Serious Scheduling

Web Services Technical Guide

Document Version 2.0.20150623

Last Updated: 6/23/2015 4:17 PM

Table of Contents

Overview

The Appointment-Plus Web Services system provides clients with access to their appointment, customer, staff, and services data to use in other applications outside of the Appointment-Plus GUI. This allows clients to integrate AP with their other existing applications and systems.

The AP Web Services system is a loosely RESTful service. Requests can be sent via HTTP to specific URI's, depending on the desired action. The AP Web Service returns an XML document, a JSON string or a JSONP callback as the response.

Usage Requirements for XML

The XML Web Services can be invoked by sending an HTTP POST request to a specified URI. The response will be an XML document.

The following items are required to be sent with every request:

Required HTTP Headers

Name	Valid Values	Notes
Authorization	Authorization: Basic *site_id*:*api_key*	Must set the Authorization HTTP header using Basic Authorization and pass the *site_id* and *api_key* as the username:password pair

Required POST Fields

Name	Valid Values	Notes
response_type	xml	Must specify the response type as XML

Optional POST Fields

Name	Valid Values	Notes

encoding	valid encoding types	For XML only, you can specify the desired encoding type for the returned XML document.
		
		For example, if you send:
		
		`encoding=UTF-16`
		
		the XML returned will include:
		
		`<?xml version="1.0" encoding="UTF-16" ?>`

Each service will have its own set of required and/or optional POST fields that can be used to filter the result set for Retrieve functions, or that can be used to insert data for Create and Update functions. More details about the POST fields that can be sent can be found in the Functions/Services Details section of this document.

A typical response will include metadata about the initial request in addition to the actual queried data. This metadata includes the type of request/action, the resource, the result (success/fail), any errors, and the number of results returned. For details of specific fields returned for each service, see the Functions/Services Details section of this document.

Example XML Response

An example of a request and an XML response is as follows. The details for each of the items returned would be in the `<data>` element. The example below is for getting the location resource, thus the child nodes are `<location>`. If the resource was Appointments or Customers the child node would be `<appointment>` or `<customer>`, respectively.

Request

Description:	Get location details for location_id 84
URI:	https://ws.securedata-trans1u.com/Locations/GetLocations
HTTP Header:	Authorization: Basic appointplus000/01:abcdef1234abcdef1234
POST:	response_type=xml&location=84

Response

```
<?xml version="1.0" encoding="utf-8" ?>
<APResponse>
       <resource>locations</resource>
       <action>getlocations</action>
       <request>GetLocationsById</request>
       <result>success</result>
       <errors>
              <error>If there were any errors they would show here.</error>
       </errors>
       <count>1</count>
       <data>
              <location>
                     <c_id>49</c_id>
                     <location_id>84</location_id>
                     <headquarters>N</headquarters>
                     <sort_order>2</sort_order>
                     <location_name>PA - Pottsville Campus</location_name>
                     <headquarters_id>49</headquarters_id>
              </location>
       </data>
</APResponse>
```

Usage Requirements for JSON

The JSON Web Services can be invoked by sending an HTTP POST request to a specified URI. The response will be a string formatted as a JSON object.

The following items are required to be sent with every request:

Required HTTP Headers

Name	Valid Values	Notes
Authorization	Authorization: Basic *site_id*:*api_key*	Must set the Authorization HTTP header using Basic Authorization and pass the *site_id* and *api_key* as the username:password pair

Required POST Fields

Name	Valid Values	Notes
response_type	json	Must specify the response type as JSON

Each service will have its own set of required and/or optional POST fields that can be used to filter the result set for "Get" functions, or that can be used to insert data for "Create" and "Update" functions. More details about the POST fields that can be sent can be found in the Functions/Services Details section of this document.

A typical response will include metadata about the initial request in addition to the actual queried data. This metadata includes the type of request/action, the resource, the result (success/fail), any errors, and the number of results returned. For details of specific fields returned for each service, see the Functions/Services Details section of this document.

Example JSON Response

An example of a request and a JSON response is as follows. The details for each of the items returned would be in the "`data`" element.

Request

Description:	Get location details for all locations
URI:	https://ws.securedata-trans1u.com/Locations/GetLocations
HTTP Header:	Authorization: Basic appointplus000/01:abcdef1234abcdef1234
POST:	response_type=json

Response

```
{
    "resource": "locations",
    "action": "getlocations",
    "request": "GetLocationsAll",
    "result": "success",
    "count": "2",
    "data":
        [{
            "c_id": "617",
            "location_id": "617",
            "headquarters": "Y",
            "sort_order": "0",
            "location_name": "Scottsdale",
            "headquarters_id": "617"
        },
        {
            "c_id": "617",
            "location_id": "620",
            "headquarters": "N",
            "sort_order": "1",
            "location_name": "San Francisco",
            "headquarters_id": "617"
        }]
}
```

Usage Requirements for JSONP

The JSONP version of AP Web Services functions very differently from the XML and JSON versions. Using JSONP allows for cross-domain requests from the client-side (inside the browser), usually via AJAX. The response is a JavaScript function containing the result set in JSON format.

JSONP request may only be sent as HTTP GET requests, and do not require the Authorization header that XML and JSON requests require.

The following items are required to be sent with every JSONP request:

Required GET Fields

Name	Valid Values	Notes
response_type	jsonp	Must specify the response type.
callback	JavaScript function name	Must be a valid JavaScript function to execute after successful response. This function name will be added as the padding in the JSONP response.
site_id	Your Site ID	
api_key	Your API Key	

Each service will have its own set of required and/or optional GET fields that can be used to filter the result set for Retrieve functions, or that can be used to insert data for Create and Update functions. More details about the GET fields that can be sent can be found in the Functions/Services Details section of this document.

A typical response will include metadata about the initial request in addition to the actual queried data. This metadata includes the type of request/action, the resource, the result (success/fail), any errors, and the number of results returned. For details of specific fields returned for each service, see the Functions/Services Details section of this document.

Example JSONP Response

An example of a request and a JSONP callback response is as follows. The details for each of the items returned would be in the “`data`” element.

Request

Description:	Get location details for all locations
URI:	https://ws.securedata-trans1u.com/Locations/GetLocations
GET:	response_type=jsonp&callback=myFunction&site_id=ap123/456&api_key=abcde12345

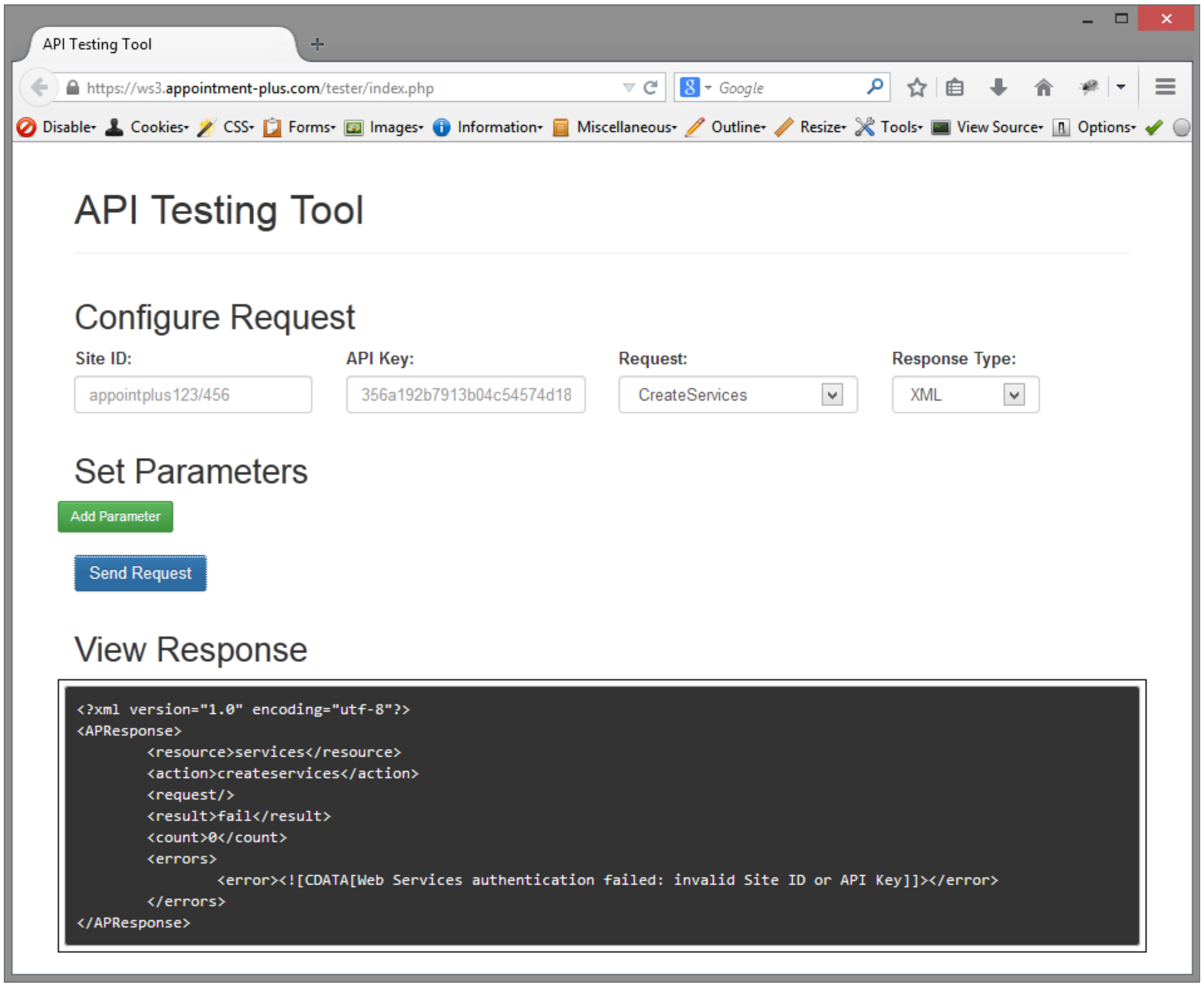
Response

```
myFunction({
    "resource": "locations",
    "action": "getlocations",
    "request": "GetLocationsAll",
    "result": "success",
    "count": "2",
    "data":
        [{
            "c_id": "617",
            "location_id": "617",
            "headquarters": "Y",
            "sort_order": "0",
            "location_name": "Scottsdale",
            "headquarters_id": "617"
        },
        {
            "c_id": "617",
            "location_id": "620",
            "headquarters": "N",
            "sort_order": "1",
            "location_name": "San Francisco",
            "headquarters_id": "617"
        }]
})
```

API Testing Tool

An online testing tool is available at https://ws.appointment-plus.com/tester/index.php. You can use this to send requests against the live service and view the responses.

Valid working credentials are required to use the testing tool.


