZFNO0C2ZHAFBG`", in the placeholder region of that page as defined in the page's properties. This page must be a "placeholder" page, and it must be defined for the current VSys Live site as the "Applications display page".

vlive.pages

Description
Returns a list of valid VSys Live pages along with the default root pages for both the logged in (auth) and not logged in states, the 404 page and the permalink handler page with region.

Parameters
(none)

Example

```
<REQUEST command="vlive.pages"/>
```

Returns

```
<PAGES>
   <SPECIALPAGE Use="Root" Code="E4KF8IG0892GLNWT"/>
   <SPECIALPAGE Use="RootAuth" Code="E4KF8IG0892GLNWT"/>
   <SPECIALPAGE Use="404" Code="N2RQ5DOK2M51QTVG"/>
   <SPECIALPAGE Use="Permalinks" Code="NK711KNCXKU8WUJN" Region="mbody"/>
   <PAGE Code="8R4X9ROLXV92MGHT" AuthMode="auth" Description="Page 3" Friendly="P3"/>
   <PAGE Code="E4KF8IG0892GLNWT" AuthMode="*" Description="Another page" Friendly=""/>
   <PAGE Code="N2RQ5DOK2M51QTVG" AuthMode="*" Description="404 page" Friendly="404"/>
   <KIOSKPAGE Use="Login" Code="O45Y8FO4UI5GUIHS"/>
   <KIOSKPAGE Use="Messages" Code="TP89GHOHGUI4OIWW"/>
   <KIOSKPAGE Use="Checkin" Code="3EJKLHFOY57OTRGT"/>
   <KIOSKPAGE Use="Checkout" Code="45UIOG8O45YT78YH"/>
   <KIOSKPAGE Use="Agreements" Code="G1EHN0UBCY2ERNJ7"/>
</PAGES>
```

The attribute *`Friendly`*, if present, should be used as the URL within a browser.

vlive.page

Description
Returns a specific VSys Live page along with the contents of each region and any configuration properties.

When calling `vlive.page`, VSys Live should include in the HTTP headers of the request to VOXI those cookies present in the client's request to VSys Live. While the specific actions taken by VOXI based on those cookies is not defined, example behavior may be to vary content based on cookies or settings previously set by *`COOKIESET`*.

Parameters

Name	Required	Type	Description
`Page`	Required	Text	Identifier of the page as provided in `vlive.pages`.
`State`	Required	Text	`auth` – A user is validly logged in. `unauth` – no user is logged in.
`User`	Optional	Text	16-digit ID of the currently logged-in user.

Example

```
<REQUEST command="vlive.page" page="8R4X9ROLXV92MGHT" user="EUW38W898JD7YZU5" state="auth"/>
```

Returns

If the attribute *SpecialResultCode* is included as part of the *PAGE* element, that value must be returned as the HTTP status of the result. This will normally be used to indicate 404 pages.

- Note that zero, one or more *JS* and *CSS* and *HEADER* elements may be returned. If present they must be included in the resulting HTML output in the order in which they are returned by VOXI.
- A single *HEADER* element, if present, should be placed in the `<HEAD>` section of the HTML output, but should not be wrapped in any other HTML elements; i.e. this text should be placed raw within the HTML `<HEAD>` output.
- Zero or more *METADATA* elements may be returned; if present they must all be included in the page's HTML metadata.
- *BLOCK* and *LISTBLOCK* elements may be included in regions (see below).
- See the section "`vlive.page` post-processing" for information about `VOXI://` text within the returned content.
- Zero or more *COOKIESET* elements may be present. For each, VSys Live must set the corresponding cookie when returning the generated page to the calling browser.
- Zero or one *REDIRECT* elements may be present. If one exists, VOXI will usually return valid page content, but rather than rendering the page, the browser should redirect the browser to the designated location. Note that it is important for any *COOKIESET* elements to be handled even if the page is not being displayed.

