Edilion Uniconta REST API

The Edilion Uniconta API is a REST API built in order to facilitate technology-independent integration with the Uniconta ERP System. The REST API relies on the official Uniconta C# API to provide its functionality.

The sections of this paper follow the structure of the C# API manual to describe how the same goals can be achieved using the REST API presented here.

Connection

The REST API uses a HTTP Basic Authentication mechanism to ensure the authorization and authentication of a client in Uniconta. Therefore, every HTTP request sent to the service must include the `Authorization` header containing the Uniconta credentials as shown below

```
Authorization: <uniconta username>:<uniconta password>
```

In addition, every HTTP request must include the the Edilion credentials as shown below

```
edilion-authenticate: <edilion username>:<edilion password>
```

CRUD on entities

The format used by the REST API for both request and response bodies (where not empty) is JSON. This means that all entities are represented as JSON objects, having a structure identical to that of the original Uniconta C# entity classes (i.e. for every class attribute/property in C# there is a corresponding key-value pair in JSON, where the JSON key is the attribute/property field's name in C# and the corresponding JSON value is the field's value).

CRUD operations are supported on all Uniconta entities and can be executed using HTTP requests. The type of the operation is specified by the HTTP method used for the request. The table below shows the mapping between the CRUD operations to the HTTP methods supported.

Operation	**HTTP method**
Create	PUT
Read	GET
Update	POST
Delete	DELETE

All requests that have a body can also specify a value for the `Content-Type` header. The legal values for this header are `application/json`, `text/plain` or `application/pdf`. While `application/json` and `text/plain` can be used interchangeably or even ignored, the value `application/pdf` **must** be used when sending binary data (more information regarding this case in the Update section)

The following two sections describe these operations in detail on *Simple* and *Master/Detail* entities respectively.

Simple entities

An entity is *simple* if it is not part of an aggregation relationship with any other entity in Uniconta (as the contained entity). In other words, a simple entity is not considered to be *part of* any other entity. Most of the Uniconta entities are simple entities.

The following four sub-sections show how to apply the Read, Create, Update and Delete operations on simple entities, respectively.

Read

To read all entities of one type, an HTTP GET request must be sent to a URL of the following form

```
https://login.edilion.com/a/api/uniconta-api/<entity name>
```

The response will be an array containing the JSON representation of all the entities of the specified type.

Example

A GET request to `https://login.edilion.com/a/api/uniconta-api/Creditor` will return a JSON array with all the vendors in Uniconta.

Additional filtering can be added using URL-encoded parameters

```
https://login.edilion.com/a/api/uniconta-api/<entity name>?
<field1>=<value1>&<field2>=<value2>&...
```

Example

A GET request to `https://login.edilion.com/a/api/uniconta-api/Creditor?_Group=Ydelser` will return a JSON array with all vendors in the group “Ydelser”.

In both cases mentioned above, if there are no entities to return, the body of the response will be an empty JSON array (“[]”).

To read one specific entity, the URL must indicate the ID of the entity being requested (the ID is the value of the `RowId` C# property).

```
https://login.edilion.com/a/api/uniconta-api/<entity name>/<entity
id>
```

In this case, the response will contain a single JSON object representing the entity. If the requested entity is not found, an empty HTTP response will be returned with the status code `404 Not Found`.

Create

In order to create a new entity, a PUT request must be sent to a URL of the form

`https://login.edilion.com/a/api/uniconta-api/<entity name>`

The body of the request must contain a JSON representation of the new entity (the structure of an entity can be obtained by reading an entity of the same type). Note that this JSON does not need to include all the fields available for an entity in Uniconta. However, it must include the minimum subset of fields required by the Uniconta ERP Server in order to save the new entity. If the operation succedes, the API returns the full JSON representation of the newly created entity.