API Testing Tool
https://ws3.appointment-plus.com/tester/index.php
Google
Disable Cookies CSS Forms Images Information Miscellaneous Outline Resize Tools View Source Options

API Testing Tool

Configure Request

Site ID: appointplus123/456

API Key: 356a192b7913b04c54574d18

Request: CreateServices

Response Type: XML

Set Parameters

Add Parameter
Send Request

View Response

```
<?xml version="1.0" encoding="utf-8"?>
<APResponse>
        <resource>services</resource>
        <action>createservices</action>
        <request/>
        <result>fail</result>
        <count>0</count>
        <errors>
                <error><![CDATA[Web Services authentication failed: invalid Site ID or API Key]]></error>
        </errors>
</APResponse>
```

Available Methods

Live Base URL: https://ws.appointment-plus.com

Method Details

The following contains detailed information about each available web service method. You will find a brief description, a list of required and optional POST/GET variables to send, and a list of the returned fields. For definitions of the required variables and the returned fields, please see the Web Services Data Dictionary.

**Note that the Required HTTP Headers only apply to XML and JSON response types. The site_id and api_key key fields must be sent via GET for the JSONP response type.

Appointments

GetAppointments		
URI: /Appointments/GetAppointments		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp)	**Optional POST/GET Vars:** date (date: yyyymmdd) start_date (date: yyyymmdd) end_date (date: yyyymmdd) appointment (integer,multiple) customer (integer,multiple) staff (integer,multiple) service (integer,multiple) location (integer,multiple) status (integer,multiple) first_name (text,multiple) last_name (text,multiple) updated(date/time:yyyymmddhhii) created (date/time:yyyymmddhhii)

Description:	**Returns**:
Returns the details for appointments.The returned dataset can be filtered by sending any of the optional POST/GET variables. All string variables are case-insensitive. For POST/GET variables that accept multiple values, separate values with a comma (e.g. *location=1331,2442,3553*) If none of the optional POST/GET variables are sent, the service will return all of the appointments for the current day. Note that start_date and end_date must be used in tandem to return all appointments within that date range.	c_id, appt_id, customer_id, account, last_name, first_name, middle_name, employee_id, staff_screen_name, staff_type_id, room_id, date, start_time, end_time, service_id, service, event_template_id, customer_notes, employee_notes, status_id, appt_status_description, rep_id, cost, tip, payment_type_id, coupon_code, type_id, rebook_id, spots, recur_id, reason, creation_timestamp, creation_emp_id, last_timestamp, last_emp_id, customer_package_id, duration_id, link_id, location_id, appt_status_type, lead_description, service_time_description, po_number, addons **If using legacy vehicle fields, the following are also returned:** make_id, model_id, model_year, other_vehicle, vin, odometer

Using ‘updated’ and ‘created’ will filter the resultset to only include appointments that have been updated or created since the given timestamp. Please see the Data Dictionary for details on the format of the timestamp.	**If using the ACES Vehicle fields, the following are also returned:** aces_year, aces_make, aces_model, aces_submodel

CreateAppointments		
URI: /Appointments/CreateAppointments		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) c_id (integer) customer_id (integer) employee_id (integer) date (date: yyyymmdd) start_time (time [in minutes]) service_id (integer)	**Optional POST/GET Vars:** type_id (integer) room_id (integer) addons (integer, multiple) end_time (time [in minutes]) customer_notes (text) employee_notes (text) rep_id (integer) cost (decimal) tip (decimal) payment_type_id (integer) customer_package_id (integer) spots (integer) reason (text) creation_emp_id (integer) last_emp_id (integer) po_number (text) coupon_code (text) make_id (integer) model_id (integer) model_year (integer) other_vehicle (text) vin (text) odometer (text) aces_year (integer) aces_make (text) aces_model (text) aces_submodel (text) override _aces_submodel (boolean) override (boolean)
Description: Creates an appointment. The c_id field should contain the location_id of the location where you want to book the appointment. The *start_time* and *end_time* fields should be represented by the number of minutes past 12:00am. For example, 8:00am should be sent as ‘480’ (8hrs x 60min = 480min) and 5:00pm should be sent as ‘1020’ (17hrs x 60min = 1020min). If you don’t send	**Returns**: c_id, appt_id, customer_id, account, employee_id, room_id, date, start_time, end_time, service_id, addons, customer_notes, employee_notes, status_id, rep_id, cost, tip, payment_type_id, spots, reason, creation_emp_id, last_timestamp, last_emp_id, po_number, coupon_code **If using legacy vehicle fields, the following are also returned:** make_id, model_id, model_year, other_vehicle, vin,	

an *end_time*, the system will calculate *end_time* based on the default service duration. When sending *addons*, you may send a comma separated list of the addon service service_id's. For example: *addons=1357,2468,987*. When sending *coupon_code*, you may also want to send *cost*, as this method will not automatically discount the cost of the appointment. If the new appointment does not fit into the selected schedule (based on date/time, location, employee, and service) the method will return an error. You can bypass the availability checks and force the appointment to be booked by providing the '*override=true*' parameter. When providing ACES vehicle fields, the year, make, and model are required and must be a valid combination. The submodel is optional, but must also be valid if provided. The *override _aces_submodel* flag will attempt to lookup the vehicle information **without** the submodel if the method can't find it initially using the submodel.	odometer **If using the ACES Vehicle fields, the following are also returned:** aces_year, aces_make, aces_model, aces_submodel

CreateReserves		
URI: /Appointments/CreateReserves		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) c_id (integer) customer_id (integer) employee_id (integer) room_id (integer) date (date: yyyymmdd) start_time (time [in minutes]) status_id (integer)	**Optional POST/GET Vars:** end_time (time [in minutes]) reason (text) creation_emp_id (integer) last_emp_id (integer)

Description: Creates a reserve in a time slot. The c_id field should contain the location_id of the location where you want to book the appointment. Only one of either *room_id* or *employeed_id* is required, depending on which you want to reserve. The *start_time* and *end_time* fields should be represented by the number of minutes past 12:00am. For example, 8:00am should be sent as '480' (8hrs x	**Returns**: c_id, appt_id, customer_id, account, employee_id, room_id, date, start_time, end_time, service_id, addons, customer_notes, employee_notes, status_id, rep_id, cost, tip, payment_type_id, spots, reason, creation_emp_id, last_timestamp, last_emp_id, po_number, coupon_code **If using legacy vehicle fields, the following are also returned:** make_id, model_id, model_year, other_vehicle, vin, odometer

60min = 480min) and 5:00pm should be sent as '1020' (17hrs x 60min = 1020min). If you don't send an *end_time*, the system will calculate *end_time* based on the default service duration.	**If using the ACES Vehicle fields, the following are also returned:** aces_year, aces_make, aces_model, aces_submodel

UpdateAppointments		
URI: /Appointments/UpdateAppointments		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) appt_id (integer, multiple)	**Optional POST/GET Vars:** customer_id (integer) employee_id (integer) room_id (integer) date (date: yyyymmdd) start_time (time [in minutes]) end_time (time [in minutes]) service_id (integer) customer_notes (text) employee_notes (text) status_id (integer) rep_id (integer) cost (decimal) tip (decimal) payment_type_id (integer) customer_package_id (integer) spots (integer) reason (text) creation_emp_id (integer) last_emp_id (integer) po_number (text) coupon_code (text) make_id (integer) model_id (integer) model_year (integer) other_vehicle (text) vin (text) odometer (text) aces_year (integer) aces_make (text) aces_model (text) aces_submodel (text) override _aces_submodel (boolean) override (boolean)

Description: Updates details of an appointment or reserve, or of multiple appointments/reserves if passing a comma separates list of *appt_id*'s. Only the fields/values you send to the service will be updated, so you may update any number and combination of fields. Sending no data will simply	**Returns**: c_id, appt_id, customer_id, account, employee_id, room_id, date, start_time, end_time, service_id, customer_notes, employee_notes, status_id, rep_id, cost, tip, payment_type_id, spots, reason, creation_emp_id, last_timestamp, last_emp_id, po_number, coupon_code

update the *last_timestamp* field with the current timestamp. If you send a field with no value, the system will assume you want to clear the value that is currently in the field. When sending *coupon_code*, you may also want to send *cost*, as the service will not automatically discount the cost of the appointment. The *start_time* and *end_time* fields should be represented by the number of minutes past 12:00am. For example, 8:00am should be sent as '480' (8hrs x 60min = 480min) and 5:00pm should be sent as '1020' (17hrs x 60min = 1020min). If you don't send an *end_time*, the system will calculate *end_time* based on the default service duration. If the appointment does not fit into the selected schedule (based on date/time, location, employee, and service) the service will throw an error. You can bypass the availability checks and force the appointment to be booked by providing the 'override=true' parameter. When providing ACES vehicle fields, the year, make, and model are required and must be a valid combination. The submodel is optional, but must also be valid if provided. The *override _aces_submodel* flag will attempt to lookup the vehicle information **without** the submodel if the method can't find it initially using the submodel.	**If using legacy vehicle fields, the following are also returned:** make_id, model_id, model_year, other_vehicle, vin, odometer **If using the ACES Vehicle fields, the following are also returned:** aces_year, aces_make, aces_model, aces_submodel

GetAppointmentStatuses		
URI: /Appointments/GetAppointmentStatuses		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp)	**Optional POST/GET Vars:**
Description: Returns all appointment statuses.	**Returns**: c_id, status_id, sort_order, description, status_type, color, display	

CreateAppointmentStatuses		
URI: /Appointments/CreateAppointmentStatuses		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) description (text) color (hexadecimal color code) display (yes | no)	**Optional POST/GET Vars:**
Description:	**Returns**:	

Create a single or multiple (using XML) appointments statuses.	c_id, status_id, sort_order, description, status_type, color, display

UpdateAppointmentStatuses		
URI: /Appointments/UpdateAppointmentStatuses		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) status_id (integer)	**Optional POST/GET Vars:** sort_order (integer) description (text) color (hexadecimal color code) display (yes | no)
Description: Updates details of a single or multiple (using XML) appointment statuses. Note that this is different from updating the status of an appointment, which is done by using UpdateAppointment and sending a new *status_id*.	**Returns**: c_id, status_id, sort_order, description, status_type, color, display	

GetOpenSlots		
URI: /Appointments/GetOpenSlots		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp)	**Optional POST/GET Vars:** store_number (text) start_date (date: yyyymmdd) start_time (time [in minutes]) num_days (integer) num_slots (integer) location (integer) staff (integer) room (integer) service (integer) addons (integer, multiple) show_duplicates (yes | no) randomize (yes | no) limit_start_time (integer) limit_end_time (integer) include_time_zone (yes | no)
Description: Returns all available time slots for *num_days* number of days after the *start_date* date. The returned dataset can be filtered by sending any of the optional POST/GET variables. Using the *num_days* variable allows you to specify how many days of open time slots to return. Additionally, you can further limit the resultset using the *num_slots* variable, to only return that number of time slots. The *location*, *staff*, *room*, *service,* and *addons* variables will filter the results to only include time	**Returns**: c_id, date, start_time, employee_id, room_id, num_appts, num_slots **If using the *include_time_zone* parameter, the following are also returned:** time_zone, gmt_timestamp	

slots where each of those is available. The filter will find the total duration needed based on the *service* and *addons* variables. When sending *addons*, you may send a comma separated list of the addon service service_id's. For example: *addons=1357,2468,987*. By default, the service will return "duplicate" time slots, if more than one staff member is available during that time. To disable this and only return unique time slots, use *show_duplicates=no*. The service will return the time slot for the first staff member based on the staff sort order. If none of the optional POST/GET variables are sent, the service will return all of the open time slots for the current day. Note that this service checks the current time (based on your account's time zone settings) and only returns future open time slots. If you specify a past date for *start_date*, you will still only receive future open slots. Using the *limit_start_time* and *limit_end_time* parameters will filter the resultset to only return time slots that fall within that time range.	