Example page results

```
<PAGE Title="Page name" SpecialResultCode="404">
   <JS>function ()&#xD;&#xA;  { }&#xD;&#xA;&#xD;&#xA;</JS>
   D:\bespoke\VSys\VSys Web\2.1<CSS>
   <![CDATA[
   img {
       max-width: none;
   }
   ]]>
   </CSS>

   <CSS>&#xD;&#xA;</CSS>

   <METADATA>
   <![CDATA[VSys Live by Bespoke Software, a member of the VSys One family of volunteer management solution tools]]>
   </METADATA>
   <METADATA>Put some information about your organization here. It won't be visible to site&#xD;&#xA;visitors but it will
   appear in each page's metadata and be seen by search&#xD;&#xA;engines.&#xD;&#xA;</METADATA>

   <HEADER><![CDATA[<link href="/download/spirelle2.css" media="all" rel="stylesheet" type="text/css" />
   ]]></HEADER>

   <COOKIESET Code="SomeVSysLiveCookie">ABC123ThisIsWhereWeStoreTheData!</COOKIESET>

   <REDIRECT>https://demo.vsyslive.com</REDIRECT>

   <CONFIGURATION>
      <SETTING Name="template.header" Value="Top!"/>
      <SETTING Name="template.orgname" Value="Our org."/>
      <SETTING Name="template.date" Value="2013/03/17 00:00:00.000"/>
      <SETTING Name="template.time" Value="14:45:00.000"/>
      <SETTING Name="template.check" Value="1"/>
      <SETTING Name="template.memo"
      Value="Hi.&#xD;&#xA;You.&#xD;&#xA;&#xD;&#xA;Ugly.&#xD;&#xA;&#xD;&#xA;&#xD;&#xA;Bastard.&#xD;&#xA;"/>
      <SETTING Name="template.float" Value="3.14159"/>
      <SETTING Name="template.integer" Value="4"/>
      <SETTING Name="template.image" Value="VOXI://RNSLFP737V2HFW42"/>
      <SETTING Name="template.combo" Value="d"/>
      <SETTING Name="template.checklist" Value="a,e"/>
   </CONFIGURATION>
   <REGION Code="topleft" Name="Top left">
      <BLOCK Code="7SCF23A1QL8PUOCH" Source="VOXI" Name="Static #2"/>
      <BLOCK Code="CDURK6EF8VVPNSP3" Source="VOXI" Image="RNSLFP737V2HFW42"/>
      <BLOCK Code="399KPON1QO3ZB077" Source="VOXI" Image="5GXVU3OUAUCX5NUA"/>
      <MENU>
         <ITEM Code="DXKCMFFTZ843AG81" Caption="Huh?" Link="VLIVE://sendmessage/CN5YGFO34RWEOFUY">
         </ITEM>
         <ITEM Code="JKLOIELUM4GHBPUL" Caption="Apply!" Link="VLIVE://app/TALKAJ4EQ5NDIG2E">
         </ITEM>
         <ITEM Code="U55I5GDNGRIUDGFP" Caption="Whine, whine, whine" Link="VLIVE://sendmessage/WERPYF78Y23O3FYQ">
         </ITEM>
         <ITEM Code="DYBUP0JZRYDT48YD" Caption="App" Link="VLIVE://app/ZMM9I1LU7MUNINHL">
            <ITEM Code="JKLOIELUM4GHBPUL" Caption="Apply!" Link="VLIVE://app/TALKAJ4EQ5NDIG2E">
            </ITEM>
         </ITEM>
         <ITEM Code="DZ6GPJ8N92Y5PISL" Caption="Log me out!" Action="logout">
         </ITEM>
         <ITEM Code="0PLCI4G6QWSDD1A2" Caption="Page 3" Link="VLIVE://page/8R4X9ROLXV92MGHT">
         </ITEM>
         <ITEM Code="D2GPJV091QWCKTSF" Caption="2b(1)" Link="http://www.vsysone.com">
         </ITEM>
         <ITEM Code="NNA5SADNEE8RDFDU" Caption="Item #1" Link="VLIVE://block/signin" Rel="1">
         </ITEM>
         <ITEM Code="WINETNS5VSYPM2RI" Caption="Item #2" Link="">
            <ITEM Code="DYBUP0JZRYDT48YD" Caption="App" Link="VLIVE://app/ZMM9I1LU7MUNINHL">
               <ITEM Code="JKLOIELUM4GHBPUL" Caption="Apply!" Link="VLIVE://app/TALKAJ4EQ5NDIG2E">
               </ITEM>
            </ITEM>
            <ITEM Code="DZ6GPJ8N92Y5PISL" Caption="Log me out!" Action="logout">
            </ITEM>
            <ITEM Code="0PLCI4G6QWSDD1A2" Caption="Page 3" Link="VLIVE://page/8R4X9ROLXV92MGHT">
            </ITEM>
         </ITEM>
         <ITEM Code="AIQ0PYKEZJ5J7S7O" Caption="Talk to Us">
            <ITEM Code="DXKCMFFTZ843AG81" Caption="Huh?" Link="VLIVE://sendmessage/CN5YGFO34RWEOFUY">
            </ITEM>
            <ITEM Code="U55I5GDNGRIUDGFP" Caption="Whine, whine, whine" Link="VLIVE://sendmessage/WERPYF78Y23O3FYQ">
            </ITEM>
         </ITEM>
      </MENU>
   </REGION>
   <REGION Code="topright" Name="Top right">
      <BLOCK Code="3GP9VYSH451VSVV2" Source="web" Name="Assignment listings">
      </BLOCK>
      <BLOCK Code="GHGJNAS7C4Z3XGU4" Source="VOXI" Name="Static Block #1"/>
      <BLOCK Code="BR0XA3EFR1XWFY4Z" Source="web" Name="Assignment listings">
         <CONFIGURATION>
            <SETTING Name="assignments.pagesize" Value="100"/>
            <SETTING Name="assignments.mode" Value="c"/>
         </CONFIGURATION>
      </BLOCK>
   </REGION>
   <REGION Code="mmenu" Name="Main menu region">
   </REGION>
   <REGION Code="mbody" Name="Main body">
      <BLOCK Code="NQAMUBZ91U2UGKJ4" Source="VOXI" Name="Static Block #1"/>
   </REGION>
   <REGION Code="footleft" Name="Footer left">
   </REGION>
   <REGION Code="footmid" Name="Footer middle">
   </REGION>
   <REGION Code="footright" Name="Footer right">
   </REGION>
</PAGE>
```

Example *BLOCK* – inline content

BLOCK elements with `Inline="1"` represent raw HTML which should be rendered directly to the browser; other *BLOCK* elements require calls to `vlive.staticblock`.

```
<BLOCK Code="XDPH41BBIKSCE1WB" Caption="" Source="VOXI" Name="Hours" Inline="1">
   <BLOCK>
      <![CDATA[<DIV class="block-container my-info-trainings-block-container" style="">
         <DIV class="block" style="">
            <HEADER class="block-header" style=""><H4>Hours</H4></HEADER>
            <TABLE class="table table-striped" style="">
               <TR class="" style="">
               <TH>Project</TH>
```

```
                <TH>Date</TH>
                …
      ]]>
   </BLOCK>
</BLOCK>
```

Example *`BLOCK`* – Asynchronous loading, VSys Live content

```
<BLOCK Code="XDPH41BBIKSCE1WC" Caption="" Source="WebAsynch" Name="Some complex listing"/>
```

`BLOCK` elements with `Source="WebAsynch"` represent blocks whose contents should be retrieved via `vlive.staticblock` but that loading must be done asynchronously, i.e. not delay the generation of the page provided to the browser. Handling blocks of this type will generally involve:

1. Rendering this portion of the page as a content stub with a Javascript callback,
2. That callback requests the content from VSys Live, and
3. VSys Live requests that content from VOXI,
4. Content pushed into page when available.

These will generally be used in cases where the block's contents are computationally expensive.

Example *`BLOCK`* – Asynchronous loading, external content

```
<BLOCK Code="XDPH41BBIKSCE1WD" Caption="" Source="ExtAsynch" Name="Some complex graph" URL="/cache/dashboard/ABC123"/>
```

`BLOCK` elements with `Source="VOXIAsynch"` represent blocks whose contents should be retrieved via a request to a URL, with that URL not necessarily being handled by VSys Live. That loading must be done asynchronously, i.e. not delay the generation of the page provided to the browser. Handling blocks of this type will generally involve:

1. Rendering this portion of the page as a content stub with a Javascript callback,
2. That callback requests the content from the external URL, and
3. Content pushed into page when available.

These will generally be used in cases where the block's contents are computationally expensive. For performance reasons, these will often bypass VSys Live and make calls to VOXI through its available paths.

Example *`LISTBLOCK`*

`LISTBLOCK` elements contain structured data which should be rendered in an appropriate container with sorting and pagination controls. Possible values for *`Type`* (attributes of a column) are "`string`", "`integer`", "`float`", "`date`", "`time`", "`datetime`" and "`bool`".

```
<LISTBLOCK Code="W8627MBXSINFNLEE" Caption="" Name="Expiring attributes" CountPerPage="0">
   <FILTERFIELDS>
      <FILTERFIELD Code="MinDate" Type="D" Description="Earliest date"/>
      <FILTERFIELD Code="MaxDate" Type="D" Description="Last date"/>
      <FILTERFIELD Code="type" Type="C" Description="Type">
         <OPTIONS>
            <OPTION Code="O" Value="Open"/>
            <OPTION Code="X" Value="Cancelled"/>
            <OPTION Code="R" Value="Rejected"/>
         </OPTIONS>
      </FILTERFIELD>
   </FILTERFIELDS>

   <COLUMNS>
      <COLUMN Code="kind" Name="Type" Type="string"/>
      <COLUMN Code="exp" Name="Expiration date" Type="date"/>
   </COLUMNS>
   <DATA>
      <RECORD Sort="20090813 000000.000UNE4X4FG78TXF20X">
         <VALUE Code="kind">First Aid</VALUE>
         <VALUE Code="exp">08/13/2009</VALUE>
      </RECORD>
      <RECORD Sort="20130425 000000.000BPDL9JQ4RW8Q23JT">
         <VALUE Code="kind">Switchboard</VALUE>
         <VALUE Code="exp">04/25/2013</VALUE>
      </RECORD>
      <RECORD Sort="20130910 000000.00016D4X7C02X650XFP">
         <VALUE Code="kind">Hospice</VALUE>
         <VALUE Code="exp">09/10/2013</VALUE>
      </RECORD>
      <RECORD Sort="20150720 000000.000E05LMGYNB0V0GHHE">
         <VALUE Code="kind">RN</VALUE>
```

```
        <VALUE Code="exp">07/20/2015</VALUE>
      </RECORD>
    </DATA>
</LISTBLOCK>
```

