GenericIO SOAP Web Service

Provides simple input/output access to collections of data entities within TeamWork.

The GenericIO web services provides access to query, and in some cases, update data stored within TeamWork. The various data entities in TeamWork are converted to a generic xml format that is consistent across all data entities.

Overview

The URL's below depend on where your Instance is installed. For example:

> Enterprise Instance base URL: https://www.schedulesource.net/Enterprise/

Please adjust the URL's below by replacing <Instance> with your instance's base URL.

Service URL:

<Instance>/TeamWork/Services/GenericIO.asmx

Service Description

<Instance>/TeamWork/Services/GenericIO.asmx?WSDL

Security

The GenericIO service requires valid credentials for the Enterprise Portal. These are passed as an entCredential object in the body of the request.

Methods

- **GetCollection** – Gets a collection of data based on the [entitytype] value provided.
- **GetCollectionXml** – Same as GetCollection but returns named Xml tags for entities, with fields as attributes.
- **GetCollectionXmlTags –** Same as GetCollection but returns named Xml tags for entities and fields (no attributes).
- **UpdateCollection** – Updates a collection of data based on the [entitytype] and [actions] parameters.
- **UpdateCollectionXml** – Same as UpdateCollection but expects an XmlElement in tag + attribute format.
- **UpdateCollectionXmlTags** - Same as UpdateCollection but expects an XmlElement in tags-only format.

Entity Types

The xml format used in this API is the same for all entities. However, the entitytype is specified as an attribute on the collection. Not all entities are updateable, and each entity has a list of fields with varying attributes (data-type, read-only, deprecated, etc.). These entities are described in more detail later in the document.

- User
- Business
- Station
- Employee
- Skill
- LocalUser
- LocalEmployee
- LocalSkill
- Schedule
- ScheduleShift
- Template
- TemplateShift
- PayPeriod
- PaySheet
- TimeProject
- TimeTask
- TimeActivity
- TimeEntry
- TimeAccrualBalance
- LeaveType
- LeaveEntry
- Credential
- EmployeeCredential
- StationCredential

Method: GetCollection

The GetCollection method will fetch a collection of entities. At its simplest, the request only needs to specify enterprise credentials and an EntityType. Here's an example request (to fetch a list of User's):

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ser="http://ws.schedulesource.com/teamwork/services/">
   <soapenv:Header/>
   <soapenv:Body>
      <ser:GetCollection>
         <ser:entCredential>
            <ser:Code>EnterpriseCode</ser:Code>
            <ser:User>EnterpriseUsername</ser:User>
            <ser:Password>EnterpriseUserPassword</ser:Password>
         </ser:entCredential>
         <ser:filter>
            <ser:EntityType>User</ser:EntityType>
         </ser:filter>
      </ser:GetCollection>
   </soapenv:Body>
</soapenv:Envelope>
```

For some collections, it is desirable, and sometimes required, to filter the request by dates and/or Id's. To add filtering criteria, use FilterItem's within a FilterValues collection. Here's an example request to fetch all scheduled shifts, within a single schedule on a particular date (note – for shifts, MinDate and MaxDate are required):

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ser="http://ws.schedulesource.com/teamwork/services/">
   <soapenv:Header/>
   <soapenv:Body>
      <ser:GetCollection>
         <ser:entCredential>
            <ser:Code>EnterpriseCode</ser:Code>
            <ser:User>EnterpriseUsername</ser:User>
            <ser:Password>EnterpriseUserPassword</ser:Password>
         </ser:entCredential>
         <ser:filter>
            <ser:EntityType>ScheduleShift</ser:EntityType>
            <ser:FilterValues>
               <ser:FilterItem>
             <ser:Key>MinDate</ser:Key>
             <ser:Value>3/11/2012</ser:Value><ser:Operator>=</ser:Operator>
               </ser:FilterItem>
                <ser:FilterItem>
                  <ser:Key>MaxDate</ser:Key>
                  <ser:Value>3/11/2012</ser:Value><ser:Operator>=</ser:Operator>
               </ser:FilterItem>
             <ser:FilterItem>
                  <ser:Key>ScheduleId</ser:Key>
                  <ser:Value>124739</ser:Value><ser:Operator>=</ser:Operator>
               </ser:FilterItem>
            </ser:FilterValues>
         </ser:filter>
      </ser:GetCollection>
   </soapenv:Body>
</soapenv:Envelope>
```