DeleteReserve		
URI: /Appointments/DeleteReserve		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) employee_id (integer) room_id (integer)	**Optional POST/GET Vars:** c_id (integer) appt_id (integer) date (date: yyyymmdd) start_time (time [in minutes])
Description: Unreserves the specified time slot, or deletes the specified reserved time. Once deleted, the record cannot be restored. You must specify either an employee_id or room_id, along with either an appt_id or both a date and start_time in order properly identify what time should be unreserved. If you do not specify a c_id (location_id for the location of the reserved time), the method will assume the headquarters location.	**Returns**: appt_id, c_id, employee_id, room_id, date, start_time, reason	

The method will return the details of the reserved time that was removed. **NOTE**: A “reserved time” is simply a special type of appointment.	

CancelAppointments		
URI: /Appointments/CancelAppointments		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) appt_id (integer)	**Optional POST/GET Vars:** reason (text) last_emp_id (integer)
Description: Cancels the specified appointment. Once canceled, the appointment will be removed from the appointment grid, but the record will still be available in the customer’s appointment history, showing the canceled status. You may optionally provide a ‘reason’ for the cancellation. You may also specify the staff member performing the cancellation using *last_emp_id*.	**Returns**: c_id, appt_id, customer_id, account, employee_id, room_id, date, start_time, end_time, service_id, customer_notes, employee_notes, status_id, rep_id, cost, tip, payment_type_id, spots, reason, creation_emp_id, last_timestamp, last_emp_id, coupon_code	

Events

GetEvents		
URI: /Events/GetEvents		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp)	**Optional POST/GET Vars:** c_id (integer, multiple) service_id (integer, multiple) event_template_id (integer, multiple) room_id (integer, multiple) employee_id (integer, multiple) hide_unavailable (boolean) include_time_zone (boolean) start_date end_date
Description: Returns the details of all events or a single event specified by service_id. If you do not specify a c_id (location_id for the location of the event), the method will return events from all locations. For POST/GET variables that accept multiple values, separate values with a comma (e.g. *service_id=1331,2442,3553*). Use the start_date and end_date parameters to filter the resultset to only include events that have dates within the specified range. Use the hide_unavailable parameter to only include events that have available future dates. Valid values for event_date.status are 0 = active, 1 = cancelled.	**Returns**: c_id, service_id, event_template_id, title, description, default_service_time, default_cost, sort_order, event_time, max_participants, employee_id, room_id, display, multi_day, product_code, taxable, category_id, event_dates[event_date[date, participants, spots_remaining, status, cancel_reason_id]]	

CreateEvents		
URI: /Events/CreateEvents		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) c_id (integer) title (text) display (yes/no) duration (integer) event_dates (date, multiple) start_time (ineger) span_dates (yes/no)	**Optional POST/GET Vars:** event_template_id (integer) employee_id (integer) room_id (integer) description (text) cost (decimal) max_participants (integer) sort_order (integer) category_id (integer) code (text) taxable (yes/no)

Description: Creates a new event, either standalone or from an event template. If the event_template_id parameter is included, the method will override the following parameters and use the defaults from the template: *description*, *display*, *category_id*, *code*, *span_dates*.	**Returns**: service_id, c_id, event_template_id, title, description, display, cost, duration, max_participants, event_dates, start_time, sort_order, category_id, code, taxable, employee_id, room_id

GetParticipants		
URI: /Events/GetParticipants		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) service_id (integer)	**Optional POST/GET Vars:** c_id (integer) date (date: yyyymmdd) customer_id (integer) event_template_id (integer)
Description: Returns the details of all participants of a single event specified by service_id. If the event is not a multi-day event, then the *date* parameter becomes required. If using the event_template_id parameter, the service_id parameter is optional.	**Returns**: c_id, event_template_id, appt_id, date, start_time, end_time, spots, status_id, employee_id, cost, customer_notes, employee_notes, creation_timestamp, creation_emp_id, last_timestamp, last_emp_id, customer_id, first_name, middle_name, last_name, email, account	

CreateParticipants		
URI: /Events/CreateParticipants		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) service_id (integer) customer_id (integer)	**Optional POST/GET Vars:** c_id (integer) date (date: yyyymmdd) spots (integer) cost (decimal) tip (decimal) payment_type_id (integer) special_instructions (text) employee_notes (text) status_id (integer) customer_package_id(integer)
Description: If the event is not a multi-day event, then the *date* parameter becomes required.	**Returns**: c_id, appt_id, date, start_time, end_time, spots, status_id, employee_id, cost, customer_notes, employee_notes, creation_timestamp, creation_emp_id, last_timestamp, last_emp_id, customer_id, first_name, middle_name, last_name, email, account	

UpdateParticipants		
URI: /Events/UpdateParticipants		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) service_id (integer) customer_id (integer)	**Optional POST/GET Vars:** date (date: yyyymmdd) status_id (integer) spots (integer) cost (decimal) special_instructions (text) employee_notes (text)

Description: If the event is not a multi-day event, then the *date* parameter becomes required.	**Returns**: c_id, appt_id, date, start_time, end_time, spots, status_id, employee_id, cost, customer_notes, employee_notes, creation_timestamp, creation_emp_id, last_timestamp, last_emp_id, customer_id, first_name, middle_name, last_name, email, account

DeleteParticipants		
URI: /Events/DeleteParticipants		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) service_id (integer) customer_id (integer)	**Optional POST/GET Vars:** date (date: yyyymmdd)

Description: Removes a participant from an event (cancels the appointments). If the event is not a multi-day event, then the *date* parameter becomes required.	**Returns**: c_id, appt_id, date, start_time, end_time, spots, status_id, employee_id, cost, customer_notes, employee_notes, creation_timestamp, creation_emp_id, last_timestamp, last_emp_id, customer_id, first_name, middle_name, last_name, email, account

GetEventTemplates		
URI: /Events/GetEventTemplates		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp)	**Optional POST/GET Vars:** c_id (integer, multiple) event_template_id (integer, multiple) name (text) status (integer, multiple)

Description: Returns details of event templates. You can specify multiple c_id's or multiple event_template_id's to filter the response. Otherwise, the method will return all event templates from all locations. Unless specified with the *status* parameter, the	**Returns**: event_template_id, c_id, name, sort_order, status, event_title, event_display, event_description, event_category_id, event_code, event_multi_day

method will return deleted and inactive event templates. Values for status are *1 = active*, *2 = inactive, 3 = deleted*.	

CreateEventTemplates

URI: /Events/CreateEventTemplates

Required HTTP Headers:	**Required POST/GET Vars**:	**Optional POST/GET Vars:**
Authorizaton (Basic site_id:api_key)	response_type (xml | json | jsonp) c_id (integer) name (text) sort_order (integer) event_title (text) event_display (yes/no) event_multi_day (yes/no)	event_description (text) event_category_id (integer) event_code (text)

Description:	**Returns**:
Creates a new event template. The *name* parameter is the name of the event template, not the name to be applied to the event instances. The *event_** parameters are used to set the default parameters of the event instances. Values for status are *1 = active*, *2 = inactive, 3 = deleted*.	event_template_id, c_id, name, sort_order, status, event_title, event_display, event_description, event_category_id, event_code, event_multi_day

UpdateEventTemplates

URI: /Events/UpdateEventTemplates

Required HTTP Headers:	**Required POST/GET Vars**:	**Optional POST/GET Vars:**
Authorizaton (Basic site_id:api_key)	response_type (xml | json | jsonp) event_template_id (integer, multiple)	name (text) sort_order (integer) status (integer) event_title (text) event_display (yes/no) event_multi_day (yes/no) event_description (text) event_category_id (integer) event_code (text)

Description:	**Returns**:
Pass a comma separated list for event_template_id to update multiple event templates.	event_template_id, c_id, name, sort_order, status, event_title, event_display, event_description, event_category_id, event_code, event_multi_day

DeleteEventTemplates

URI: /Events/DeleteEventTemplates

Required HTTP Headers:	**Required POST/GET Vars**:	**Optional POST/GET Vars:**
Authorizaton (Basic site_id:api_key)	response_type (xml | json | jsonp)	

	event_template_id (integer, multiple)	

Description: Pass a comma separated list for event_template_id to remove multiple event templates. This method will not allow deleting event templates that have event instances assigned.	**Returns**: event_template_id, c_id, name, sort_order, status, event_title, event_display, event_description, event_category_id, event_code, event_multi_day

CancelEventDates		
URI: /Events/CancelEventDates		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) service_id (integer, multiple) dates (date, multiple)	**Optional POST/GET Vars:** cancel_reason_id (integer)
Description: Cancels dates within an event. Passing a comma separated list of service_id's will remove the same dates from each event, and apply the same cancel reason to each date in each event.	**Returns**: service_id, c_id, event_template_id, title, description, display, cost, duration, max_participants, event_dates, start_time, sort_order, category_id, code, taxable, employee_id, room_id	

GetCancelReasons		
URI: /Events/GetCancelReasons		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp)	**Optional POST/GET Vars:** c_id (integer)
Description: Returns the predefined event date cancellation reasons.	**Returns**: cancel_reason_id, c_id, description, sort_order	

Waiting List

GetList			
URI: /WaitingList/GetList			
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp)	**Optional POST/GET Vars:**	
Description: Returns all customers in the waiting list.		**Returns**: c_id, status_id, sort_order, description, status_type, color, display	

AddToList			
URI: /WaitingList/AddToList			
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) customer_id (integer)	**Optional POST/GET Vars:** c_id (integer) notes (text)	
Description: Adds a customer to the waiting list and returns the details of the waiting list entry. If you do not send the c_id, the customer will be added to the waiting list for the headquarter location.		**Returns**: wait_id, c_id, customer_id, date, time, notes	

Customers

GetCustomers		
URI: /Customers/GetCustomers		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp)	**Optional POST/GET Vars:** customer (integer,multiple) location (integer,multiple) first_name (text, multiple) last_name (text, multiple) first_name_search (text) last_name_search (text) account (text,multiple) email (text,multiple) status_id (integer,multiple) updated (timestamp)

Description: Returns the details for customers. The returned dataset can be filtered by sending any of the optional POST/GET variables. All text variables are case-insensitive. For POST/GET variables that accept multiple values, separate values with a comma (e.g. *location=1331,2442,3553&last_name=brown,Smith*). If none of the optional POST/GET variables are sent, the service will return all of the customers for every location. For first_name_search and last_name_search, you may enter partial search strings to find customers whose last or first names start with that string (e.g. *first_name_search=Jo*). Use the *updated* parameter to find customers that were created or updated after the specified timestamp. The *updated* parameter will accept any valid format for date/time. Unless otherwise specified in the value provided, it will be assumed that the timestamp provided is in the time zone of the location.	**Returns**: c_id, customer_id, last_name, first_name, middle_name, address1, address2, city, state, zip, day_phone, night_phone, cell_phone, email, login, notes, signup_date, customer_type_id, contact_okay, call_okay, email_okay, mail_okay, payment_type_id, last_update_id, last_update, lead_id, rep_id, employer, occupation, birth_date, gender, status_id, allow_login, account, needs, name_title, referred_by, alert, timezone_id, last_login_update, last_pw_update, last_info_prompt, location_id