Update

Similarly, an entity can be updated using a POST request containing the JSON representation of the entity as its body. The POST request must be sent to a URL indicating the entity being updated by its ID

`https://login.edilion.com/a/api/uniconta-api/<entity name>/<entity id>`

Example

A POST request to `https://login.edilion.com/a/api/uniconta-api/Creditor/42` with the body

`{ “_Phone”: “34657524” }` will update the phone number of the vendor with ID 42. All other fields remain unmodified.

As shown in the example above, when updating an entity it is enough only to specify the fields that are to be updated in the body of the request (instead of all the fields available for that entity). All other fields will remain unmodified. If the operation succeeds, the API will return en empty HTTP response with the status code `200 OK`.

In addition to this, if there is only one field that needs to be updated, the fields can be specified in the URL as below

`https://login.edilion.com/a/api/uniconta-api/<entity name>/<entity id>/<field name>`

In this case the body of the request must be a value (the new value of the field to update) instead of a JSON structure.

Example

A POST request to `https://login.edilion.com/a/api/uniconta-api/Creditor/42/_Phone` with the body

`34657524` will update the phone number of the vendor with ID 42.

Physical image submission

This functionality is particularly useful when submitting a document to Uniconta together with a physical image (e.g. a PDF file). After submitting the Document JSON structure using a PUT request (without the physical image), a PDF can also be submitted by updating the newly created document's `_Data` field. This can be accomplished using a POST request with the `Context-Type` header set to `application/pdf` and the body containing the binary representation of the image file. This two-step approach to submitting a document with a physical image was preferred to the alternative of using a one-step multipart request.

Delete

To delete an entity, a DELETE request (without body) must be sent to a URL indicating which entity to delete as below

```
https://login.edilion.com/a/api/uniconta-api/<entity name>/<entity
id>
```

If the operation succeeds, the API will return en empty HTTP response with the status code `200 OK`.

Master/Detail entities

Several Uniconta entities are part of aggregation relationships. The relation between an entity that contains other entities and those being contained is called a Master/Detail relation.

The only difference between a simple entity and a detail entity is the way it is created. The way to create the detail entities is to include the master type and the ID of a concrete master entity in the URL.

```
https://login.edilion.com/a/api/uniconta-api/<master entity
name>/<master  entityid>/<detail entity name>
```

Example

For instance, when creating a Journal Line one also has to specify which Journal the new Journal Line will belong to. In this case, a PUT request could be sent to

```
https://login.edilion.com/a/api/uniconta-
api/GLDailyJournal/1/GLDailyJournalLine
```

the outcome being that a new Journal Line is created in the Journal with ID 1.

Note that create is the only operation that requires the master to be specified for a detail entity. All other operations (read, update, delete) must be executed without specifing a master (i.e. as presented in the previous section). This is due to the fact that the ID of a detail entity is ensured to be unique across detail entities of all master entities.

Example

When updating the Journal Line with ID 3, a POST request must be sent to

`https://login.edilion.com/a/api/uniconta-api/GLDailyJournalLine/3`

Enumerations

Besides the entities, Uniconta also defines a set of static Enumerations. The values of the enumerations are referred from the entities' fields by their index number within the enumeration.

The values of one enumeration can be obtained by sending a GET request to a URL of type

```
https://login.edilion.com/a/api/uniconta-api/Enum/<enum name>
```

The response will be a JSON array containing all the values in the enumeration.

Example

A GET request to `https://login.edilion.com/a/api/uniconta-api/Enum/CostType` will return the JSON array `[“Fixed”, “Average”, “FIFO”]`.

Further, the index of one value within an enumeration can be obtained by sending a GET request to a URL of type

```
https://login.edilion.com/a/api/uniconta-api/Enum/<enum name>/<enum
value>
```

In this case, the response will simply be a number representing the required index.

Example

A GET request to `https://login.edilion.com/a/api/uniconta-api/Enum/CostType/Average` will return the value `1`.