Response Format

All responses share the same xml format. Here's an example response, with some fields and entities removed for brevity:

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
   <soap:Body>
      <GetCollectionResponse xmlns="http://ws.schedulesource.com/teamwork/services/">
         <GetCollectionResult entitytype="ScheduleShift">
             <Entity>
               <Field n="Id" v="4267976"/>
               <Field n="ScheduleId" v="124739"/>
               <Field n="BusinessId" v="2"/>
               <Field n="DayId" v="1"/>
               <Field n="ClientId"/>
               <Field n="Date" v="3/11/2012 12:00:00 AM"/>
               <Field n="ShiftStart" v="1/1/1900 8:00:00 AM"/>
               <Field n="ShiftEnd" v="1/1/1900 12:00:00 PM"/>
               <Field n="BreakStart" v="1/1/1900 9:00:00 AM"/>
               <Field n="BreakEnd" v="1/1/1900 9:15:00 AM"/>
```

```
            <Field n="Hours" v="3.75"/>
          </Entity>
          <Entity>
            <Field n="Id" v="4267977"/>
            <Field n="ScheduleId" v="124739"/>
            <Field n="BusinessId" v="2"/>
            <Field n="DayId" v="1"/>
            <Field n="ClientId" v="0"/>
            <Field n="Date" v="3/11/2012 12:00:00 AM"/>
            <Field n="ShiftStart" v="1/1/1900 8:00:00 AM"/>
            <Field n="ShiftEnd" v="1/1/1900 12:00:00 PM"/>
            <Field n="BreakStart" v="1/1/1900 9:00:00 AM"/>
            <Field n="BreakEnd" v="1/1/1900 9:15:00 AM"/>
             <Field n="Hours" v="3.75"/>
          </Entity>
        </GetCollectionResult>
      </GetCollectionResponse>
   </soap:Body>
</soap:Envelope>
```

In the response above, each “Entity” is of type “ScheduleShift”. The fields contain two attributes each: “n” for name and “v” for value. In the appendices, we provide some XSLT transformations to convert this xml format to other useful formats.

The filter values that are available and/or required are listed with the Entity descriptions later in this document.

Filtering Fields

Using FilterItem's within the FilterValues tag, you can specify limitations on the data to be returned. Any of the fields in the EntityCollection can be used as a filter. In addition, two built-in filter Fields exist: **MinDate** and **MaxDate**. Certain entity types require these dates be passed in the filter or they simply default to the current date.

Filtering Operations

The operations available for filtering generally correspond to those available within a SQL environment. The following operations are currently supported:

- **=** (equal)
- **<** (less than)
- **>** (greater than)
- **<=** (less than or equal)
- **>=** (greater than or equal)
- **<>** (not equal)
- **NULL** (has no value. Note: will not match zero-length strings "")
- **NOT NULL** (has a value)
- **IN** (matches one of the values in comma-separated list)
- **LIKE** (matches pattern with wildcards)
    - **%** - Any string of zero or more characters.
    - **_** - Any single character.
    - **[]** - Any single character within the specified range ([a-f]) or set ([abcdef]).
    - **[^]** - Any single character not within the specified range ([^a-f]) or set ([^abcdef]).

Selecting Fields

A special built-in filter value named "Fields" exists to specify the fields you want returned. The value should be a comma-separated list of the fields to return (case-sensitive).

Method: UpdateCollection

The UpdateCollection method accepts a collection of entities and performs updates as specified in the request. Two key fields specify what types of updates are desired and on what entities. These are:

- entitytype
- actions

The entitytype specifies what data is in the collection. The actions attribute is a comma-separated list of desired updates. An action must be supported by the entitytype for it to be executed (see entity descriptions for supported actions). If not specified, the actions value defaults to "Add, Update", which will attempt to add or update records in the database based on the data provided.

Updates are only performed on items that can be found in the database and so key identification fields must be provided in the data for updates to occur. When adding data, certain key fields will be required (these are listed with entity descriptions).

Each entity in the collection passed to the UpdateCollection method can have different fields that may be updated. Unless loading data, these records can be sparse and only contain the fields that have changed (along with the required fields).

Here's an example of updating an Employee's LastName:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ser="http://ws.schedulesource.com/teamwork/services/">
   <soapenv:Header/>
   <soapenv:Body>
      <ser:UpdateCollection>
         <ser:entCredential>
            <ser:Code>EnterpriseCode</ser:Code>
            <ser:User>EnterpriseUsername</ser:User>
            <ser:Password>EnterpriseUrerPassword</ser:Password>
         </ser:entCredential>
         <ser:collection entitytype="Employee" actions="Update">
            <ser:Entity>
               <ser:Field n="LastName" v="Walker"/>
               <ser:Field n="ExternalId" v="44555"/>
             </ser:Entity>
         </ser:collection>
      </ser:UpdateCollection>
   </soapenv:Body>
</soapenv:Envelope>
```

In this case, the only required field is "ExternalId", which is a unique field from an external source (not TeamWork, but from one of your systems).