FILTERFIELDS handling

LISTBLOCK elements returned by `vlive.page` can support end-user filtering. A *LISTBLOCK* may include zero or more *FILTERFIELDS\FILTERFIELD* elements (as shown in the example above), each of which represents a filter that should be displayed to the user.

For *LISTBLOCK* elements, `vlive.page` has already been called and the full page's content has been returned and probably rendered to the user. Rather than re-calling `vlive.page` (slow; forces a page reload), call `vlive.page.block` (see below) with the same parameters already sent with `vlive.page`, plus extras and dynamically replace that element in the HTML DOM. See the VOXI method `vlive.page.block` below.

MENU items

Menu items can be any of several different types.

Type	*Link*
External URL	An HTTP or HTTPS URL.
Page	`VLIVE://page/pageID`
Action	Blank; *Action* is the action to be taken.
Send a message	`VLIVE://sendmessage/messageType`
Placeholder	Blank
Report	`VLIVE://report/reportID/email` (if `email` is provided, this should be used as the default destination e-mail address for the delivered report)

PAGETEMPLATE elements

In the default configuration, VSys Live knows the regions available to it and has its own built-in template for actually generating the HTML that comprises the page returned to the end user browser. VOXI is permitted to return a *PAGETEMPLATE* element which overrides that built-in template.

The details on how that template is to be formatted and understood are implementation-dependent. In the example case of a default Rails-based implementation of VSys Live, the template comes from the file `template.html.haml`. In this case the content returned by VOXI would replace the pre-existing *%body.vsyslive* region of that template through the *#vsys-modal-spinner.vsys-spinner.modal.hide* element. (This is based on the built-in template as of the time of this writing; again, this is very implementation-specific.) For example, something like the following (which here is identical to the current version of the built-in template) would be returned.

```
<PAGETEMPLATE>
<![CDATA[
  %body.vsyslive
    - if content_for? :page_top_js
      = yield :page_top_js

    - if content_for?(:supertop_panel)
      #supertop-bar.navbar.navbar-top.navbar-inverse
        .navbar-inner
          .container
            = yield :supertop_panel

    - if content_for?(:top_panel) or content_for?(:top_panel_right)
      #top-bar.navbar.navbar-top.navbar-inverse
        .navbar-inner
          .container
            - if content_for?(:top_panel)
              = yield :top_panel
            - if content_for?(:top_panel_right)
              .pull-right
                = yield :top_panel_right
]]>
</PAGETEMPLATE>
```

When this is done, VOXI is expected to:

1. Provide valid HAML content,

2. Return page *REGION\BLOCK\* content that corresponds to the `= yield :region_name elements` in the HAML,
3. Provide appropriate CSS for the expected page.

vlive.staticblock

Description

Static HTML blocks are chunks of HTML defined by the end-user within VSys One to be displayed directly within a VSys Live region. This command returns a given static HTML block for the given page.

Parameters

Name	Required	Type	Description
`Page`	Required	Text	Code of the page on which this block is being rendered.
`State`	Required	Text	`auth` – A user is validly logged in. `unauth` – no user is logged in.
`Id`	Required	Text	ID of the block as provided in `vlive.page`.
`User`	Optional	Text	16-digit ID of the currently logged-in user.

Example

```
<REQUEST command="vlive.staticblock" state="auth" page="8R4X9ROLXV92MGHT" id="GHGJNAS7C4Z3XGU4" user="EUW38W898JD7YZU5"/>
```

Returns

Example:

```
<BLOCK>
   <![CDATA[<div style="width: 100%;">
   <h3 style="text-align: center;">Case Studies</h3>
   <table style="width: 100%;">
   <tr>
   <td style="width: 33%; min-width: 200px; vertical-align: top;">
     <a href="/calgary-food-bank"> Food Bank<br>img src="VOXI://N5PQUIRMEUAKK7CG">a>
   </td>
   </tr>
   </table>
   </div>
   ]]>
</BLOCK>
```

Example:

```
<BLOCK>
   <![CDATA[<a href="/"><img src="VOXI://6HFXKCT3K69JR0JX" alt="VSys One by Bespoke Software"></a>]]>
</BLOCK>
```

vlive.page.block

Description

This call is designed to allow the dynamic re-calculation of a single block already returned by `vlive.page` and rendered to the user. It can be used for refreshing existing content, or to apply filters to a *`LISTBLOCK`*.

Returns a single list view block from the designated page.

Parameters

Name	Required	Type	Description
`Page`	Required	Text	Code of the page on which this block is being rendered.
`State`	Required	Text	`auth` – A user is validly logged in. `unauth` – no user is logged in.
`Id`	Required	Text	ID of the block as provided in `vlive.page`.
`User`	Optional	Text	16-digit ID of the currently logged-in user.
(filters)	Optional	(varies)	Note: Checklist field results should be sent as a comma-delimited list of selected values.

Example

```
<REQUEST command="vlive.page.block" page="home" user="HIIJ779YTPZD5ZCH" state="auth" id="9O20CE1M3LBB9QIR">
   <FILTER code="MinDate" value="2012-01-01"/>
   <FILTER code="Status" value="X,R"/>
</REQUEST>
```

Returns

```
<BLOCK Code="9O20CE1M3LBB9QIR" Caption="" Source="VOXI" Name="Trainings" Inline="1">
   <BLOCK>
      <![CDATA[<DIV class="block-container my-info-trainings-block-container" style="">
      <DIV class="block" style="">
      <HEADER class="block-header" style=""><H4>Trainings</H4></HEADER>
      <TABLE class="table table-striped" style="">
      <TR class="" style="">
      <TH>Type</TH>
      <TH>Status</TH>
      <TH>Date</TH>
      <TH>Expiration date</TH>
      </TR>
      <TR class="" style="">
      <TD>Webinars</TD>
      <TD>Registered</TD>
      <TD>01/24/2014 01:30PM</TD>
      <TD></TD>
      </TR>
      </TABLE>
      </DIV>
      </DIV>
      ]]>
   </BLOCK>
</BLOCK>
```

vlive.popupblock

Description
Retrieves the content to be displayed in a custom popup window.