Companies

Uniconta offers the possibility to manage multiple companies from one account. In addition, one of these company can be marked as the **default** company. All endpoint presented above operate on the company marked as default (since the reference to the Company is omitted). However, a company can be explicitely specified in all endpoints as shown below:

```
https://login.edilion.com/a/api/uniconta-api/Company/<company
id>/<endpoint>
```

For instance, retrieving Document with id 5 in company with ID 180 can be achieved by sending a GET request to the following endpoint

```
https://login.edilion.com/a/api/uniconta-api/Company/180/Document/5
```

The company can be prepended to all endpoints as shown above, with the following exceptions:

- *Operations on the Company entity itself*. When creating, reading, updating or deleting Companies, only the Company on which the operation applies must be specified (since a Company cannot be contained by another Company).
- *Reading enumerations / enumeration values*. Enumerations are static and therefore endpoints reading enumerations must not contain any information regarding the Company.

HTTP Status Codes

All HTTP responses from the endpoints described above include a status code which can be interpreted as shown below

HTTP Status Code	**Cause**
200 OK	Operation succeeded.
401 Unauthorized	Login to Uniconta failed. No credentials or wrong credentials were provided.
404 Not Found	The entity to apply the operation on does not exist or endpoint not found. The exception is the DELETE operation, where if the entity to be deleted does not exists 200 OK will be returned instead of 404 Not Found.
405 Method Not Allowed	The HTTP request was not supported (other than GET, POST, PUT or DELETE).
500 Internal Server Error	An error occurred internally. Check the body of the response for more details.

Available entities & enumerations

The endpoints described above are generic, in the way that they can be used in the exact same way to manipulate any of the entities or enumerations available in Uniconta (this approach is very similar to the original Uniconta C# API). Below are the lists of available entities and enumerations in Uniconta.

Entities:

```
Accountant
AccountMethod
AccountSum
AccountSumContext
Balance
BalanceColumn
BankImportFormat
BankImportMap
```

```
BankStatement
BankStatementLine
BankStatementLineToTrans
Company
CompanyDataCount
CompanyDocument
CompanyDocumentLayout
CompanyFinanceYear
CompanyNumber
CompanyPaymentSetup
CompanySettings
CompanySetup
CompanyUserAccess
Contact
Creditor
CreditorAgeTotal
CreditorGroup
CreditorInvoice
CreditorOrder
CreditorOrderLine
CreditorTrans
CreditorTransOpen
DCAccount
DCAgeTotal
DCGroup
DCInvoice
DCOrder
DCOrderLine
DCOrderLineSerieBatch
DCTrans
DCTransOpen
Debtor
DebtorAgeTotal
DebtorEmailSetup
DebtorGroup
DebtorInvoice
DebtorOffer
DebtorOfferLine
DebtorOrder
DebtorOrderLine
DebtorTrans
DebtorTransOpen
Distributor
Document
DocumentFolder
DocumentNoRef
Employee
EmployeeJournalLine
EmpPayrollCategory
EmpPayrollCategoryEmployee
EmpPayrollLine
ExchangeRate
GLAccount
GLBudget
GLBudgetLine
GLClosingSheet
GLClosingSheetLine
GLDailyJournal
GLDailyJournalLine
GLDailyJournalPosted
GLDimType
GLDimType1
GLDimType2
GLDimType3
GLDimType4
```

```
GLDimType5
GLReportLine
GLReportTemplate
GLSplitAccount
GLSplitLine
GLSplitTemplate
GLTrans
GLTransSum
GLTransType
GLVat
GLVatType
InvBOM
InvGroup
InvItem
InvItemNameGroup
InvItemStorage
InvItemText
InvJournal
InvJournalLine
InvLocation
InvPriceList
InvPriceListLine
InvSerieBatch
InvSerieBatchOpen
InvStandardVariant
InvStandardVariantCombi
InvTrans
InvVariant
InvVariant1
InvVariant2
InvVariantCombi
InvWarehouse
Note
NumberSerie
Partner
PartnerDLL
PaymentTerm
PrCategory
PrJournal
PrJournalLine
PrJournalPosted
Project
ProjectBudget
ProjectBudgetCategorySum
ProjectBudgetCategoryTypeSum
ProjectBudgetLine
ProjectBudgetSum
ProjectCategory
ProjectCategoryCharge
ProjectCostCategoryGroup
ProjectGroup
ProjectInvoiceLine
ProjectOnAccountInvoiceLine
ProjectTrans
ProjectTransCategorySum
ProjectTransCategoryTypeSum
ProjectTransSum
PrStandard
PrStandardCategory
PrStandardCategoryCharge
PrType
Reseller
Subscription
SubscriptionCompany
SubscriptionInvoice
```