The response data will provide a summary of the actions taken. In addition, if any data is invalid, the data is returned in the response. Below are examples of a successful and unsuccessful update request.

SUCCESSFUL RESPONSE

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
   <soap:Body>
      <UpdateCollectionResponse
xmlns="http://ws.schedulesource.com/teamwork/services/">
         <UpdateCollectionResult>
            <Note k="Added" v="0"/>
            <Note k="Updated" v="1"/>
            <Note k="Found" v="1"/>
            <Note k="Skipped" v="0"/>
            <Note k="Invalid" v="0"/>
         </UpdateCollectionResult>
      </UpdateCollectionResponse>
   </soap:Body>
</soap:Envelope>
```

A collection of Note's are returned, with "k" = key and "v" = value. These provide a summary of the update. In the response above we see that one record was found (due to correct Id's) and one record was updated. If an entity is "updated", but no data has changed (e.g. change LastName from "Smith" to "Smith"), the update will be tallied as "Skipped".

UNSUCCESSFUL RESPONSE

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
   <soap:Body>
      <UpdateCollectionResponse
xmlns="http://ws.schedulesource.com/teamwork/services/">
         <UpdateCollectionResult>
            <EntityCollection entitytype="Employee" classification="Invalid">
               <Entity>
                  <Field n="LastName" v="Walker B"/>
                  <Field n="BadId" v="44555"/>
                  <Note k="Error" v="Missing ExternalId"/>
               </Entity>
            </EntityCollection>
            <Note k="Added" v="0"/>
            <Note k="Updated" v="0"/>
            <Note k="Found" v="0"/>
            <Note k="Skipped" v="0"/>
            <Note k="Invalid" v="1"/>
         </UpdateCollectionResult>
      </UpdateCollectionResponse>
   </soap:Body>
</soap:Envelope>
```

Here you'll notice that we have a collection of "Invalid" entities. These are the ones that failed. In addition, with each failed entity is a Note with "k" (key) = "Error" and an error description. Finally the summary Note with "k" = "Invalid" has a value of "1".

The inclusion of invalid data and error messages in the response will allow you to quickly troubleshoot UpdateCollection errors. Any integration service that relies on this service should log all responses locally for some period of time.

ENTITY DESCRIPTIONS

The generic nature of the xml format used in this service allows us to add entities and fields in the future without breaking major functionality. Some fields, however, that are included in the service right now could be removed at some point (these are listed as deprecated).

FIELDS

Fields are the named values associated with an entity. Typically fields are inherent to the entity, but in some cases they exist on other entities and are provided for convenience. An example is BusinessName on a ScheduleShift. The BusinessName is stored on the Business entity but is provided in the GetCollection response nonetheless.

Each field has a data type which specifies what kind of value the field holds. Data types include: DateTime, Integer, String, Real, and Boolean.

Some fields are read-only. These fields can be fetched by GetCollection but are not updatable by UpdateCollection. One field, "Password", cannot be fetched but is updatable when the action "Password" is included in the UpdateCollection request.

Finally, some fields may be available in this service but are currently unused in the application.

EXTERNAL ID'S

Some entities require "ExternalId's" for adding or updating. These externalId's must be generated by the customer and exist on records within TeamWork for the updates to work. This allows customers to manage records in their systems and link them to records in TeamWork without having to import Id's from TeamWork into their systems.

Currently, the ExternalId's only exist on some top-level items, like "Employee", "Business", and "User" (actually for "User", it's Username). We carry them down to entities like "Skills" and "LocalSkills". Some entities lacking dedicated "ExternalId's" include Schedule and ScheduleShift. When integrating schedule data, the customer will have to rely on Id's generated in TeamWork to successfully isolate and update specific records.