CreateCustomers		
URI: /Customers/CreateCustomers		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) c_id (integer) last_name (text) first_name (text)	**Optional POST/GET Vars:** employee_id (integer) middle_name (text) address1 (text) address2 (text) city (text)

		state (text) zip (text) day_phone (text) night_phone (text) cell_phone (text) fax (text) email (text) other_address1 (text) other_address2 (text) other_city (text) other_state (text) other_zip (text) other_phone1 (text) other_phone2 (text) login (text, unique) password (text) signup_date (date: yyyymmdd) customer_type_id (integer) contact_okay (yes | no) call_okay (yes | no) email_okay (yes | no) mail_okay (yes | no) payment_type_id (integer) lead_id (integer) rep_id (integer) employer (text) occupation (text) birth_date (date: yyyymmdd) birth_date_noyear (date: mmdd) gender (text) status_id (integer) allow_login (yes | no) dropdown1_type_id (integer) dropdown2_type_id (integer) dropdown3_type_id (integer) account (text, unique) needs (text) name_title (text) referred_by (text) alert (text) carrier_id (integer) notification_id (integer) timezone_id (integer)
Description: Creates new customers and returns the details for the new customers. The required c_id field should be the location_id of the location where the customer will be added. Note that both the login and account fields should be	**Returns**: c_id, customer_id, last_name, first_name, middle_name, day_phone, email, login, account	

unique. If duplicate entries are found, the record will not be added and an error will be thrown.	

UpdateCustomers		
URI: /Customers/UpdateCustomers		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) customer_id (integer)	**Optional POST/GET Vars:** c_id (integer) employee_id (integer) middle_name (text) address1 (text) address2 (text) city (text) state (text) zip (text) day_phone (text) night_phone (text) cell_phone (text) fax (text) email (text) other_address1 (text) other_address2 (text) other_city (text) other_state (text) other_zip (text) other_phone1 (text) other_phone2 (text) login (text) password (text) signup_date (date: yyyymmdd) customer_type_id (integer) contact_okay (yes | no) call_okay (yes | no) email_okay (yes | no) mail_okay (yes | no) payment_type_id (integer) lead_id (integer) rep_id (integer) employer (text) occupation (text) birth_date (date: yyyymmdd) birth_date_noyear (date: mmdd) gender (text) status_id (integer) allow_login (yes | no) account (text, unique) needs (text) name_title (text) referred_by (text) alert (text) carrier_id (integer)

		notification_id (integer) timezone_id (integer)
Description: Creates new customers and returns the details for the new customers. Note that both the login and account fields should be unique. If duplicate entries are found, the record will not be added and an error will be thrown.	**Returns**: c_id, customer_id, last_name, first_name, middle_name, day_phone, email, login, account	

DeleteCustomers		
URI: /Customers/DeleteCustomers		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) customer_id (integer)	**Optional POST/GET Vars:**
Description: Deletes a customer along with their appointments, notes, pets/children, waiting list entries, assigned packages, and future SMS reminders. Once deleted, the customer record cannot be restored.	**Returns**: c_id, customer_id, last_name, first_name, middle_name, email, account	

GetFutureAppointments		
URI: /Customers/GetFutureAppointments		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) customer_id (integer)	**Optional POST/GET Vars:**
Description: Returns the number of appointments and waiting list entries for the specified customer.	**Returns**: customer_id, appointments, waiting_list	

GetPackages		
URI: /Customers/GetPackages		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp)	**Optional POST/GET Vars:** customer_id (integer) event_template_id (integer, multiple) service_id (integer, multiple) updated (date/time)
Description: Returns all packages assigned to the specified customer, or for all customers. Use the *updated* parameter to find customers that were created or updated after the specified timestamp. The *updated* parameter will accept any valid format for date/time. Unless otherwise specified in the value provided, it will be assumed that the timestamp	**Returns**: customer_id, customer_package_id, item_id, creation_date, customer_price, per_session_cost, monthly_fee, num_sessions, num_days, enrollment_fee, down_payment, status, expiration_date	

provided is in the time zone of the location.	

UpdatePackages		
URI: /Customers/UpdatePackages		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) customer_package_id (integer) customer_id (integer)	**Optional POST/GET Vars:** monthly_fee (decimal) num_sessions (integer) num_days (integer) per_session_cost (decimal) down_payment (decimal) enrollment_fee (decimal) creation_date (date: yyyymmdd) expiration_date (date: yyyymmdd)
Description: Updates details of the specified package assigned to a customer.	**Returns**: customer_id, customer_package_id, item_id, creation_date, customer_price, per_session_cost, monthly_fee, num_sessions, num_days, enrollment_fee, down_payment, status, expiration_date, package_services, package_statuses	

GetCustomerTypes		
URI: /Customers/GetCustomerTypes		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp)	**Optional POST/GET Vars:**
Description: Returns all customer types	**Returns**: customer_type_id, c_id, description, sort_order, customer_type	

GetCustomerStatuses		
URI: /Customers/GetCustomerStatuses		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp)	**Optional POST/GET Vars:**
Description: Returns all customer statuses	**Returns**: status_id, c_id, description, sort_order, status_type	

GetLeadTypes		
URI: /Customers/GetLeadTypes		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp)	**Optional POST/GET Vars:**
Description: Returns all of the details for the existing Customer Lead Types.	**Returns**: lead_id, c_id, description, sort_order	

CreateCustomerTypes		
URI: /Customers/CreateCustomerTypes		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) description (text)	**Optional POST/GET Vars:** sort_order (integer)
Description: Creates new customer types and returns all the details for the new customer types	**Returns**: customer_type_id, c_id, description, sort_order	

CreateLeadTypes		
URI: /Customers/CreateLeadTypes		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) description (text)	**Optional POST/GET Vars:** sort_order (integer)
Description: Creates new Lead Types and returns all of the details for the new Lead Types.	**Returns**: lead_id, c_id, description, sort_order	

GetChildren		
URI: /Customers/GetChildren		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) customer_id (integer)	**Optional POST/GET Vars:** child_id (integer, multiple) first_name (string) last_name (string) first_name_search (string) last_name_search (string)
Description: Returns the details for children of the specified customer. The returned dataset can be filtered by sending any of the optional POST/GET variables. All text variables are case-insensitive. For POST/GET variables that accept multiple values, separate values with a comma (e.g. *location=1331,2442,3553&last_name=brown,Smith*). If none of the optional POST/GET variables are sent, the service will return all of the children for the specified customer. For first_name_search and last_name_search, you may enter partial search strings to find children whose last or first names start with that string (e.g. *first_name_search=Jo*).	**Returns**: c_id, customer_id, child_id, first_name, last_name, middle_name, birth_date, notes, age	

CreateChildren		
URI: /Customers/CreateChildren		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) customer_id (integer) first_name (string)	**Optional POST/GET Vars:** last_name (string) middle_name (string) birth_date (date: yyyymmdd) notes (text) age (string)
Description: Creates new children and returns the details for the new children. You must specify the customer to whom the child will belong.	**Returns**: c_id, customer_id, child_id, first_name, last_name, middle_name, birth_date, notes, age	

UpdateChildren		
URI: /Customers/UpdateChildren		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) customer_id (integer) child_id (integer)	**Optional POST/GET Vars:** first_name (string) last_name (string) middle_name (string) birth_date (date: yyyymmdd) notes (text) age (string)
Description: Updates details of a single or multiple (using XML) children. Only the fields/values you send to the service will be updated, so you may update any number and combination of fields. If you send a field with no value (empty string), the system will assume you want to clear the value that is currently in the field. You cannot change the parent customer or the location (c_id) of the child.	**Returns**: c_id, customer_id, child_id, first_name, last_name, middle_name, birth_date, notes, age	

DeleteChildren		
URI: /Customers/DeleteChildren		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) customer_id (integer)	**Optional POST/GET Vars:** upload_xml (XMLstring)

	child_id (integer)	

Description: Delete a single or multiple (using XML) children. Once deleted, the child record cannot be restored. Deleting a child will not remove its associated appointments.	**Returns**: c_id, customer_id, child_id, first_name, last_name, middle_name, birth_date, notes, age

GetPets

URI: /Customers/GetPets

Required HTTP Headers:	**Required POST/GET Vars**:	**Optional POST/GET Vars:**
Authorizaton (Basic site_id:api_key)	response_type (xml | json | jsonp) customer_id (integer)	pet_id (integer, multiple) pet_name (string) breed (string) pet_name_search (string) breed_search (string)

Description: Returns the details for pets of the specified customer. The returned dataset can be filtered by sending any of the optional POST/GET variables. All text variables are case-insensitive. For POST/GET variables that accept multiple values, separate values with a comma (e.g. *location=1331,2442,3553&last_name=brown,Smith*). If none of the optional POST/GET variables are sent, the service will return all of the pets for the specified customer. For pet_name_search and breed_search, you may enter partial search strings to find pets whose name or breed starts with that string (e.g. *pet_name_search=Spar&breed_search=terrier*).	**Returns**: c_id, customer_id, pet_id, pet_name, type_id, breed, weight, color, gender, age, notes, vac_info, health_info, vet_info

CreatePets

URI: /Customers/CreatePets

Required HTTP Headers:	**Required POST/GET Vars**:	**Optional POST/GET Vars:**
Authorizaton (Basic site_id:api_key)	response_type (xml | json | jsonp) customer_id (integer) pet_name (string)	type_id (integer) breed (string) weight (string) color (string) gender (string) age (string) notes (text) vac_info (text) health_info (text) vet_info (text)

Description: Creates new pets and returns the details for the new pets. You must specify the customer to whom the pet will belong.	**Returns**: c_id, customer_id, pet_id, pet_name, type_id, breed, weight, color, gender, age, notes, vac_info, health_info, vet_info

UpdatePets		
URI: /Customers/UpdatePets		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) customer_id (integer) pet_id (integer)	**Optional POST/GET Vars:** pet_name (string) type_id (integer) breed (string) weight (string) color (string) gender (string) age (string) notes (text) vac_info (text) health_info (text) vet_info (text)
Description: Updates details of a single or multiple (using XML) pets. Only the fields/values you send to the service will be updated, so you may update any number and combination of fields. If you send a field with no value (empty string), the system will assume you want to clear the value that is currently in the field. You cannot change the parent customer or the location (c_id) of the pet.	**Returns**: c_id, customer_id, pet_id, pet_name, type_id, breed, weight, color, gender, age, notes, vac_info, health_info, vet_info	

DeletePets		
URI: /Customers/DeletePets		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) customer_id (integer) pet_id (integer)	**Optional POST/GET Vars:**
Description: Delete a single or multiple (using XML) pets. Once deleted, the pet record cannot be restored. Deleting a pet will not remove its associated appointments.	**Returns**: c_id, customer_id, pet_id, pet_name, type_id, breed, weight, color, gender, age, notes, vac_info, health_info, vet_info	

