LUCITY REST API

INTRODUCTION AND CORE CONCEPTS

SHARPEN your TOOLS

lucity act16

REST API OFFERINGS

- Lucity Citizen Portal REST API
- Lucity REST API
- Both products are included in our REST API
- Historically we also offered a COM API and a .NET API
  - COM API was deprecated with 2015r2
  - .NET API should not be used for new development
  - Clients and developers with products developed with the .NET API should plan for a transition to the REST API (we can help!)
- Lucity Import and Update is also a great resource in lieu of custom coding

LUCITY CITIZEN PORTAL REST API

- Citizen facing interfaces
- Anonymous access
- Only services publically available data (a flag you control on a per record basis)
- Protects Personally Identifiable Data (PII) in return responses
- Restrictive access
  - Get requests and limited request child information
  - Create requests and limited request child information
  - Uploading documents
- Facilitates some automated coordinate translation between Geographic (lat/long) and UTM (Mercator) and state plane systems

LUCITY REST API

- Full featured (almost every Lucity module)
- Full support (get, add, edit, delete)
- Requires authentication using a Lucity login
- Targeted at internal facing applications including:
  - Custom GIS applications
  - Custom Mobile applications
  - Integrations
  - Other custom internal development

BOTH PROVIDE LUCITY BUSINESS MODEL FUNCTIONALITY

- They enforce business rules, defaulting, and validation
- They apply special module behavior
  - Updating last inspection dates
  - Updating tracking records for changes
  - Calculations
  - Notifications are sent
  - Etc

REST API SUPPORT AND LICENSING

- REST API is a product
- It is sold as a site license (only)
- The license includes both REST API products
- It is a separately installed application (two actually)
- Support is developer support
  - Our developers support your developers

REST API

CORE CONCEPTS

TRADITIONAL REST API

- Lucity follows a traditional REST approach

TRADITIONAL REST API

- Lucity follows a traditional REST approach
- Every resource has a uri

- All work orders
- http://restapi.lucity.net/LucityRESTAPI/work/workorder/
- A single work order
- http://restapi.lucity.net/LucityRESTAPI/work/workorder/**1**
- First 500 open work orders that were created or the status changed in the last 30 days
- http://restapi.lucity.net/LucityRESTAPI/work/workorder**/?statusflag=open&statusdays=30&take=500**
- Work order based on work order number
- http://restapi.lucity.net/LucityRESTAPI/work/workorder/**?CommonId=2016-00001**
- Sewer Pipe based on a pipe number
- http://restapi.lucity.net/LucityRESTAPI/sewer/pipeinventory/**?CommonId=12041**

TRADITIONAL REST API

- Lucity follows a traditional REST approach
- Every resource has a uri
- Resources can support one or more actions
  - GET, PUT, POST, DELETE

TRADITIONAL REST API

- Lucity follows a traditional REST approach
- Every resource has a uri
- Resources can support one or more actions
  - GET, PUT, POST, DELETE
- Data is accepted and returned in 2 formats:
  - xml (default)
  - json

XML AND JSON FORMAT

- Use the format query parameter to specify which format you want
- ?format=json, ?format=xml
- Even on requests without a response you should include this because it will dictate the format of the error response text if there is one
- Default is xml, if you do not provide a format query parameter, you will get xml
- For requests that include content (POST, PUT) you must also include a content type in the header
  - Content-Type: application/json
  - Content-Type: application/xml
- json is unordered
- Xml is ordered (you must adhere to the order of the xml schema, which is alphabetical for all core properties)
- Because of this, we recommend json over xml

TRADITIONAL REST API

- Lucity follows a traditional REST approach
- Every resource has a uri
- Resources can support one or more actions
  - GET, PUT, POST, DELETE
- Data is accepted and returned in 2 formats:
  - xml (default)
  - json
- Return responses adhere to web standards
  - Status Codes….

HTTP STATUS CODES (MOST)

- 200 – OK (Congratulations! It worked)
- 201 – Created (Success! We created a record for you)
- 204 – No Content (we probably just deleted something for you, and we succeeded but there is no data we are going to return to you other than the status)
- 304 – Not Modified (you told us only to return data if it had changed since you last got it, and it has not)
- 400 – Bad Request (you made a mistake (or we did), sorry, but we will try to tell you what the problem is in the response)
- 401 – Unauthorized (you need to login, or login again)
- 403 – Forbidden (sorry your security does not allow this or your administrator has forbidden it)
- 404 – Not Found

ERROR RESPONSES

- Most 400 (and some other 4x) errors return helpful responses

```
{
    "ApplicationErrorCode": 400
    "Description": "Cannot save, object is invalid: WorkOrder: CategoryCode is required."
}
```

```
<CustomErrorMessage>
   <ApplicationErrorCode>400</ApplicationErrorCode>
   <Description>Cannot save, object is invalid: WorkOrder: CategoryCode is required.</Description>
</CustomErrorMessage>
```

- Show these to your users (or log them if it is a headless interface)
- By default we do not log these messages in our logs

TRADITIONAL REST API

- Lucity follows a traditional REST approach
- Every resource has a uri
- Resources can support one or more actions
  - GET, PUT, POST, DELETE
- Data is accepted and returned in 2 formats:
  - xml (default)
  - json
- Return responses adhere to web standards
  - Status Codes….
- Authentication

AUTHENTICATION

- Authentication is not required for certain components (such as citizen facing requests)
- Current authentication technique is Basic Authentication
- Soon we will be moving to Open ID Connect protocol
  - This is the current authentication protocol used by our mobile and web applications
  - This is more secure
  - We will allow clients to keep basic authentication enabled to provide time for transition
- Soon we will be providing a way for clients to grant individual applications private keys.
  - Better control and security

RESOURCES AVAILABLE

- Help guide http://help.lucity.com
- Github repository https://github.com/LucityInc/lucity-restapi-samples
  - Samples
  - Citizen app template
- Service directory
  - Can be disabled starting with Lucity 2016r2
  - At the root of your REST API install http://restapi.lucity.net/LucityRESTAPI
- Support

RECENT API CHANGES

- REST API Changes are included in release notes which are in the install manual http://help.lucity.com/webhelp/latest/install/index.htm#38648.htm
- Some highlights from recent releases:

LUCITY 2015R2

- New PublicallyAvailable flag for comments in the comment grid. This means the Lucity Citizen Portal REST API will now only serve up requests marked publically available. Previously it served all comments on requests marked publically available.

LUCITY 2016

- When a user is not authorized to perform an action due to permissions, the REST API now returns a **403 Forbidden** code. Previously, the system returned a **401 Unauthorized** code. **401** is now used only to indicate a failure to log into ***Lucity*** or an expired login session.

LUCITY 2016R2

- All dates are now returned in ISO8601 UTC format (2016-07-28T16:43:36Z) https://en.wikipedia.org/wiki/ISO_8601
- We now support extensionless endpoints such as
  - **Work/WorkOrder/1212/WorkOrderTaskList/**

  In addition to the older style:

  - **Work/WorkOrder.svc/1212/WorkOrderTaskList/**

  Both styles will be supported for the next couple releases but new development should be done using the extensionless format. Eventually the .svc will no longer be supported.
- The service directory can now be hidden. By default this directory will be hidden. We recommend leaving it hidden except for test or development environments

FUTURE CHANGES

- Security improvements
  - Configurable client application authorization
  - OpenID connect (Identity Server) access tokens instead of basic auth
- Long term we will be combining most of our server components (REST API, Mobile, etc)