Parameters

Name	Required	Type	Description
Page	Required	Text	Code of the page from which the popup is called.
State	Required	Text	auth – A user is validly logged in. unauth – no user is logged in.
URL	Required	Text	The URL clicked, exclusive of the /custompopup/ prefix.
User	Optional	Text	16-digit ID of the currently logged-in user.

Example

```
<REQUEST command="vlive.popupblock" state="auth" page="8R4X9ROLXV92MGHT" URL="ABC12356/whatever/morethings/optionalthings"
user="EUW38W898JD7YZU5"/>
```

Returns
Example:

```
<BLOCK ButtonCaption="Close" HTMLStyle="min-width: 600px;" HTMLClass="some-popup-window" Title="This is the caption for the
popup">
   <CONTENT>
   <![CDATA[<h3>INFORMATION</h3><p>We want information.</p><p>And we're going to get it, number six.</p>]]>
   </CONTENT>
</BLOCK>
```

vlive.page post-processing

Content returned by VOXI requires additional processing before returning to the browser. The specialized content is signified with `VOXI://`.

Note that some of these URLs may be pushed into the page's content through channels other than `vlive.page` and other standard sources. For example, VSys Live custom calendars rendered using FullCalendar may be populated asynchronously or via JavaScript. This may require specialized JavaScript on the page to handle the end-user clicking of these URLS rather than just parsing them out of the initial page content.

VSys Live "actions"

(Only within an HTML `HREF`.)

Takes the action "logout" as specified in the VSys Live metadata.

```
<a href="VLIVE://action/logout">Log out</a>
```

Send Message

(Only within an HTML `HREF`.)

Must popu up the appropriate "Send message" dialog box when clicked on.

```
<a href="VLIVE://sendmessage/messageType">Talk to us!</a>
```

(This is mostly deprecated in favor of using VOXI-defined virtual applications for sending messages.)

Custom Popups

(Only within an HTML `HREF`.)

Opens a popup window with contents retrieved from VOXI.

```
<a href="VLIVE://custompopup/ABC12356/whatever/morethings/optionalthings">Details</a>
```

The content to be displayed must be retrieved from VOXI using `vlive.popupblock`. Results of that will include:

- An optional *`Title`* attribute to be used as the caption for the popup window,
- An optional caption for the button as *`ButtonCaption`* (the displayed button will close the window),
- Arbitrary HTML in the *`CONTENT`* element,
- Optional *`HTMLClass`* and *`HTMLStyle`* attributes which should be appended to any default *`class`* and *`style`* attributes assigned to the created popup window element.

Sign Up for a Job

(Only within an HTML `HREF`.)

Opens the job slots detail popup window.

```
<a href="VLIVE://jobsignup/context/job/slot/mergetemplate/signup">More information</a>
```
or
```
<a href="VLIVE://jobsignup/context/job/slot/mergetemplate/nosignup">More information</a>
```

These are the same windows used in the job signup tool to give additional information about a job, and which usually include a [Sign up] button. In this case VSys Live will use the specified merge template code to show details about the given job slot by calling `jobs.getSlotDetails`.

If the link ends in `/signup/`, the [Sign up] button should be displayed, otherwise it should be suppressed. This is intended to allow display of additional job information in cases where the current user is not eligible to actually select that job opening.

General content

(Anywhere in VOXI-generated text.)

Other `VOXI://` prefixed text in these blocks refer to images which must be accessed by using an HTTP `GET` to VOXI. Since VOXI itself should not be exposed to the public Internet, this means replacing the `VOXI://xyz` with your own publicly-accessible handler which then calls VOXI on `/cache/xyz` to return the result to the end user.

Example

```
	<IMG src="VOXI://ABCDEF0123456789"/>
```

should be replaced with

```
	<IMG src="http://whateversite.com/cache/ABCDEF0123456789"/>
```

which in turn would then be handled by Apache, forwarding the request to VOXI, which through Apache, returns the requested image's content.

(This is mostly deprecated in favor of using URLs that specifically point to VOXI-handled paths, e.g. `/image/something`.)

VSys Live Volunteer Request Portal

The VSys Live Volunteer Request Portal is a mechanism integrated into the existing VSys One/VSys Live architecture designed to permit non-VSys One users to, using the VSys Live infrastructure:

- Submit requests for volunteers for their department, organization, etc.
- Edit and cancel open requests
- View (but not edit) volunteer assignments and slots associated with this non-VSys One user.

VSys One users (administrators) will approve, edit, or reject those requests. On approval, job openings will be posted to VSys Live for actual volunteers to sign up for just as any other job slot would be.

Browser implementation
While VOXI does not specify that this feature must be implemented in a specific way, it does provide support for automatic menu items, list views and pseudo-pages for the portal. Menu items and list views will generate URLs like `/pages/volreq:layoutcode:existingcode`. These represent pseudo-pages, and generating those pages does require a specific VSys Live block with specific attributes:

- The block's code must be "`volreqapp`".

It must have two specific configurable properties:

- "Request layout" having the code "`volreqapp.layout`", of type "`volreqapp`".
- "Request existing application" having the code "`volreqapp.existing`", of type "`string`".

When generating the pseudo-page via `vlive.page`, VOXI will use the pre-defined template page and place the block into the appropriate region, configuring the block as applicable to the URL, then pass it back to the caller just as any other page.

vrequest.types.allowcreate

Description
Returns a list of valid request types that the current user can access.

Parameters

Name	Required	Type	Description
`User`	Required	Text	16-digit ID of the currently logged-in user.

Example
```
<REQUEST command="vrequest.types.allowcreate"/>
```

Returns
```
<LAYOUTS>
   <LAYOUT Code="E9L2Y1Q4QXAM409C" Context="0000700000000000" Description="Regular (scheduled) volunteers"/>
   <LAYOUT Code="M907XB2GN8CLHLWG" Context="0000700000000000" Description="Flex volunteer"/>
   <LAYOUT Code="NNUAXHIT5SOGW9WE" Context="0000700000000000" Description="Intern request"/>
</LAYOUTS>
```

vrequest.type.getlayout

Description
Returns the layout for a single volunteer request application. Note that all volunteer request application layouts are "Dynamic", meaning that they come with no field positioning information and should placed on the page in the order given.