Pets

GetPetTypes			
URI: /Pets/GetPetTypes			
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp)	**Optional POST/GET Vars:**	
Description: Returns all of the details for the existing Pet Types.		**Returns**: lead_id, c_id, description, sort_order	

Staff

GetStaff		
URI: /Staff/GetStaff		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp)	**Optional POST/GET Vars:** staff (integer, multiple) location (integer) first_name (text, multiple) last_name (text, multiple) first_name_search (text) last_name_search (text) screen_name_search (text) email (text, multiple) status (text, multiple) show_deleted (yes | no)

Description: Returns the details for staff members. The returned dataset can be filtered by sending any of the optional POST/GET variables. All string variables are case-insensitive. For POST/GET variables that accept multiple values, separate values with a comma (e.g. *location=1331,2442,3553&last_name=brown,Smith*). If none of the optional POST/GET variables are sent, the service will return all of the staff members for every location. For first_name_search, last_name_search and screen_name_search, you may enter partial search strings to find customers whose last, first or screen names start with that string (e.g. *first_name_search*=*Jo*). If the location you specify is configured to use the Multiple Appointments version of the application, the additional scheduling fields will not be returned. You will need to use the Staff/GetSchedule service to get the staff member's schedule. By default the method will not return deleted staff members. Using the show_deleted flag will also return deleted staff members.	**Returns**: c_id, employee_id, last_name, first_name, middle_name, screen_name, company, picture_link, bio, notes, login, status, user_group, email, type_id, display, sort_order, relay1_id, relay2_id, home_phone, work_phone, cell_phone, fax, sales_person_id, tax_number, address, city, state, zip, hide_views, carrier_id, timezone_id, entity_type_id, bill_rate, location_id ***If the account is configured to use the Single Appointments version and is not using Staff Sharing; OR if the account is configured to use the Single Appointments version and is using staff sharing but a location is specified, the following fields are also returned:*** waiting_list, first_appt_time_Monday, last_appt_time_Monday, off_Monday, first_appt_time_Tuesday, last_appt_time_Tuesday, off_Tuesday, first_appt_time_Wednesday, last_appt_time_Wednesday, off_Wednesday, first_appt_time_Thursday, last_appt_time_Thursday, off_Thursday, first_appt_time_Friday, last_appt_time_Friday, off_Friday, first_appt_time_Saturday, last_appt_time_Saturday, off_Saturday, first_appt_time_Sunday, last_appt_time_Sunday, off_Sunday, appt_start_times, email_reminder_advance, display_for_no_pref, appt_advance, location_id

CreateStaff

URI: /Staff/CreateStaff

Required HTTP Headers:	**Required POST/GET Vars**:	**Optional POST/GET Vars:**
Authorizaton (Basic site_id:api_key)	response_type (xml | json | jsonp) c_id (integer) screen_name (text)	last_name (text) first_name (text) middle_name (text) company (text) bio (text) notes (text) login (text) password (text) temp_password (yes | no) user_group (text, see notes) email (text) type_id (integer) display (yes | no) sort_order (integer) home_phone (text) work_phone (text) cell_phone (text) fax (text) tax_number (text) address (text) city (text) state (text) zip (text) timezone_id (integer) bill_rate (decimal)

Description:	**Returns**:
Creates a new Staff Member and returns all of the details for the new Staff Member. The staff member is created with a *status* of 'Inactive', meaning it has no schedule and is not available for appointments. Use Staff/CreateSchedules to create the staff members schedule, then use Staff/UpdateStaff to update the staff member's *status* to 'Active'. The c_id parameter should be set to the location_id of the location where the new staff member should be created. The user_group parameter sets the access level for the new staff member. Note that this does not currently support Custom Access Types. Valid values include: 'ha' – Headquarters Administration 'a' – Location Administrator 'c' – Call Center User	c_id, employee_id, last_name, first_name, middle_name, screen_name, company, bio, notes, login, temp_password, user_group, email, type_id, display, sort_order, home_phone, work_phone, cell_phone, fax, tax_number, address, city, state, zip, timezone_id, bill_rate

'l' – Location User (lowercase 'L') 'v' – View Only User	

UpdateStaff

URI: /Staff/UpdateStaff

Required HTTP Headers:	**Required POST/GET Vars**:	**Optional POST/GET Vars:**
Authorizaton (Basic site_id:api_key)	response_type (xml | json | jsonp) employee_id (integer)	screen_name (text) last_name (text) first_name (text) middle_name (text) company (text) bio (text) notes (text) login (text) password (text) temp_password (yes | no) user_group (text, see notes) email (text) type_id (integer) display (yes | no) sort_order (integer) home_phone (text) work_phone (text) cell_phone (text) fax (text) tax_number (text) address (text) city (text) state (text) zip (text) timezone_id (integer) bill_rate (decimal) status (text)

Description:	**Returns**:
Updates a Staff Member and returns all of the details for the updated Staff Member. The user_group parameter sets the access level for the new staff member. Note that this does not currently support Custom Access Types. Valid values include: 'ha' – Headquarters Administration 'a' – Location Administrator 'c' – Call Center User 'l' – Location User (lowercase 'L') 'v' – View Only User Valid values for status are "active" or "Inactive". Active staff members are available for scheduling appointments; Inactive staff members cannot be	c_id, employee_id, last_name, first_name, middle_name, screen_name, company, bio, notes, login, temp_password, user_group, email, type_id, display, sort_order, home_phone, work_phone, cell_phone, fax, tax_number, address, city, state, zip, timezone_id, bill_rate

booked for appointments.	

DeleteStaff		
URI: /Staff/DeleteStaff		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) employee_id (integer)	**Optional POST/GET Vars:**
Description: Deletes a Staff Member and returns all of the details for the updated Staff Member. Once deleted, staff members can be recovered; however, any information about the staff member (schedule parameters, assigned rooms, services, and/or locations, etc) cannot be recovered.	**Returns**: c_id, employee_id, last_name, first_name, middle_name, screen_name, company, bio, notes, login, temp_password, user_group, email, type_id, display, sort_order, home_phone, work_phone, cell_phone, fax, tax_number, address, city, state, zip, timezone_id, bill_rate	

GetSchedules		
URI: /Staff/GetSchedules		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) employee_id (integer) c_id (integer)	**Optional POST/GET Vars:** weekday (integer)
Description: Returns the regular schedule parameters for a staff member at a specified location. This function does not return the details for overrides for specific dates, i.e. scheduling templates, exceptions, days off, etc. The returned dataset can be filtered by sending any of the optional POST/GET variables. The *weekday* filter only applies to accounts using the Multiple Appointments version of the application. When sending *weekday*, use integers 1 through 7 to represent Sunday through Saturday. Using this filter will only return the schedule for that day of the week.	**Returns**: ***If the account is configured to use the Multiple Appointments version of the application:*** location_id, employee_id, day, start_time, end_time, num_appts_per_slot ***If the account is configured to use the Single Appointments version of the application:*** location_id, employee_id, first_appt_time_Monday, last_appt_time_Monday, off_Monday, first_appt_time_Tuesday, last_appt_time_Tuesday, off_Tuesday, first_appt_time_Wednesday, last_appt_time_Wednesday, off_Wednesday, first_appt_time_Thursday, last_appt_time_Thursday, off_Thursday, first_appt_time_Friday, last_appt_time_Friday, off_Friday, first_appt_time_Saturday, last_appt_time_Saturday, off_Saturday, first_appt_time_Sunday, last_appt_time_Sunday, off_Sunday	

CreateSchedules		
URI: /Staff/CreateSchedules		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) employee (integer, multiple) c_id (integer, multiple) ***Multiple Appointments:*** weekday (integer, mutiple) start_time (time [in minutes], multiple) num_appts_per_slot (integer)	**Optional POST/GET Vars:** ***Single Appointments:*** first_appt_time_sunday (time [in minutes]) last_appt_time_sunday (time [in minutes]) first_appt_time_monday (time [in minutes]) last_appt_time_monday (time [in minutes]) first_appt_time_tuesday (time [in minutes]) last_appt_time_tuesday (time [in minutes]) first_appt_time_wednesday (time [in minutes]) last_appt_time_wednesday (time [in minutes]) first_appt_time_thursday (time [in minutes]) last_appt_time_thursday (time [in minutes]) first_appt_time_friday (time [in minutes]) last_appt_time_friday (time [in minutes]) first_appt_time_saturday (time [in minutes]) last_appt_time_saturday (time [in minutes]) ***Multiple Appointments:*** duration (integer) end_time (time [in minutes])

Description:	**Returns**:
Creates a regular schedule for the specified employees at the specified locations. Send multiple values in *c_id* and/or multiple values in *employee_id* to create the same schedule across multiple locations and/or staff members. ***If the account is configured to use the Single Appointments version:*** You must set the schedule for the entire week at once. Send the open and close times for each day of the week. If *first_appt_time_* and *last_appt_time_*	***Single Appointments:*** c_id, employee_id, first_appt_time_Sunday, last_appt_time_Sunday, off_Sunday, first_appt_time_Monday, last_appt_time_Monday, off_Monday, first_appt_time_Tuesday, last_appt_time_Tuesday, off_Tuesday, first_appt_time_Wednesday, last_appt_time_Wednesday, off_Wednesday, first_appt_time_Thursday, last_appt_time_Thursday, off_Thursday, first_appt_time_Friday, last_appt_time_Friday, off_Friday,

parameters are not sent for a day of the week or if an empty value is sent, the method assumes that day is an off day in the regular schedule. ***If the account is configured to use the Multiple Appointments version:*** You can set the schedule parameters for individual time slots and days. Send multiple values in *weekday* and/or *start_time* to create the same start times across multiple days (even when specifying multiple locations and employees). Either *end_time* or *duration* must be used to specify the end time of the time slots. If sending multiple values in *start_time*, you must use *duration*. All time slots created will have the same duration. When sending *weekday*, use integers 1 through 7 to represent Sunday through Saturday.	first_appt_time_Saturday, last_appt_time_Saturday, off_Saturday ***Multiple Appointments:*** c_id, employee_id, day, start_time, end_time, num_appts_per_slot

UpdateSchedules		
URI: /Staff/UpdateSchedules		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) employee (integer, multiple) c_id (integer, multiple) ***Multiple Appointments:*** weekday (integer, mutiple) start_time (time [in minutes], multiple) num_appts_per_slot (integer)	**Optional POST/GET Vars:** ***Single Appointments:*** first_appt_time_sunday (time [in minutes]) last_appt_time_sunday (time [in minutes]) first_appt_time_monday (time [in minutes]) last_appt_time_monday (time [in minutes]) first_appt_time_tuesday (time [in minutes]) last_appt_time_tuesday (time [in minutes]) first_appt_time_wednesday (time [in minutes]) last_appt_time_wednesday (time [in minutes]) first_appt_time_thursday (time [in minutes]) last_appt_time_thursday (time [in minutes]) first_appt_time_friday (time [in minutes]) last_appt_time_friday (time [in minutes]) first_appt_time_saturday (time [in

		minutes]) last_appt_time_saturday (time [in minutes])