ACTIONS

The following actions are supported by some of the entities. Support for actions will increase as new releases are developed. You can include multiple actions in the Actions attribute by separating the keywords by commas (e.g. Actions="Update,Password").

Add – Add a new record (entity) to the system.
Update – Updates a record's field values.
Password – Updates the password of a User or Employee.
Deploy – Deploys a User or Employee to a Business (Location).
Delete – Deletes the entity from the system (very limited support).
Inactivate – Sets an employee (or user) as inactive. Requires field Termdate.
Assign – Assigns (or clears) an employee to a shift.

Business

Also known as a “Location” in TeamWork. The business organizes all scheduling and time tracking activities. Currently Business entities are READ-ONLY. In addition, the Business entity stores a lot of settings that might be deprecated by moving them to a more generic and flexible settings entity.

FIELD	DATA TYPE	DESCRIPTION
BusinessId	Integer	Unique Id generated by TeamWork.
Name		
Code		
BusinessExternalId	String	An Id that corresponds to an external system.
City		
State		
Country		
Address1		
Address2		
Zip		
Phone		
Fax		
Enabled		
MaxEmp		
MinHours		
MaxHours		
MaxHoursStrict		
EmpConsecutiveDays		
MinDowntime		
MaxUsers		
Guid		
AllowSwapping		
TimeZoneOffset		
TimeZoneUsDay		
EmpFulLView		
Randomize		
RandomShifts		
…		

(NOT COMPLETE)

User

Users have management-level access to the system. A user might have access to only the Enterprise portal, only Location Portal, or both. LocalUser entities must exist for a User to access a Location. The User entity however, stores the Name, Email, Username, etc. for Users and LocalUsers.

FIELD	DATA TYPE	DESCRIPTION
UserId	Integer	Internal Id generated by TeamWork.
Username	String	Unique Id for sign-in. Also used as "ExternalId" for this entity.
FirstName	String	
LastName	String	
Email	String	The user's email address.
Admin	Boolean	A flag that turns Enterprise Portal access on or off.
Enabled	Boolean	A flag that enables or disables the User. (Will disable notifications for example.)
Phone	String	The user's phone number.
Udf1	String	User-defined field.
Udf2	String	User-defined field.
Udf3	String	User-defined field.
Udf4	String	User-defined field.
Udf5	String	User-defined field.
Udf6	String	User-defined field.
IsReadOnly	Boolean	A flag to specify user has read-only rights to the system. Now controlled by Role Security.
FolderId	Integer	Internal Id for User's top-level folder access.
Password	String	Not visible but can be updated by including action "Password".

Filters

Currently no filter values are supported. The GetCollection method will return a list of all Users.

Updates

The UpdateCollection method supports the following actions with the given conditions:

ACTION	Required Identifiers	Required Fields
Add	**Username**	FirstName, LastName
Update	**Username**	
Password	**Username**	Password

Employee

FIELD	DATA TYPE	DESCRIPTION
EmployeeId	Integer	Internal Id generated by TeamWork.
LastName	String	
FirstName	String	
NickName	String	
PhoneNo	String	
PhoneNo2	String	
PhoneNo3	String	
Fax	String	
Address	String	
Adddress2	String	
City	String	
State	String	
Postalcode	String	
Country	String	
EmployeeNum	String	
Email	String	
MinHours		
MaxHours		
MaxShiftsPerDay		
Created	DateTime	
MaxDays		
Notes		
ExternalId	String	
Hiredate	DateTime	
Termdate	DateTime	
MaxHousrPerDay		
Birthdate	DateTime	
Udf1	String	
Udf2	String	
Udf3	String	
Udf4	String	
Udf5	String	
Udf6	String	
AllowSwap	Boolean	
Rank		
CardId		
DefaultBid		
TimeZoneOffset		
TimeZoneUsday		
IVRPinCode		
DefaultTimeCode		

BioID		
DefaultTimeActivity		
MaxConsecDays		
WeeklyOT		
DailyOT		
Guid	String	

Filters

Currently no filter values are supported. The GetCollection method will return a list of all Employees.

Updates

The UpdateCollection method supports the following actions with the given conditions:

ACTION	Required Identifiers	Required Fields
Add	**ExternalId**	FirstName, LastName
Update	**ExternalId**	
Password	**ExternalId**	Password

Station

Stations represent work positions for shifts. Schedules use LocalStation records for tracking shifts. A LocalStation is a Station that has been deployed to a Business (Location). The Station entity is the master record for stations with attributes shared across the enterprise.