Example
```
<REQUEST command="vrequest.type.getlayout" layout="E9L2Y1Q4QXAM409C"/>
```

Returns
```
<LAYOUT Code="E9L2Y1Q4QXAM409C" Description="Regular (scheduled) volunteers" DynamicLayout="1">
   <LAYOUTFIELDS>
      <LAYOUTFIELD Code="vrapp.flex" Description="Flex" Type="B" Required="0" Hint="" LineBreak="0" CorrectAnswer=""
      CorrectAnswerMessage="" Password="0" IncludeBlankOptionOnWeb="0" ReadOnly="0">
      </LAYOUTFIELD>
```

```
        <LAYOUTFIELD Code="job.availnrq" Description="Availability not required" Type="B" Required="0" Hint="" LineBreak="0"
        CorrectAnswer="" CorrectAnswerMessage="" Password="0" IncludeBlankOptionOnWeb="0" ReadOnly="0">
           <POSITION Width="250" Height="0"/>
        </LAYOUTFIELD>

        <LAYOUTFIELD Code="job.userRTF.1" Description="User text 1" Type="M" Required="0" Hint="" LineBreak="0"
        CorrectAnswer="" CorrectAnswerMessage="" Password="0" IncludeBlankOptionOnWeb="0" ReadOnly="0">
           <LABEL Caption="User text 1:" Visible="1" Alignment="L"/>
        </LAYOUTFIELD>

        <LAYOUTFIELD Code="job.skills" Description="Required skills" Type="C" Required="1" Hint="" LineBreak="0"
        CorrectAnswer="" CorrectAnswerMessage="" Password="0" IncludeBlankOptionOnWeb="0" ReadOnly="0">
           <POSITION Width="500" Height="0"/>
           <OPTIONS>
              <OPTION Code="QXQMY8DVERA07BQU" Value="Athlete"/>
              <OPTION Code="VPFA56JKZ8KKOSEH" Value="Audio/Visual"/>
              <OPTION Code="DZ38VLRLCZEDHRST" Value="Transportation - Driver (bus/van)"/>
              <OPTION Code="GQT0BDESOVM4NR3N" Value="Transportation - Driver (car)"/>
              <OPTION Code="NJ2ESU6YBU2HPHGS" Value="Transportation - Emergency"/>
              <OPTION Code="U48JLK0D3WD7ZDWH" Value="Transportation - Moving"/>
              <OPTION Code="CS9V0ZSDOM7NG9QD" Value="Transportation - Parking"/>
              <OPTION Code="9SZUAT3WCH4PQ9EJ" Value="Video Production"/>
              <OPTION Code="R22HJT75IRS24H2L" Value="Writing/Reporting"/>
           </OPTIONS>
           <LABEL Caption="Required skills:" Visible="1" Alignment="L"/>
        </LAYOUTFIELD>

        <LAYOUTFIELD Code="job.duties" Description="Duties" Type="M" Required="0" Hint="" LineBreak="0" CorrectAnswer=""
        CorrectAnswerMessage="" Password="0" IncludeBlankOptionOnWeb="0" ReadOnly="0">
           <LABEL Caption="Duties:" Visible="1" Alignment="L"/>
        </LAYOUTFIELD>

        <LAYOUTFIELD Code="vrapp.dow" Description="Days of week" Type="C" Required="0" Hint="" LineBreak="0"
        CorrectAnswer="" CorrectAnswerMessage="" Password="0" IncludeBlankOptionOnWeb="0" ReadOnly="0">
           <OPTIONS>
              <OPTION Code="0" Value="Sunday"/>
              <OPTION Code="1" Value="Monday"/>
              <OPTION Code="2" Value="Tuesday"/>
              <OPTION Code="3" Value="Wednesday"/>
              <OPTION Code="4" Value="Thursday"/>
              <OPTION Code="5" Value="Friday"/>
              <OPTION Code="6" Value="Saturday"/>
           </OPTIONS>
           <LABEL Caption="Days of week:" Visible="1" Alignment="L"/>
        </LAYOUTFIELD>

        <LAYOUTFIELD Code="cfield.8E2BWUUVSQZHWQLA" Description="Alpha" Type="F" Required="0" Hint="" LineBreak="0"
        CorrectAnswer="" CorrectAnswerMessage="" Password="0" IncludeBlankOptionOnWeb="0" ReadOnly="0">
           <CONSTRAINTS MinValue="5" MaxValue="12"/>
           <LABEL Caption="Alpha:" Visible="1" Alignment="L"/>
        </LAYOUTFIELD>

        <LAYOUTFIELD Code="cfield.O34UIYF789TOH30H" Description="Comments" Type="M" Required="0" Hint="" LineBreak="0"
        CorrectAnswer="" CorrectAnswerMessage="" Password="0" IncludeBlankOptionOnWeb="0" ReadOnly="0">
           <LABEL Caption="Comments:" Visible="1" Alignment="L"/>
        </LAYOUTFIELD>

        <LAYOUTFIELD Code="vrapp.date.first" Description="First date" Type="D" Required="0" Hint="" LineBreak="0"
        CorrectAnswer="" CorrectAnswerMessage="" Password="0" IncludeBlankOptionOnWeb="0" ReadOnly="0">
           <LABEL Caption="First date:" Visible="1" Alignment="L"/>
        </LAYOUTFIELD>

        <LAYOUTFIELD Code="vrapp.date.last" Description="Last date" Type="D" Required="0" Hint="" LineBreak="0"
        CorrectAnswer="" CorrectAnswerMessage="" Password="0" IncludeBlankOptionOnWeb="0" ReadOnly="0">
           <LABEL Caption="Last date:" Visible="1" Alignment="L"/>
        </LAYOUTFIELD>
    </LAYOUTFIELDS>
</LAYOUT>
```

vrequest.get

Description

Returns the properties of a single pre-existing submitted request.

Parameters

Name	Required	Type	Description
User	Required	Text	16-digit ID of the currently logged-in user.
Code	Required	Text	Code for pre-existing request.

Example

```
<REQUEST command="vrequest.get" user="VOLUNTEERREQAPP0" code="C7TOJXZ5CEBE07C1UK281ZO8LJ2U0GIL"/>
```

Returns

```
<DATA Code="0000700000000000" LayoutID="J02KWVML1KTRMKDP" Layout="Regular (scheduled) volunteers">
   <FIELD Code="vrapp.flex">True</FIELD>
   <FIELD Code="job.availnrq">True</FIELD>
   <FIELD Code="job.userRTF.1">Some text. It's nice.</FIELD>
   <FIELD Code="job.skills">1BF588R7IC3HV5IZ,546ZL0FOJT5L92LC,QXQMY8DVERA07BQU</FIELD>
   <FIELD Code="job.duties">Do work.</FIELD>
   <FIELD Code="cfield.8E2BWUUVSQZHWQLA">0</FIELD>
   <FIELD Code="cfield.O34UIYF789TOH30H"></FIELD>
</DATA>
```

vrequest.post

Description

Creates a new volunteer request, or updates an existing one.

Parameters

Name	Required	Type	Description
User	Required	Text	16-digit ID of the currently logged-in user.
Layout	Required	Text	16-digit ID of the layout.