Description: Updates a regular schedule for the specified employees at the specified locations. Send multiple values in *c_id* and/or multiple values in *employee_id* to update multiple schedules across multiple locations and/or staff members. ***If the account is configured to use the Single Appointments version:*** You may update the entire week's schedule or only specific days. If the *first_appt_time_* or *last_appt_time_* parameters are not sent for any of the weekdays, the schedule will remain the same for those days. If an empty value is sent for *first_appt_time_* and *last_appt_time_* parameters, the method assumes that day is an off day in the regular schedule. ***If the account is configured to use the Multiple Appointments version:*** You can update multiple time slots and/or multiple days in a schedule by sending multiple values in *start_time*and/or *weekday*. The only schedule parameter that can be modified after the schedule is created is the number of appointments that the slot can hold (*num_appts_per_slot*). To shorten/extend a time slot, you will need to delete it first then recreate it. When sending *weekday*, use integers 1 through 7 to represent Sunday through Saturday.	**Returns**: ***Single Appointments:*** c_id, employee_id, first_appt_time_Sunday, last_appt_time_Sunday, off_Sunday, first_appt_time_Monday, last_appt_time_Monday, off_Monday, first_appt_time_Tuesday, last_appt_time_Tuesday, off_Tuesday, first_appt_time_Wednesday, last_appt_time_Wednesday, off_Wednesday, first_appt_time_Thursday, last_appt_time_Thursday, off_Thursday, first_appt_time_Friday, last_appt_time_Friday, off_Friday, first_appt_time_Saturday, last_appt_time_Saturday, off_Saturday ***Multiple Appointments:*** c_id, employee_id, day, start_time, end_time, num_appts_per_slot

DeleteSchedules		
URI: /Staff/DeleteSchedules		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) employee (integer, multiple) c_id (integer, multiple)	**Optional POST/GET Vars:** weekday (integer, multiple) start_time (time [in minutes], multiple)
Description: Deletes the regular schedule for the specified employees at the specified locations. This action does not affect templates, exceptions, or days off.	**Returns**: *No data is returned for this method*	

The *weekday* and *start*_time filters only apply to accounts using the Multiple Appointments version of the application. When sending *weekday*, use integers 1 through 7 to represent Sunday through Saturday. Using this filter will only return the schedule for that day of the week.	

GetStaffTypes		
URI: /Staff/GetStaffTypes		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp)	**Optional POST/GET Vars:**
Description: Returns all staff types	**Returns**: type_id, c_id, description, sort_order	

CreateStaffTypes		
URI: /Staff/CreateStaffTypes		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) description (text)	**Optional POST/GET Vars:** sort_order (integer)
Description: Creates new Staff Types and returns all of the details for the new Staff Types.	**Returns**: type_id, c_id, description, sort_order	

GetServicesOffered		
URI: /Staff/GetServicesOffered		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) employee_id (integer)	**Optional POST/GET Vars:** location_id (integer)
Description: Returns the details for the services offered by (assigned to) the specified staff member. The returned dataset can be filtered by sending any of the optional POST/GET variables. If none of the optional POST/GET variables are sent, the service will return all of the assigned services at each location the staff member is assigned to. The location_id parameter is only effective when the account is configured to share staff members across locations. Otherwise, the service returns the services assigned at the staff member's home location.	**Returns**: location_id, service_id, addon, employee_id, employee_service_time, avail_Mon, avail_Tue, avail_Wed, avail_Thu, avail_Fri, avail_Sat, avail_Sun, employee_cost ***If limiting service availability by staff member, the following fields are also returned:*** first_time_Monday, last_time_Monday, first_time_Tuesday, last_time_Tuesday, first_time_Wednesday, last_time_Wednesday, first_time_Thursday, last_time_Thursday, first_time_Friday, last_time_Friday, first_time_Saturday, last_time_Saturday, first_time_Sunday, last_time_Sunday	

AddServicesOffered		
URI: /Staff/AddServicesOffered		
Required HTTP Headers:	**Required POST/GET Vars**:	**Optional POST/GET Vars:**

Authorizaton (Basic site_id:api_key)	response_type (xml | json | jsonp) employee_id (integer)	location_id (integer)

Description: Assigns a service to a staff member and additionally specifies the hours of availability, cost, and duration of the service.	**Returns**: location_id, service_id, employee_id, employee_service_time, avail_Mon, avail_Tue, avail_Wed, avail_Thu, avail_Fri, avail_Sat, avail_Sun, cost

GetOpenDates		
URI: /Staff/GetOpenDates		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) employee_id (integer) c_id (integer) num_days (integer)	**Optional POST/GET Vars:** start_date (date: yyyymmdd)
Description: Returns the dates on which the employee is available for scheduling, based on the employee's regular schedule, templates, exceptions, and days off settings. The method will return all available dates between start_date + num_days. If start_date is not specified, it will default to today's date.	**Returns**: c_id, employee_id, date, month, day, year, day_name	

GetScheduleExceptions		
URI: /Staff/GetScheduleExceptions		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) employee_id (integer) c_id (integer)	**Optional POST/GET Vars:** date (date: yyyymmdd)) start_time (time [in minutes]) exception_id (integer)
Description: Get a staff member's schedule exceptions for a particular location, and optionally, a particular date. Returns all of the matching exceptions, if any exist.	**Returns**: exception_id,c_id, employee_id, date, start_time, end_time, spots	

CreateScheduleExceptions		
URI: /Staff/CreateScheduleExceptions		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) employee_id (integer) c_id (integer) date (date: yyyymmdd)) start_time (time [in minutes]) end_time (time [in minutes]) spots (integer)	**Optional POST/GET Vars:**
Description:	**Returns**:	

Creates a staff member schedule exception at a particular location, date, and time	exception_id, c_id, employee_id, date, start_time, end_time, spots

DeleteScheduleExceptions		
URI: /Staff/DeleteScheduleExceptions		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) employee_id (integer) c_id (integer) date (date: yyyymmdd)) start_time (time [in minutes]) end_time (time [in minutes])	**Optional POST/GET Vars:** exception_id (integer)
Description: Deletes a staff member schedule exception. This cannot be undone.	**Returns**: exception_id, location_id, employee_id, date, start_time, end_time	

GetDaysOff		
URI: /Staff/GetDaysOff		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) c_id (integer, multiple) employee_id (integer) start_date (date: yyyymmdd) end_date (date: yyyymmdd)	**Optional POST/GET Vars:**
Description: Returns the dates between the start_date and end_date on which the staff member is unavailable, based on the Days Off settings. The method will accept a list of multiple, comma separated locations (c_id) and return the dates for each of the locations.	**Returns**: c_id, employee_id, date	

DeleteDaysOff		
URI: /Staff/DeleteDaysOff		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) c_id (integer, multiple) employee_id (integer, multiple) date (date: yyyymmdd, multiple)	**Optional POST/GET Vars:**
Description: Removes days off from the staff member's calendar (i.e. reopens those days for scheduling). The method will accept a list of multiple, comma separated locations (c_id), multiple staff members, and multiple dates, and will remove each of the dates	**Returns**: c_id, employee_id, date	

for each of the staff members at each of the locations.	

CreateDaysOff		
URI: /Locations/CreateDaysOff		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) c_id (integer, multiple) employee_id (integer, multiple) date (date: yyyymmdd, multiple)	**Optional POST/GET Vars:**
Description: Marks date(s) as being a day off on the location's calendar (i.e. closes the day for scheduling). The method will accept multiple dates and multiple locations. It will mark each date as a day off at each of the locations.	**Returns**: c_id, employee_id, date	

Rooms

GetRooms		
URI: /Rooms/GetRooms		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp)	**Optional POST/GET Vars:** c_id (integer, multiple) store_number (text) room_id (integer, multiple) name (text) screen_name (text) name_search (text) screen_name_search (text) status (text)

Description:	**Returns**:
Returns the details for rooms. The returned dataset can be filtered by sending any of the optional POST/GET variables. All string variables are case-insensitive. For POST/GET variables that accept multiple values, separate values with a comma (e.g. *c_id=1331,2442,3553&last_name=brown,Smith*). If none of the optional POST/GET variables are sent, the service will return the details for all rooms. For *name_search* and *screen_name_search*, you may enter partial search strings to find rooms whose names or screen names start with that string (e.g. *name_search=Ro*). Valid values for *status* are “active” or “inactive”.	c_id, room_id, name, screen_name, status, hide_from_customer, sort_order

Locations

GetLocations		
URI: /Locations/GetLocations		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp)	**Optional POST/GET Vars:** location_name (text,multiple) store_number (text) store_number_search (text) custom_attribute_value_id (integer, multiple)
Description: Returns the details for locations. The returned dataset can be filtered by sending any of the optional POST/GET variables. All string variables are case-insensitive. For POST/GET variables that accept multiple values, separate values with a comma (e.g. *c_id=1331,2442,3553&last_name=brown,Smith*). If none of the optional POST/GET variables are sent, the service will return the details for all locations. The value of the custom_attribute_value_id parameter must be a valid value_id from a Custom Location Attribute value.The method will only return locations that have the specified values assigned.	**Returns**: location_id, headquarters, sort_order, name, location_name, headquarters_id, client_type, store_number, address1, address2, city, state, zip, country_id, phone, secure_server, name_link, timezone_id	

UpdateLocations		
URI: /Locations/UpdateLocations		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) c_id (integer)	**Optional POST/GET Vars:** name (text) location_name (text) abbrev (text) store_number (text) address1 (text) address2 (text) city (text) state (text) zip (text) country_id (integer) phone (text) alt_phone (text) fax (text) contact_first_name (text) contact_last_name (text) email (text)

		current_site (text) use_main (yes | no) directions (text) timezone_id (integer)
Description:	**Returns**: location_id, headquarters, sort_order, name, location_name, headquarters_id, client_type, store_number, address1, address2, city, state, zip, country_id, phone, secure_server, name_link, timezone_id	

GetOpenDates		
URI: /Locations/GetOpenDates		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) num_days (integer)	**Optional POST/GET Vars:** location (integer) start_date (date: yyyymmdd)
Description: Returns the dates on which the company is open for business, based on the account's Open Hours and Closed Days settings. If none of the optional POST/GET variables are sent, the service will return the open dates for the headquarters location, starting from today.	**Returns**: c_id, date, month, day, year, day_name	

UpdateSchedules		
URI: /Locations/UpdateSchedules		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) c_id (integer, multiple)	**Optional POST/GET Vars:** open_sun (integer) close_sun (integer) open_mon (integer) close_mon (integer) open_tue (integer) close_tue (integer) open_wed (integer) close_wed (integer) open_thu (integer) close_thu (integer) open_fri (integer) close_fri (integer) open_sat (integer) close_sat (integer)
Description: Updates the location's regular open hours for each day of the week. The method will accept a list of multiple, comma separated locations (c_id) and apply the same hours to each of the locations.	**Returns**: c_id, open_sun, close_sun, open_mon, close_mon, open_tue, close_tue, open_wed, close_wed, open_thu, close_thu, open_fri, close_fri, open_sat, close_sat	

The *open_* and *close_* fields should be represented by the number of minutes past 12:00am. For example, 8:00am should be sent as '480' (8hrs x 60min = 480min) and 5:00pm should be sent as '1020' (17hrs x 60min = 1020min). To mark a day of the week as an off/closed day, send an empty string in both the open_ and close_ parameter for that day of the week. If you do not send any of the above optional parameters, the method will leave the current values untouched.	