FIELD	DATA TYPE	DESCRIPTION
StationId	Integer	Internal Id generated by TeamWork.
Name	String	The name of the station or position.
Durations	String	(Deprecated)
Notes	String	
Udf1	String	
Udf2	String	
Udf3	String	
Udf4	String	
Udf5	String	
Udf6	String	
PayRate		
AllowSwap	Boolean	
ExternalId	String	
Alias	String	
Color	String	

Filters

Currently no filter values are supported. The GetCollection method will return a list of all Stations.

Updates

The UpdateCollection method supports the following actions with the given conditions:

ACTION	Required Identifiers	Required Fields
Add	**ExternalId**	Name
Update	**ExternalId**	

Skill

A master record (enterprise-level) that links an Employee to a Station. This link defines the master list of allowed stations for an employee. Within a location (Business) an employee might have some sub-set of this master list. The LocalSkill is the localized list for a location.

FIELD	DATA TYPE	DESCRIPTION
Level	Integer (0-9)	The numeric skill level, or priority of the employee working the station.
EmployeeExternalId	String	The unique employee identifier from external system.
StationExternalId	String	The unique station identifier from external system.

Filters

Currently no filter values are supported. The GetCollection method will return a list of all Skills.

Updates

The UpdateCollection method does not support updates to Skill entities.

LocalUser

LocalUser entities must exist for a User to access a Location. Almost all values for a LocalUser is inherited from the User entity.

FIELD	DATA TYPE	DESCRIPTION
UserName	String	Unique Id for sign-in. Also used as "ExternalId" for this entity.
FirstName	String	Read-Only
LastName	String	Read-Only
Disabled	Boolean	A flag to turn off or on sign-in access to the location.
BusinessExternalId	String	The unique location identifier from external system.

Filters

The GetCollection for LocalUser supports a filter on **BusinessId**.

Updates

The UpdateCollection method supports the following actions with the given conditions:

ACTION	Required Identifiers	Required Fields
Update	**UserName, BusinessExternalId**	Disabled (the only updatable field)
Deploy	**UserName, BusinessExternalId**	

LocalStation

FIELD	DATA TYPE	DESCRIPTION
Name	String	
ExternalId	String	The unique station identifier from external system.
BusinessExternalId	String	The unique location identifier from external system.

Filters

The GetCollection for LocalStation supports a filter on **BusinessId**.

Updates

The UpdateCollection method does not support updates to LocalStation entities.

LocalSkill

FIELD	DATA TYPE	DESCRIPTION
Level	Integer (0-9)	The numeric skill level, or priority of the employee working the station.
EmployeeExternalId	String	The unique employee identifier from external system.
StationExternalId	String	The unique station identifier from external system.
BusinessExternalId	String	The unique location identifier from external system.

Filters

The GetCollection for LocalStation supports a filter on **BusinessId**.

Updates

The UpdateCollection method does not support updates to LocalSkill entities.

LocalEmployee

LocalEmployee entities must exist for an Employee to access, and be scheduled within, a Location. Almost all values for a LocalEmployee is inherited from the Employee entity.

FIELD	DATA TYPE	DESCRIPTION
EmployeeNum	String	A unique id that is used for sign-in (equivalent to Username).
FirstName	String	Read-Only
LastName	String	Read-Only
HireDate	DateTime	The “start” or “effective” date for the employee at the location.
TermDate	DateTime	The “end” or “termination” date for the employee at the location.
ExternalId	String	The unique employee identifier from external system.
BusinessExternalId	String	The unique location identifier from external system.

Filters

The GetCollection for LocalEmployee supports a filter on **BusinessId**.

Updates

The UpdateCollection method supports the following actions with the given conditions:

ACTION	Required Identifiers	Required Fields
Update	**ExternalId, BusinessExternalId (or BusinessId)**	
Deploy	**ExternalId, BusinessExternalId (or BusinessId)**	

Schedule

A master record (location-level) for a collection of shifts. The schedule defines a date range and holds attributes such as “Publish” and “Firm”.