Example

```
<REQUEST command="vrequest.post" user="VOLUNTEERREQAPP0" layout="E9L2Y1Q4QXAM409C">
   <FIELD Code="vrapp.flex">True</FIELD>
   <FIELD Code="job.availnrq">True</FIELD>
   <FIELD Code="job.userRTF.1">Some text. It's nice.</FIELD>
   <FIELD Code="job.skills">1BF588R7IC3HV5IZ,546ZL0FOJT5L92LC,QXQMY8DVERA07BQU</FIELD>
   <FIELD Code="job.duties">Do work.</FIELD>
   <FIELD Code="cfield.8E2BWUUVSQZHWQLA">0</FIELD>
   <FIELD Code="cfield.O34UIYF789TOH30H"></FIELD>
</REQUEST>
```

Returns

```
<POST Valid="1" Code="42Z125M1S9OWCGVUSOJ5ROOLKE41KFK3"/>
<OK Command="vrequest.post" Result=""/>
<Redirect Delay="10">http://www.VSysOne.com</Redirect>
```

vrequest.setstatus

Description

Updates the status of a single request; usually only available to cancel a request.

Parameters

Name	Required	Type	Description
User	Required	Text	16-digit ID of the currently logged-in user.
Code	Required	Text	32-digit ID of the pre-existing request.
Status	Required	Text	New status for the request.

Example

```
<REQUEST command="vrequest.setstatus" user="VOLUNTEERREQAPP0" code="C7TOJXZ5CEBE07C1UK281ZO8LJ2U0GIL" status="ucancel"/>
```

Returns

```
<OK Command="vrequest.setstatus" Result=""/>
```

vrequest.user.list

Description

Returns a list of volunteer request applications associated with the current user.

- May return zero or more fields *FILTERFIELDS/FILTERFIELD/* that can be used to filter this list. These fields use the same specifications (*Type*, *OPTIONS*, etc.) as fields do in an application.
- For each request, returns zero or more `AVAILABLESTATUSES/AVAILABLESTATUS/` elements representing statuses to which the application can be changed using `vrequest.setstatus`.

Parameters

Name	Required	Type	Description
`User`	Required	Text	16-digit ID of the currently logged-in user.
(others)	Optional	(varies)	To apply filters to this request, send as attributes the values to be applied as filters – one per filter. (Checklist field results should be sent as a comma-delimited list of selected values.)

Example

```
<REQUEST command="vrequest.user.list" user="VOLUNTEERREQAPP0">
   <FILTER Code="stat" Value="pend"/>
   <FILTER Code="sub.min" Value="2015-01-01"/>
</REQUEST>
```

Returns

```
<REQUESTS>
   <REQUEST Code="42Z125M1S9OWCGVUSOJ5ROOLKE41KFK3" Owner="VOLUNTEERREQAPP0" Layout="E9L2Y1Q4QXAM409C"
   LayoutDescription="Regular (scheduled) volunteers" Description="Regular (scheduled) volunteers" StatusCode="pend"
   Status="Pending" Submitted="2015/04/21 16:53:11.757" LastChanged="2015/04/21 16:53:12.095" CanEdit="1">
      <AVAILABLESTATUSES>
         <AVAILABLESTATUS Code="ucancel" Description="Submitter cancelled"/>
         <AVAILABLESTATUS Code="part" Description="Partially complete"/>
      </AVAILABLESTATUSES>
   </REQUEST>

   <REQUEST Code="C7TOJXZ5CEBE07C1UK281ZO8LJ2U0GIL" Owner="VOLUNTEERREQAPP0" Layout="E9L2Y1Q4QXAM409C"
   LayoutDescription="Regular (scheduled) volunteers" Description="Regular (scheduled) volunteers" StatusCode="ucancel"
   Status="Submitter cancelled" Submitted="2015/04/19 12:58:39.453" LastChanged="2015/04/21 16:44:06.548" CanEdit="0">
   </REQUEST>
</REQUESTS>

<FILTERFIELDS>
   <FILTERFIELD Code="typ" Type="C" Description="Type">
      <OPTIONS>
         <OPTION Code="E9L2Y1Q4QXAM409C" Value="Regular (scheduled) volunteers"/>
         <OPTION Code="M907XB2GN8CLHLWG" Value="Flex volunteer"/>
         <OPTION Code="NNUAXHIT5SOGW9WE" Value="Intern request"/>
      </OPTIONS>
   </FILTERFIELD>
   <FILTERFIELD Code="stat" Type="C" Description="Status">
      <OPTIONS>
         <OPTION Code="app" Value="Approved"/>
         <OPTION Code="bounce" Value="Returned to requestor"/>
         <OPTION Code="part" Value="Partially complete"/>
         <OPTION Code="pend" Value="Pending"/>
         <OPTION Code="rej" Value="Rejected"/>
         <OPTION Code="ucancel" Value="Submitter cancelled"/>
      </OPTIONS>
   </FILTERFIELD>
   <FILTERFIELD Code="sub.min" Type="D" Description="Date submitted (Minimum)"/>
   <FILTERFIELD Code="sub.max" Type="D" Description="Date submitted (Maximum)"/>
</FILTERFIELDS>
```

VSys Live Kiosk

Certain VOXI functions are specific to the kiosk functionality of VSys Live and the corresponding setup tools within the VSys Kiosk.

kiosk.settings

Description
Returns a list of general VSys Kiosk settings.

Parameters
(none)

Example
```
<REQUEST command="kiosk.settings"/>
```

Returns
```
<SETTINGS Enabled="1" LoginButtonText="GO" RequireSiteAccessLogin="1" RequirePassword="0" HidePINS="1" PINChar="*"
NumericPINs="1" DynamicPINs="1" TitleBarText="Kiosk Site" GeneralInactivityTimeout="30"
GeneralPromptDismissalTimeout="25">
   <SITELOGINS>
      <SITELOGIN UserID="User" Password="pw1"/>
      <SITELOGIN UserID="User2" Password="pw2"/>
   </SITELOGINS>
</SETTINGS>
```

Property	Meaning
Enabled	If "0", VSys Live Kiosk should not function.
LoginButtonText	Caption for the [Go] button on the login page.
RequireSiteLoginAccess	If "1", see `kiosk.validatesiteaccess` below.
TitleBarText	If present, use as the page title.
RequirePassword	If "1", after accepting a PIN, prompt the user for password and return that password to `kiosk.pinlogin`.
HidePINs	If "1", rather than displaying the actual PIN as it's entered, show a string the same length as the entered PIN but comprised of the character returned as *PINChar*.
PINChar	See *HidePINs*.
NumericPINs	If "0", an alphanumeric rather than numeric keypad should be shown. (In the initial implementation this will be a purely numeric value; in future implementations it may be alphanumeric.)
DynamicPINs	If "0", once at least three characters have been entered, after each keypad button press, call `kiosk.dynamicpinlogin`. If this fails, display no error; if it succeeds, treat the same as if `kiosk.pinlogin` was called and succeeded.
GeneralInactivityTimeout	Where a block does not provide a configurable inactivity timeout, or where that configurable value is zero, use this value (in seconds) to determine how long a block should wait for user interaction before timing out and either resetting itself or returning to the *Login* page.
GeneralPromptDismissalTimeout	Where a block does not provide a configurable prompt display timeout, or where that configurable value is zero, use this value (in seconds) to determine how long a prompt or notification should stay visible before closing itself automatically.

kiosk.validatesiteaccess

Description
A VSys Live Kiosk site can require HTTP Basic authentication for access in a browser. This is not intended to authenticate users, but to verify that access to the browser-based kiosk is permitted.