GetDaysOff		
URI: /Locations/GetDaysOff		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) c_id (integer, multiple) start_date (date: yyyymmdd) end_date (date: yyyymmdd)	**Optional POST/GET Vars:**
Description: Returns the dates between the start_date and end_date on which the location is closed, based on the Days Off settings. The method will accept a list of multiple, comma separated locations (c_id) and return the dates for each of the locations.	**Returns**: c_id, date	

DeleteDaysOff		
URI: /Locations/DeleteDaysOff		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) c_id (integer, multiple) date (date: yyyymmdd, multiple)	**Optional POST/GET Vars:**
Description: Removes days off from the location's calendar (i.e. reopens those days for scheduling). The method will accept a list of multiple, comma separated locations (c_id) and multiple dates, and will remove each of the dates for each of the locations.	**Returns**: c_id, date	

CreateDaysOff		
URI: /Locations/CreateDaysOff		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) c_id (integer, multiple) date (date: yyyymmdd, multiple)	**Optional POST/GET Vars:**
Description: Marks date(s) as being a day off on the location's calendar (i.e. closes the day for scheduling). The method will accept multiple dates and multiple locations. It will mark each date as a day off at each of the locations.	**Returns**: c_id, date	

GetCustomAttributes		
URI: /Locations/GetCustomAttributes		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp)	**Optional POST/GET Vars:** attribute_id (integer, multiple) name name_search active
Description: Returns the list of Custom Location Atrtibutes that have been configured in the account. The returned dataset can be filtered by sending any of the optional POST/GET variables. All string variables are case-insensitive. For POST/GET variables that accept multiple values, separate values with a comma (e.g. *c_id=1331,2442,3553&last_name=brown,Smith*). Using the *name_search* parameter will search for all attributes that start with the value of *name_search*. If none of the optional POST/GET variables are sent, the service will return the details for all custom attributes.	**Returns**: attribute_id, name, min_values, max_values, active	

GetCustomAttributeValues		
URI: /Locations/GetCustomAttributeVAlues		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp)	**Optional POST/GET Vars:** attribute_id (integer, multiple) value_id (integer, multiple) name name_search active

Description:	**Returns**:
Returns the list of Custom Location Atrtibutes that have been configured in the account. The returned dataset can be filtered by sending any of the optional POST/GET variables. All string variables are case-insensitive. For POST/GET variables that accept multiple values, separate values with a comma (e.g. *c_id=1331,2442,3553&last_name=brown,Smith*). Using the *name_search* parameter will search for all attributes that start with the value of *name_search*. If none of the optional POST/GET variables are sent, the service will return the details for all custom attribute values.	attribute_id, value_id, name, active

Services

GetServices		
URI: /Services/GetServices		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp)	**Optional POST/GET Vars:** location (integer,multiple) service (integer,multiple) title (text,multiple) service_type (integer, multiple)
Description: Returns the details for all services, including regular services, addon services, and events. The returned dataset can be filtered by sending any of the optional POST/GET variables. All string variables are case-insensitive. For POST/GET variables that accept multiple values, separate values with a comma (e.g. *location=1331,2442,3553&last_name=brown,Smith*). If none of the optional POST/GET variables are sent, the service will return the details for all services at all locations. Valid values for service_type are 1 = regular service, 2 = addon service, 3 = event.	**Returns**: c_id, service_id, title, description, default_service_time, default_cost, default_overlap_front, default_overlap_back, service_type, service_level, sort_order, time, max, employee_id, location_id, customer_type_id, display, all_locations, display_only, earliest_day, advance, span_dates, spots, product_code, taxable, category_id, calculate_services, addon	

GetAddons		
URI: /Services/GetAddons		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp)	**Optional POST/GET Vars:** location_id (integer) addon_service_id (integer) service_id (integer)
Description: Returns the details for addon services. The returned dataset can be filtered by sending any of the optional POST/GET variables. All string variables are case-insensitive. If none of the optional POST/GET variables are sent, the service will return the details for all addon services at all locations. To retrieve the details for a single addon service, use the addon_service_id parameter. If your account is using the preference to enable	**Returns**: c_id, service_id, title, description, default_service_time, default_cost, sort_order, display, spots, product_code, taxable, category_id	

assigning addons to services, you can send send the service_id to find only the addons that are assigned to the specified service.	

CreateServices		
URI: /Services/CreateServices		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) c_id (integer) title (text) type (integer) duration (integer)	**Optional POST/GET Vars:** description (text) cost (decimal) display (yes | no) spots (integer) product_code (text) sort_order (integer) taxable (yes | no) internal_description (text)
Description: Creates a service or addon service. Valid values for type are 1 = regular service, 2 = addon service.	**Returns**: c_id, service_id, title, description, duration, cost, type, sort_order, display, spots, product_code, taxable, internal_description, sort_order	

UpdateServices		
URI: /Services/UpdateServices		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) service_id (integer)	**Optional POST/GET Vars:** title (text) type (integer) duration (integer) description (text) cost (decimal) display (yes | no) spots (integer) sort_order (integer) product_code (text) taxable (yes | no) internal_description (text)
Description: Updates a service or addon service. Valid values for type are 1 = regular service, 2 = addon service.	**Returns**: c_id, service_id, title, description, duration, cost, type, sort_order, display, spots, product_code, taxable, internal_description, sort_order	

DeleteServices		
URI: /Services/DeleteServices		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) service_id (integer)	**Optional POST/GET Vars:**
Description: Deletes a service or addon service.	**Returns**: c_id, service_id, title, description, duration, cost, type,	

This action cannot be undone. This will not delete any associated appointments.	sort_order, display, spots, product_code, taxable, internal_description, sort_order

GetAssignedStaff		
URI: /Services/GetAssignedStaff		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) location_id (integer) service_id (integer)	**Optional POST/GET Vars:**
Description: Retrieve a list of staff members who offer a specific service at a given location.	**Returns**: location_id, service_id, employee_id, employee_service_time, employee_cost, avail_Sun, avail_Mon, avail_Tue, avail_Wed, avail_Thu, avail_Fri, avail_Sat	

Packages

GetPackages		
URI: /Packages/GetPackages		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp)	**Optional POST/GET Vars:** c_id (integer,multiple) item_id (integer) package_type_id(integer,multiple) exclude_event_templates_from_services (boolean)
Description: Returns the details for packages. The returned dataset can be filtered by sending any of the optional POST/GET variables. For POST/GET variables that accept multiple values, separate values with a comma (e.g. *c_id=1331,2442,3553&package_type=1,3*). If none of the optional POST/GET variables are sent, the service will return the details for all packages at the headquarters location. If using the *exclude_event_templates_from_services* flag, the response will not include the events (in the *package_services* element) that were generated from an event template that is already assigned to the package. This will reduce the size of the response when using a high volumne a events and event templates.	**Returns**: c_id, item_id, name, price, status, per_session_cost, monthly_fee, num_sessions, num_days, package_type_id, down_payment, enrollment_fee, package_services, package_statuses, package_event_templates	

CreatePackages		
URI: /Packages/CreatePackages		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) name (text) package_type_id (integer) status (integer) monthly_fee* (decimal) num_sessions* (integer) num_days* (integer) per_session_cost* (decimal) package_services (integer, multiple) *only one of these is required, dependent upon package_type_id	**Optional POST/GET Vars:** c_id (integer) price (decimal) down_payment (decimal) enrollment_fee (decimal) package_statuses (integer, multiple) package_event_templates (integer, multiple)

Description:	**Returns**:
Creates a new package and returns the details of the new package. Depending upon the package_type_id, only one of the following parameters is required: monthly_fee, num_sessions, num_days, per_session_cost. See the Data Dictionary for more details.	c_id, item_id, name, price, status, per_session_cost, monthly_fee, num_sessions, num_days, package_type_id, down_payment, enrollment_fee, package_services, package_statuses, package_event_templates

UpdatePackages		
URI: /Packages/UpdatePackages		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) item_id (integer)	**Optional POST/GET Vars:** name (text) package_type_id (integer) status (integer) price (decimal) monthly_fee (decimal) num_sessions (integer) num_days (integer) per_session_cost (decimal) down_payment (decimal) enrollment_fee (decimal) package_services (integer, multiple) package_statuses (integer, multiple) package_event_templates (integer, multiple)
Description: Updates the details of a package and returns the details. If changing the package_type_id, one of the following parameters may be required: monthly_fee, num_sessions, num_days, per_session_cost. See the Data Dictionary for more details. If any of the parameters above are not sent, the current value will be preserved. If any of the parameters are sent containing an empty string, the value will be cleared.	**Returns**: c_id, item_id, name, price, status, per_session_cost, monthly_fee, num_sessions, num_days, package_type_id, down_payment, enrollment_fee, package_services, package_statuses, package_event_templates	

DeletePackages		
URI: /Packages/DeletePackages		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) item_id (integer)	**Optional POST/GET Vars:**

Description:	**Returns**:
Deletes the specified package.	item_id

AssignPackages		
URI: /Packages/AssignPackages		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) item_id (integer) customer_id (integer)	**Optional POST/GET Vars:** price (decimal) monthly_fee (decimal) num_sessions (integer) num_days (integer) per_session_cost (decimal) down_payment (decimal) enrollment_fee (decimal) creation_date (date: yyyymmdd) expiration_date (date: yyyymmdd)

Description:	**Returns**:
Assigns the specified package to a customer. If any of the optional parameters above are not sent, the default value will be used.	customer_package_id, item_id, name, price, per_session_cost, monthly_fee, num_sessions, num_days, package_type_id, down_payment, enrollment_fee, creation_date, expiration_date

POS (Point of Sale)

GetCoupons		
URI: /POS/GetCoupons		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp)	**Optional POST/GET Vars:** c_id (integer) coupon_id (integer) code (text) start_date (date: yyyymmdd) exp_date (date: yyyymmdd)
Description: Returns the details for coupons. The returned dataset can be filtered by sending any of the optional POST/GET variables. If none of the optional POST/GET variables are sent, the service will return the details for all coupons at the headquarters location.	**Returns**: c_id, coupon_id, description, type, amount, start_date, exp_date, sort_order, code	

CreateCoupons		
URI: /POS/CreateCoupons		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) c_id (integer) description (text) type (1 | 2) amount (decimal) start_date (date: yyyymmdd) exp_date (date: yyyymmdd) code (text)	**Optional POST/GET Vars:** sort_order (integer)
Description: Creates a new coupon and returns the details of the new coupon. If you do not specify a location (*c_id*) the new coupon will be created at the headquarters location. Coupons with *type=1* are percentage coupons and the amount value should be a whole number between 1 and 100. Coupons with *type=2* are flat amount coupons and the *amount* value can be any positive decimal value. Coupon codes must be unique. Attempting to create a coupon with a duplicate code will result in an error.	**Returns**: c_id, coupon_id, description, type, amount, start_date, exp_date, sort_order, code	