FIELD	DATA TYPE	DESCRIPTION
ScheduleId	Integer	Internal Id generated by TeamWork.
BusinessId	Integer	Internal Id generated by TeamWork.
Name	String	
DateStart	DateTime	
DateEnd	DateTime	
Created	DateTime	
Modified	DateTime	
Publish	Integer	
Firm	Integer	
Archived	Integer	
IsCompliant	Integer	
EnforceCompliance	Integer	
AllowSwap	Integer	Flag to turn swapping on or off for the shifts within the schedule.
IsEnterpriseDeployed	Boolean	Flag to indicate if Schedule was created by deploying enterprise shifts.
EnterpriseScheduleId	Integer	Internal Id generated by TeamWork. Links the Schedule to an EnterpriseSchedule that was the basis for its creation.
BudgetSeriesId	Integer	Internal Id generated by TeamWork. Links a Schedule to a budget data series for analysis.
BusinessExternalId	String	The unique location identifier from external system.

Filters

Required Filter Values: **MinDate, MaxDate**
Optional Filter Values: BusinessId, ScheduleId, ScheduleIdList (a comma separated list of ScheduleId’s)

Updates

The UpdateCollection method supports the following actions with the given conditions:

ACTION	Required Identifiers	Required Fields
Add	**BusinessExternalId (or BusinessId)**	DateStart, DateEnd, Name
Update	**ScheduleId, BusinessExternalId (or BusinessId)**	

ScheduleShift

A work shift, for an Employee at a Station within a Location.

FIELD	DATA TYPE	DESCRIPTION
Id	Integer	Internal Id generated by TeamWork.
ScheduleId	Integer	Internal Id generated by TeamWork. Links a shift to a Schedule.
BusinessId	Integer	Internal Id generated by TeamWork. Links a shift to a location.
DayId	Integer	Represents the day of the week. 1 = Sunday, 2 = Monday, etc.
ClientId	Integer	Internal Id generated by TeamWork. Links a shift to a Client.
Date	DateTime	
ShiftStart	DateTime	Start time of shift – currently the date portion is ignored.
ShiftEnd	DateTime	End time of shift – currently the date portion is ignored.
BreakStart	DateTime	
BreakEnd	DateTime	
Hours		Total hours of shift. Shift Times span minus Break Times span.
EmpRate		An estimate of the employee's pay rate at time of shift.
EstCost		An estimate of the cost of the shift in wages.
IsSwapping		Flag to specify if shift is posted on swap board.
SwapAvailDate		Timestamp of the swap being posted on swap board.
SwappingToId		When swap approvals are enabled, this holds the id of the employee who claimed the shift but is awaiting approval.
ShiftGroup		
Note		
Created	DateTime	
Updated	DateTime	
UpdateUserType		
UpdateUserId		
UpdateAction		
IsEmployeeApproved		
SwapReason		
BidBoardId		
EnterpriseNote		
UpdateReason		
InactiveCode		
ScheduleTourId		
EnterpriseShiftId		
LastName		
FirstName		
EmployeeExternalId		
StationName		
StationExternalId		
BusinessName		
BusinessExternalId		

Filters

Required Filter Values: **MinDate, MaxDate**
Optional Filter Values: BusinessId, ScheduleId, ScheduleIdList (a comma separated list of ScheduleId's)

Updates

The UpdateCollection method supports the following actions with the given conditions:

ACTION	Required Identifiers	Required Fields
Add	**ScheduleId,** **BusinessExternalId (or BusinessId),** **StationExternalId (or StationId)**	DateStart, DateEnd, Name
Update	**Id**	

TimeEntry

A time clock, time card, or admin entry that track time & hours for payroll.