Validate the browser-provided credentials against the list of *SITELOGINS/SITELOGIN/* values returned by `kiosk.settings` or call `kiosk.validatesiteaccess`. When manually validating these values, User IDs are not considered case-sensitive, but passwords are.

Example
```
<REQUEST command="kiosk.validatesiteaccess" userid="User2" password="pw2"/>
```

Returns
An *OK* or *ERROR* element.

kiosk.dynamicpinlogin, kiosk.pinlogin

Description

Attempts to log in a volunteer using the volunteer's kiosk PIN. If the given `PIN` value does not match the swipe/scan value of any volunteer, VOXI will attempt to match the value against a volunteer's 16-digit ID and then on 5-digit basic ID.

`kiosk.dynamicpinlogin` should be used when the volunteer has entered some PIN characters but has no yet pressed the "Login" or other commit button. It will succeed if and only if the given PIN matches exactly one user and no other user's PIN begins with that sequence. Dynamic PIN login is incompatible with *`RequirePassword`*.

Parameters

Name	Required	Type	Description
`PIN`	Required	Text	User-provided PIN.
`Password`	Optional	Text	Required only if *`RequirePassword`* returned from `kiosk.settings`.

Example

```
<REQUEST command="kiosk.pinlogin" PIN="12345"/>
```

Returns

An *`OK`* or *`ERROR`* element. An *`OK`* element means that VSys Live Kiosk should check for messages for the user, showing those messages if present or advancing to *Check In* or *Check Out*. The presence of one or more *`OPEN`* elements indicates that the volunteer is currently checked-in for an assignment; see `checkin.checkedin` for more details. An *`ERROR`* element will contain a message text that should be displayed to the volunteer.

```
<OK Code="EUW38W898JD7YZU5" Name="Cornett, William A" PIN="12345" UserID="wcornett@bespoke.us">
   <OPEN Code="6L0NR6QJT0P31ZN7" Start="2011/06/18 18:00:00.000" Location="Downstairs" LocationCode="AJKDL1TABCWBOABP"
   Job="Catbox Cleaning V" JobCode="ZWM2M14HD6E8T67Z" JobGroup="Cat Duties" JobGroupCode="U7JYV7DKP03UBU1Y"/>
</OK>
```

or

```
<ERROR>
   Volunteer "Joe Schmoe" is not permitted to log in via the kiosk due to a missing TB test certification.
</ERROR>
```

kiosk.swipelogin

Description

Attempts to log in a volunteer using the volunteer's "swipe" value, usually a value assigned via a barcode or magnetic stripe. If the given `Swipe` value does not match the swipe/scan value of any volunteer, VOXI will attempt to match the value against a volunteer's 16-digit ID and then on 5-digit basic ID.

`kiosk.swipelogin` should be used when a string is entered using other than the numeric keypad, and usually followed by a trailing CR/LF or CR. (Any trailing CR and LF characters provided here will be ignored.) Note that passwords associated with a volunteer's profile are ignored when using a swipe login.

Parameters

Name	Required	Type	Description
`Swipe`	Required	Text	Provided text.

Example

```
<REQUEST command="kiosk.swipelogin" swipe="1263464856456456"/>
```

Returns

An *`OK`* or *`ERROR`* element. See `kiosk.dynamicpinlogin` above.

Social Media - General

`social.shortenURL.google`

Description

Applies Google's URL shortener to a given URL.

Parameters

Name	Required	Type	Description
`URL`	Required	Text	Valid URL.

Example

```
<REQUEST command="social.shortenURL.google" URL="https://www.VSysOne.com"/>
```

Returns

```
<OK Command="social.shortenURL.google" Result="http://goo.gl/RzhfG4"/>
```

Social Media - Facebook

social.facebook.profiles

Description
Returns a list of available Facebook profiles.

Parameters
(none)

Example
```
<REQUEST command="social.facebook.profiles"/>
```

Returns
```
<PROFILE Code="VNQGJ2JW1ZBKU912" Profile="xxx@gmail.com" Description="Personal" Enabled="1"/>
<PROFILE Code="YVPGGG0Q9D59CQPB" Profile="open_ngsvahb_user@tfbnw.net" Description="Test account" Enabled="1"/>
```

social.facebook.post

Description
Posts content to the given Facebook profile’s timeline.

Parameters

Name	Required	Type	Description
`Profile`	Required	Text	*Code* from `social.facebook.profiles`.
`Link`	Optional	Text	Hypertext link to be used in posting.
`Image`	Optional	Text	URL of image to be included with posting.

Example
```
<REQUEST command="social.facebook.post" Profile="YVPGGG0Q9D59CQPB" Link="http://www.VSysOne.com"
Image="http://www.vsysone.com/sites/default/files/logo_0.png">
   Hey, things are going great for us – we have VSys Live now!
</REQUEST>
```

Returns
An *OK* or *ERROR* element.

social.facebook.tokendebug

Description
Describes a Facebook authentication token.

Parameters

Name	Required	Type	Description
`Token`	Required	Text	Token to be checked.
`AccessToken`	Required	Text	Your app access token or a valid user access token from a developer of the app.

Example
```
<REQUEST command="social.facebook.tokendebug" profile="UG4IOTU34OTUFIOH" token="xxx" accesstoken="yyy"/>
```

Returns
An *OK* or *ERROR* element.
```
<OK app_id="691368574219186" application="VSys One Integration" expires_at="1398978882" is_valid="1"
issued_at="1393794882" user_id="100007605694514"/>

<ERROR Message="Invalid OAuth access token." Code="190"/>
```

Social Media - Twitter

social.twitter.profiles

Description
Returns a list of available Twitter profiles.

Parameters
(none)

Example
```
<REQUEST command="social.twitter.profiles"/>
```

Returns
```
<PROFILE Code="RI16V9Q8VQ7Y1WU1" Profile="EvilGeekUS" Description="Evil 1" Enabled="1"/>
```

social.twitter.post

Description
Posts content to the given Twitter profile's feed.

Parameters

Name	Required	Type	Description
`Profile`	Required	Text	*Code* from `social.twitter.profiles`.

Example
```
<REQUEST command="social.twitter.post" Profile="RI16V9Q8VQ7Y1WU1">
   Hey, things are going great for us – we have VSys Live now!
</REQUEST>
```

Returns
An *OK* or *ERROR* element.

social.twitter.followers

Description
Retrieves a list of followers for a Twitter profile.

Parameters

Name	Required	Type	Description
`Profile`	Required	Text	*Code* from `social.twitter.profiles`.

Example
```
<REQUEST command="social.twitter.followers" Profile="RI16V9Q8VQ7Y1WU1"/>
```

Returns
An *OK* or *ERROR* element; *OK* includes zero or more *FOLLOWER* elements.
```
<FOLLOWER Code="2375341514"/>
```

RSS

rssfeed.list

Description
Returns a list of available RSS feeds for this site.