SubscriptionInvoiceLine
SumMethod
SupportLine
SupportTicket
TableAddOnData
TableCache
TableChangeLog
TableData
TableDataWithKey
TableField
TableFieldDataRow
TableHeader
TableProperty
TableTemplate
User
UserLayout
UserLogin
UserOperationLog
UserPlugin
UserReport
UserReportCombit
UserReportDevExpress
VatCacheFilter
VatTypeSQLCacheFilter
Withholding

Enumerations:

BalanceColumnFormat
BalanceColumnMethod
BOMMoveType
BOMQtyType
BudgetRecurring
CacheUpdateAction
CategoryType
CompanyDocumentUse
CompanyLayoutType
CompanyNumberType
CompareOperator
ContactTitle
ContentTypes
CostType
CountryCode
CountryISOCode
Currencies
CustomTypeCode
DateFormat
DCPostType
DebtorEmailType
ErrorCodes
FileextensionsTypes
FinancePeriodeState
GLAccountTypes
GLDailyJournalPostedReference
GLJournalAccountType
GLJournalDate
GLVatCalculationMethod
GLVatSaleBuy
InvMovementType
ItemPriceCalcMethods
ItemType
ItemUnit
Language
Layouts
MessageType
OrderType
PaymentMethodTypes

```
PaymentTypes
PayrollType
PrCategoryChargeType
ProjectPhase
SelectType
SettleValueType
StorageRegister
SubscriptionProduct
SupportType
SystemAccountTypes
VatOptions
VatZones
```

Notes

- Before developing using the API, we recommend getting familiar with it using a REST client such as Postman.

Current limitations

- The current version of the API assumes that there is at least one company in the Uniconta account.

Possible pricing model

- One-time fee of 500kr per entity access
- A monthly fee of 10kr per customer

Usage

When acquiring access to an entity type, the following are provided by Edilion:

- A full JSON representation of an instance of the type (first example below).
- cURL examples demonstrating all CRUD operations on entities of that type (second example below).

Example

A GET request to retrieve all Journals can be fired with the following cURL command

```
curl -X GET -H "Authorization: username:password"
"https://login.edilion.com/a/api/uniconta-api/GLDailyJournal"
```

Example

For the Journal (GLDailyJournal) entity, such a JSON object could be

```
{
   "_Journal": "Bank",
   "_Name": "Indlæs bankkontoudtog",
  "_Account": null,
  "_OffsetAccount": "5820",
  "_Vat": null,
  "_OffsetVat": null,
  "_NumberSerie": "NR",
  "_TransType": null,
  "_NextLineField": null,
  "_Dim1": null,
  "_Dim2": null,
  "_Dim3": null,
  "_Dim4": null,
  "_Dim5": null,
  "_TraceAccount": "5820",
  "_TraceAccount2": null,
  "_TraceAccount3": null,
  "_TraceAccount4": null,
  "_TraceAccount5": null,
  "_TraceAccount6": null,
  "KeyStr": "Bank",
  "KeyName": "Indlæs bankkontoudtog",
  "PropertyOfKeyStr": "Journal",
  "_DefaultAccountType": 0,
  "_DefaultOffsetAccountType": 0,
  "_Blocked": false,
  "_DeleteLines": true,
  "_GenerateVoucher": true,
  "_OneVoucherForAll": false,
  "_DateFunction": 0,
  "_ManualAllocation": false,
  "_TwoVatCodes": false,
  "_EmptyAccountOnHold": true,
  "_SettleValue": 0,
  "CompanyId": 753,
  "RowId": 1742
 }
```