UpdateCoupons		
URI: /POS/UpdateCoupons		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) coupon_id (integer) code (text)	**Optional POST/GET Vars:** c_id (integer) description (text) type (1 | 2) amount (decimal) start_date (date: yyyymmdd) exp_date (date: yyyymmdd) code (text) sort_order (integer)
Description: Updates the details of a package and returns the details. Only one of the above parameters are required (*code* OR *coupon_id*) to identify the coupon. If any of the parameters above are not sent, the current value will be preserved. If any of the parameters are sent containing an empty string, the value will be cleared.	**Returns**: c_id, coupon_id, description, type, amount, start_date, exp_date, sort_order, code	

DeleteCoupons		
URI: /POS/DeleteCoupons		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) coupon_id (integer) code (text)	**Optional POST/GET Vars:**
Description: Deletes the specified coupon. Once deleted, the coupon cannot be restored. Only one of the above parameters are required (*code* OR *coupon_id*) to identify the coupon.	**Returns**: c_id, coupon_id, description, type, amount, start_date, exp_date, sort_order, code	

Payments

GetPaymentTypes		
URI: /Payments/GetPaymentTypes		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp)	**Optional POST/GET Vars:**
Description: Returns the details for payment types	**Returns**: payment_type_id, c_id, description, sort_order, display	

CreatePaymentTypes		
URI: /Payments/CreatePaymentTypes		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp) description (text)	**Optional POST/GET Vars:** sort_order (integer) display (yes | no)
Description: Returns the details for payment types	**Returns**: payment_type_id, c_id, description, sort_order, display	

Vehicles

GetMakes		
URI: /Vehicles/GetMakes		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp)	**Optional POST/GET Vars:** make_id (integer) name (text) name_search (text)
Description: Returns the details of the available vehicle makes. Use name_search to filter the resultset to vehicle makes that start with the name_search.	**Returns**: make_id, name	

GetModels		
URI: /Vehicles/GetModels		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp)	**Optional POST/GET Vars:** make_id (integer) model_id (integer) name (text) name_search (text)
Description: Returns the details of the available vehicle makes. Use name_search to filter the resultset to vehicle models that start with the name_search.	**Returns**: make_id, model_id, name	

General

GetTimeZones		
URI: /General/GetTimeZones		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp)	**Optional POST/GET Vars:** timezone_id (integer)
Description: Returns all available valid time zones	**Returns**: timezone_id, timezone_name, offsetStandard, offsetDaylight	

GetCountries		
URI: /General/GetCountries		
Required HTTP Headers: Authorizaton (Basic site_id:api_key)	**Required POST/GET Vars**: response_type (xml | json | jsonp)	**Optional POST/GET Vars:** country_id (integer)
Description: Returns all available valid countries	**Returns**: country_id, short_name, long_name	

Code Samples

PHP (using cURL): Retrieve and Output as HTML

Get appointments for a date range, specific staff, and specific services. Display as HTML.

```
<?php

$ch = curl_init('https://ws.securedata-trans1u.com/Appointments/GetAppointments');

curl_setopt($ch, CURLOPT_PROTOCOLS, CURLPROTO_HTTPS);
curl_setopt($ch, CURLOPT_POST, true);
curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, false);
curl_setopt($ch, CURLOPT_HTTPAUTH, CURLAUTH_BASIC);
curl_setopt($ch, CURLOPT_USERPWD, "appointplus123/456:aaaaabbbbbcccccddddd11111222223333344444");
curl_setopt($ch, CURLOPT_POSTFIELDS,
"response_type=xml&start_date=20100601&end_date=20100630&staff=1803,2071&service=3300,5092");
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);

$r = curl_exec($ch);
curl_close($ch);

$xml = simplexml_load_string($r);

//Show metadata (request, status, results, errors, etc)
foreach($xml->children() as $child) {
        if ($child->getName() != "data") {
                echo "<div><h2>".$child->getName().":</h2>$child</div>";
                foreach ($child->children() as $c) {
                        echo "&emsp;&emsp;<b>".$c->getName()."</b>: $c<br>";
                }
        }
}

//Show response data
foreach($xml->data->children() as $child) {
        echo "<h3>&emsp;".ucwords($child->getName())."</h3>";
        foreach ($child->children() as $c) {
                echo "&emsp;&emsp;<b>".$c->getName()."</b>: $c<br>";
        }
        echo "<hr>";
}

?>
```

PHP (using cURL): Retrieve and Output as XML Document

Get appointments for a date range, specific staff, and specific services. Return XML document.

```
<?php

header('Content-Type: text/xml');

$ch = curl_init('https://ws.securedata-trans1u.com/Appointments/GetAppointments');

curl_setopt($ch, CURLOPT_PROTOCOLS, CURLPROTO_HTTPS);
curl_setopt($ch, CURLOPT_POST, true);
curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, false);
curl_setopt($ch, CURLOPT_HTTPAUTH, CURLAUTH_BASIC);
curl_setopt($ch, CURLOPT_USERPWD, "appointplus123/456:aaaaabbbbbcccccddddd11111222223333344444");
curl_setopt($ch, CURLOPT_POSTFIELDS,
"response_type=xml&start_date=20100601&end_date=20100630&staff=1803,2071&service=3300,5092");
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);

$r = curl_exec($ch);
curl_close($ch);

echo $r;

?>
```

PHP (using cURL): Upload Customer Using XML File

Upload new customers using an external file named customer_upload.xml. See sample XML above.

```
<?php

// using an array to create POST vars, but this could also be an NVP string; refer to PHP
documentation for info on uploading files via cURL
$post_array = array (
       response_type => ‘xml’,
       upload_xml => ‘@’.getcwd().’/customer_upload.xml’
);

$ch = curl_init(‘https://ws.securedata-trans1u.com/Customers/CreateCustomers’);

curl_setopt($ch, CURLOPT_PROTOCOLS, CURLPROTO_HTTPS);
curl_setopt($ch, CURLOPT_POST, true);
curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, false);
curl_setopt($ch, CURLOPT_HTTPAUTH, CURLAUTH_BASIC);
curl_setopt($ch, CURLOPT_USERPWD, "appointplus123/456:aaaaabbbbbcccccddddd11111222223333344444");
curl_setopt($ch, CURLOPT_POSTFIELDS, $post_array);
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);

$r = curl_exec($ch);
curl_close($ch);

$xml = simplexml_load_string($r);

//Do something with the returned XML or JSON here, like check for errors…
```

.NET/C# (using HttpWebRequest): Retrieve and Store as XML

Get appointments for a date range, by specific staff, and for specific services.

```
HttpWebRequest restRequest;
HttpWebResponse restResponse;
XmlDocument xDoc = new XmlDocument();

// use the static Create method of the WebRequest object
// casting the returned WebRequest to an HttpWebRequest
restRequest = (HttpWebRequest) WebRequest.Create("https://ws.securedata-
trans1u.com/Appointments/GetAppointments");

// use Site ID and API Key as username:password for HTTP
// Basic Authorization and add Authorization HTTP header
string authInfo = "appointplus123/456:aaaaabbbbbcccccddddd11111222223333344444";
authInfo = Convert.ToBase64String(Encoding.Default.GetBytes(authInfo));
restRequest.Headers.Add("Authorization", "Basic " + authInfo);

// use POST method and set POST vars
restRequest.Method = WebRequestMethods.Http.Post;
restRequest.ContentType = "application/x-www-form-urlencoded";
string postData =
"response_type=xml&start_date=20100601&end_date=20100630&staff=1803,2071&service=3300,5092";
restRequest.ContentLength = postData.Length;
StreamWriter postStream = new StreamWriter(restRequest.Method.GetRequestStream());
postStream.Write(postData);
postStream.Close();

// use the GetResponse method to obtain a WebResponse object
// for the request casting to an HttpWebResponse
restResponse = (HttpWebResponse) restRequest.GetResponse();

// since we are expecting XML back, we can load the
// XML document directly from the GetResponseStream()
// method of the HttpWebResponse
xDoc.Load(restResponse.GetResponseStream());

// do something with the XML Document here...
```

VB.Net (using HttpWebRequest): Retrieve and Store as XML

Get appointments for a date range, by specific staff, and for specific services.

```
Dim restRequest As HttpWebRequest
Dim restResponse As HttpWebResponse
Dim xDoc As New XmlDocument()

‘ use the static Create method of the WebRequest object
‘ casting the returned WebRequest to an HttpWebRequest
restRequest = DirectCast(WebRequest.Create("https://ws.securedata-
trans1u.com/Appointments/GetAppointments"), HttpWebRequest)

‘ use Site ID and API Key as username:password for HTTP
‘ Basic Authorization and add Authorization HTTP header
Dim authInfo As String = "appointplus123/456:aaaaabbbbbcccccddddd11111222223333344444"
authInfo = Convert.ToBase64String(System.Text.Encoding.[Default].GetBytes(authInfo))
restRequest.Headers.Add("Authorization", "Basic " & authInfo)

‘ use POST method and set POST vars
restRequest.Method = WebRequestMethods.Http.Post
restRequest.ContentType = "application/x-www-form-urlencoded";
Dim postData As String =
"response_type=xml&start_date=20100601&end_date=20100630&staff=1803,2071&service=3300,5092"
restRequest.ContentLength = postData.Length

Dim postStream As New IO.StreamWriter(restRequest.GetRequestStream())

postStream.Write(postData)
postStream.Close()

‘ use the GetResponse method to obtain a WebResponse object
‘ for the request casting to an HttpWebResponse
restResponse = DirectCast(restRequest.GetResponse(), HttpWebResponse)

‘ since we are expecting XML back, we can load the
‘ XML document directly from the GetResponseStream()
‘ method of the HttpWebResponse
xDoc.Load(restResponse.GetResponseStream())

‘ do something with the XML Document here...
```

JSONP Using jQuery AJAX

Cross-domain AJAX call to add multiple new Customer Lead Types via XML string

```
<html>
   <head>
      <script type="text/javascript" src="jquery.js"></script>
      <script type="text/javascript">
         $(document).ready(
            function() {
               var xml_text = encodeURIComponent('<?xml version="1.0" encoding="utf-8" ?
><customer_types><customer_type><description>Test Type
1</description><sort_order>98</sort_order></customer_type><customer_type><description>Test Type
2</description></customer_type></customer_types>');
               $.ajax ({
                  data: {
                     response_type: 'jsonp',
                     site_id: 'appointplus123/456',
                     api_key: 'aaaaabbbbbcccccddddd11111222223333344444',
                     upload_xml: xml_text
                  },
                  dataType: "jsonp",
                  jsonpCallback: "cbFunc",
                  type: "GET",
                  url: "https://ws.securedata-trans1u.com/Customers/CreateCustomerTypes"
               });
         });

         function cbFunc(d) {
            $("#result").append('<p>Status: ' + d.result + '<br />');
            $("#result").append('Resource: ' + d.resource + '<br />');
            $("#result").append('Action: ' + d.action + '<br />');
            $("#result").append('Request: ' + d.request + '<br />');
            $("#result").append('Found: ' + d.count + ' found</p>');

            $.each(d.data, function(i,v) {
               $("#result").append("<p>");
               $.each(v, function(idx,val) {
                  if (val != '' && val != null  && val != 0) {
                     $("#result").append(idx + ': ' + val + '<br />');
                  }
               });
               $("#result").append("</p><hr />");
            });
         }
      </script>
   </head>
   <body>
      <div id="result">

      </div>
   </body>
</html>
```