Parameters
(none)

Example
```
<REQUEST command="rssfeed.list"/>
```

Returns
```
<FEEDS>
   <FEED Code="FQTSP3KD64FYHIAJ" UserCode="PLANT" Description="Feed Me, Seymour!"/>
   <FEED Code="TLYW3KGTRQ8FDBHE" UserCode="FED" Description="Sir Fed-a-Lot"/>
</FEEDS>
```

rssfeed.get

Description
Retrieves the current items in a given RSS feed.

Parameters

Name	Required	Type	Description
`Code`	Required	Text	*`Code`* or *`UserCode`* from `rssfeed.list`.

Example
```
<REQUEST command="rssfeed.get" Code="PLANT">
```

Returns
```
<rss version="2.0">
   <channel>
      <title>Seymour's Musings</title>
      <description>Feed Me, Seymour!</description>
      <link>http://www.vsysone.com</link>
      <category>Volunteer Info</category>
      <generator>VSys Live</generator>
      <webMaster>test@volunteers.org</webMaster>
      <managingEditor>manager@volunteers.org</managingEditor>
      <pubDate>Wed, 21 May 2014</pubDate>
      <item>
         <title>Academic Standing expiring Cornett, Terry: 09/03/2012-05/30/2014</title>
         <description></description>
      </item>
       <item>
         <title>VSys One now supports LinkedIn...</title>
         <description>Test me.&#xD;&#xA;&#xD;&#xA;</description>
         <link>www.VSysOne.com</link>
      </item>
      <item>
         <title>VSys Live News Item</title>
         <description>And I'm the summary.&#xD;&#xA;</description>
      </item>
   </channel>
</rss>
```

Diagnostics commands

diag.sendemail

Description

Sends an e-mail directly using VOXI itself. This is intended to test VOXI's ability to send e-mails (password resets, notifications, etc.) rather than as a message delivery tool.

Parameters

Name	Required	Type	Description
`Sender`	Required	E-mail address	E-mail address only, no other name attributes.
`Recipient`	Required	E-mail address	E-mail address only, no other name attributes.
`Subject`	Optional	Text	
`Body`	Required	Text	

Example

```
<REQUEST command="diag.sendemail" sender="someone@vsysone.com" recipient="another@vsysone.com" subject="Test e-mail"
body="Joy? No joy? Almond joy?"/>
```

Returns

An `OK` or `ERROR` element, with a `LOG` element containing various details.

```
<OK>
   <LOG>
      <![CDATA[Resolving hostname mail.vsysone.com.
      Connecting to 166.70.12.30.
      Connected.
      Encoding text
      Encoding text
      Successfully e-mailed
      ]]>
   </LOG>
</OK>
```

diag.sendsms

Description

Sends an SMS/text message directly using VOXI itself. This is intended to test VOXI's ability to send text messages (password resets, notifications, etc.) rather than as a message delivery tool.

Parameters

Name	Required	Type	Description
`Sender`	Optional	Phone number	Must be a valid SMS phone number associated with your Twilio account and enabled in VSys One. If not provided, the first known, valid defined number in VSys will be used.
`Recipient`	Required	Phone number	
`Body`	Required	Text	

Example

```
<REQUEST command="diag.sendsms" sender="5185551212" recipient="5185552323" body="Joy? No joy? Almond joy?"/>
```

Returns

An `OK` or `ERROR` element.

VSys Live Site Metadata

For VSys Live, VOXI requires certain information about how VSys Live is configured. This metadata includes details about the site's template such as its name, regions, localizable properties, and configurable properties. VSys One can query that information from VSys Live or VSys Live can post that information to VOXI.

vlive.metadata.post

Description
Posts the given data to VOXI and VSys One as the new metadata for the site under which VOXI is currently running. Note that this new metadata is posted as "pending", meaning that it does not replace/overwrite the existing metadata until a user explicitly approves this in the VSys Live setup tool.

Parameters
(none)

Example
```
<REQUEST command="vlive.metadata.post">
   <VLIVE Version="1.0">
      <TEMPLATE Name="This is the name of our template" Code="CustomTemplate14">
         <REGIONS>
            <REGION Code="topleft" Name="Top left"/>
            <REGION Code="topright" Name="Top right"/>
            <REGION Code="mmenu" Name="Main menu region"/>
            <REGION Code="mbody" Name="Main body"/>
            <REGION Code="footleft" Name="Footer left"/>
            <REGION Code="footmid" Name="Footer middle"/>
            <REGION Code="footright" Name="Footer right"/>
         </REGIONS>
         <LOCALIZATIONS>
            <ENTRY Code="main.welcometext" Default="Welcome to our organization"/>
            <ENTRY Code="main.headedropdowncaption" Default="Some Caption"/>
         </LOCALIZATIONS>
         <CONFIGURABLES>
            <CONFIGURABLE Code="template.header" Description="Top graphic" Type="graphic"/>
            <CONFIGURABLE Code="template.orgname" Description="Org name" Type="text"/>
         </CONFIGURABLES>
      </TEMPLATE>
      <ACTIONS>
         <ACTION Code="logout" Description="Logs out a currently signed-in user" Mode="auth"/>
      </ACTIONS>
      <BLOCKS>
         <BLOCK Name="Sign in" Code="signin" Description="Generic signin block" Mode="Unauth">
            <LOCALIZATIONS>
               <ENTRY Code="module.signin.okbutton" Default="OK"/>
               <ENTRY Code="module.signin.cancelbutton" Default="Cancel"/>
            </LOCALIZATIONS>
            <CONFIGURABLES>
               <CONFIGURABLE Code="signin.showforgot" Description="Show 'Forgot password?' button" Type="checkbox"/>
            </CONFIGURABLES>
         </BLOCK>
         <BLOCK Name="Assignment listings" Code="assignmentview" Description="Volunteer's assignment listings (not
         calendar)" Mode="Auth">
            <LOCALIZATIONS>
               <ENTRY Code="module.assignmentview.okbutton" Default="OK"/>
               <ENTRY Code="module.assignmentview.closebutton" Default="Close"/>
               <ENTRY Code="module.assignmentview.cancelbutton" Default="Cancel"/>
            </LOCALIZATIONS>
            <CONFIGURABLES>
               <CONFIGURABLE Code="assignments.pagesize" Description="Number of items to show on each page"
               Type="integer" MinValue="10" MaxValue="99999"/>
               <CONFIGURABLE Code="assignments.mode" Description="Display mode" Type="combo" Default="l">
                  <ITEM Code="c" Description="Calendar"/>
                  <ITEM Code="l" Description="Listing"/>
               </CONFIGURABLE>
            </CONFIGURABLES>
         </BLOCK>
      </BLOCKS>
   </VLIVE>
</REQUEST>
```

Returns
An *OK* or *ERROR* element.