FIELD	DATA TYPE	DESCRIPTION
Id	Integer	Internal Id generated by TeamWork.
LocalEmployeeId	Integer	Internal Id generated by TeamWork. Is the id of the employee at the location.
BusinessId	Integer	Internal Id generated by TeamWork. Id of a location.
ScheduleShiftId	Integer	Internal Id of a shift, linked to Time Entry.
LocalStationId	Integer	Internal Id of a station at a location, linked to Time Entry.
LeaveEntryId	Integer	Internal Id of a leave entry, linked to Time Entry.
EntryType	String	The time of time entry: clock, card or admin.
ClockOn	DateTime	The UTC time of clocking on.
ClockOff	DateTime	The UTC time of clocking off.
ClockHours	Double	The calculated hours based on ClockOn and ClockOff.
FinalOn	DateTime	The UTC start time of entry, after editing and rounding.
FinalOff	DateTime	The UTC end time of entry, after editing and rounding.
FinalHours	Double	The calculated hours based on FinalOn and FinalOff.
ScheduleDate	DateTime	Date value of the clock/final on time in Location timezone.
ProjectCode	String	Code for project.
TaskCode	String	Code for task.
ActivityCode	String	Code for activity.
Location1	String	Location identifier of clock on – might be IP address or phone number, for example.
Location2	String	Location identifier of clock off – might be IP address or phone number, for example.
IsError	SmallInt	Flag to indicate that the entry is INVALID. (These are never exported to payroll.)
IsLeave	SmallInt	Flag to indicate that project/task is set as leave.
IsBillable	SmallInt	Flag to indicate that project/task is set as billable.
IsPaid	SmallInt	Flag to indicate that project/task is set as paid.
CreatedDate	DateTime	UTC timestamp of entry creation.
UpdatedDate	DateTime	UTC timestamp of last update to entry.
UpdateUserType	String	Type of user making last update.
UpdateUserId	Integer	Internal id of user making last update
IsEmployeeAlert	Byte	Employee-activated flag to alert manager of issues with entry.
EmployeeNotes	String	Free text field for employee-entered notes.
UserNotes	String	Free text field for manager-entered notes.
BreakHours	Double	Hours to removed (as unpaid) from FinalHours.
ClockOnByUserId	Integer	Internal Id of manager, if clocking on for employee.
ClockOffByUserId	Integer	Internal Id of manager, if clocking off for employee.
PayRate	Double	Calculated from Employee/Rate Matrix
BillRate	Double	Calculated from Employee/Rate Matrix
IsReviewed	SmallInt	Flag to indicate that an IsError entry has been reviewed.
PayPeriodId	Integer	Internal Id of pay period (if it exists).
EmployeeId	Integer	Internal Id of the employee.

IsBreak	SmallInt	Flag to indicate that project/task is set as break time.
ClockReason	String	Optional reason code selected by employee when certain clocking conditions exist.
IsOvertimeExcluded	SmallInt	Flag to indicate that project/task is set as non -overtime.
BusinessName	String	Name of the location.
BusinessExternalId	String	External Id of the location.
LastName	String	Employee's last name.
FirstName	String	Employee's first name.
EmployeeExternalId	String	External Id of the employee.
LeaveTypeCode	String	Code of leave type, if entry linked to a leave entry.
LeaveTypeName	String	Name of leave type, if entry linked to a leave entry.
LeaveTypeExternalId	String	External Id of leave type, if entry linked to a leave entry.
ProjectName	String	Name of project.
TaskName	String	Name of task.
ActivityName	String	Name of activity.
EmployeeUDF{1-6}	String	(6 fields) – Employee user-defined values.

Filters

Required Filter Values: **MinDate, MaxDate**

Updates

The UpdateCollection method supports the following actions with the given conditions:

ACTION	Required Identifiers	Required Fields
Add	**EmployeeExternalId (or EmployeeId, BusinessExternalId (or BusinessId),**	**ProjectCode, TaskCode, FinalOn, FinalOff**
Update	**Id**	

Support

For questions or issues related to the GenericIO Web Service, please contact support@schedulesource.com or open a ticket at https://helpdesk.schedulesource.com

APPENDIX A – XSLT Transformations

The following xslt templates will convert the GetCollection xml response to more specific xml formats. The first transformation creates an xml document with no attributes. The EntityType becomes the top tag and the field values are child tags. The second transformation creates Tags for each entity based on EntityType with attributes for the field values.

XSLT to convert response to all elements w/o attributes:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!-- Elements -->
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/">
       <xsl:for-each select="GetCollectionResult/Entity">
       <xsl:element name="{/GetCollectionResult/@entitytype}" >
              <xsl:for-each select="Field">
              <xsl:element name="{@n}">
              <xsl:value-of select="@v" />
              </xsl:element>
              </xsl:for-each>
       </xsl:element>
       </xsl:for-each>
</xsl:template>
</xsl:stylesheet>
```

XSLT to convert response to elements with attributes:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!-- Element with Attributes -->
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/GetCollectionResult">
       <xsl:for-each select="Entity">
       <xsl:element name="{/GetCollectionResult/@entitytype}" >
              <xsl:for-each select="Field">
               <xsl:attribute name="{@n}" >
              <xsl:value-of select="@v" />
              </xsl:attribute>
              </xsl:for-each>
       </xsl:element>
       </xsl:for-each>
</xsl:template>
</xsl:stylesheet>
```