



Autodesk Construction Cloud API – Deep Dive

Mikako Harada
Senior Manager, Developer Technical Services AEC

Safe Harbor Statement

The presentations during this event may contain forward-looking statements about our outlook, future results and related assumptions, total addressable markets, acquisitions, products and product capabilities, and strategies. These statements reflect our best judgment based on currently known factors. Actual events or results could differ materially. Please refer to our SEC filings, including our most recent Form 10-K and Form 10-Q filings available at www.sec.gov, for important risks and other factors that may cause our actual results to differ from those in our forward-looking statements.

The forward-looking statements made in these presentations are being made as of the time and date of their live presentation. If these presentations are reviewed after the time and date of their live presentation, even if subsequently made available by us, on our website or otherwise, these presentations may not contain current or accurate information. We disclaim any obligation to update or revise any forward-looking statements.

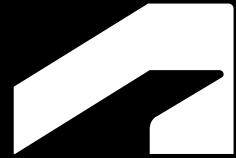
Statements regarding planned or future development efforts for our products and services are not intended to be a promise or guarantee of future availability of products, services, or features but merely reflect our current plans and based on factors currently known to us. Purchasing decisions should not be made based upon reliance on these statements.

PLEASE NOTE: All Autodesk content is proprietary. Please Do Not Copy, Post or Distribute without authorization.

Agenda

- 1 Introduction: Shared Services in ACC
- 2 Relationships API
- 3 Model Properties API
- 4 Common Questions
- 5 What's Next

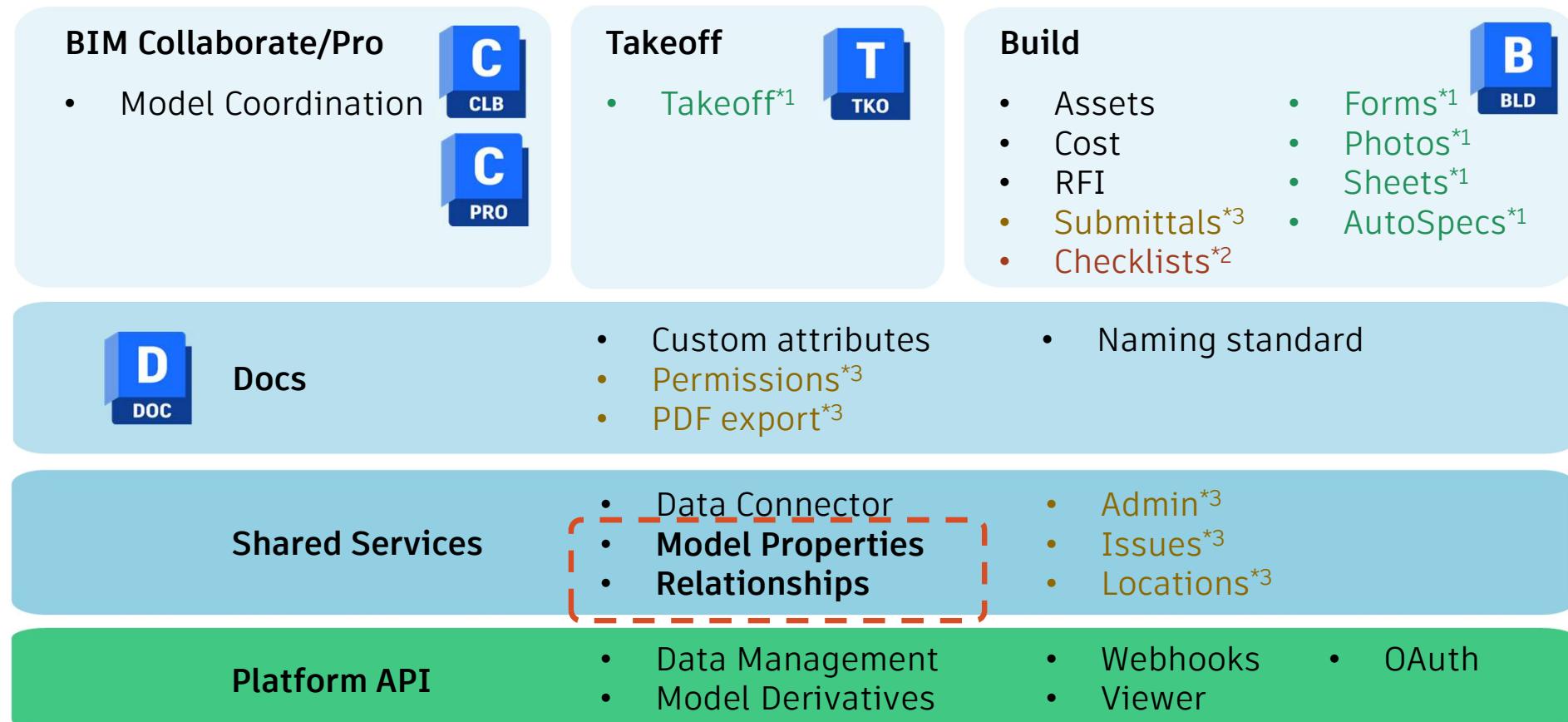




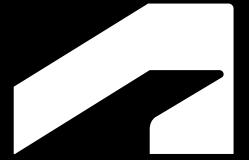
Shared Services in ACC

Shared Services in ACC

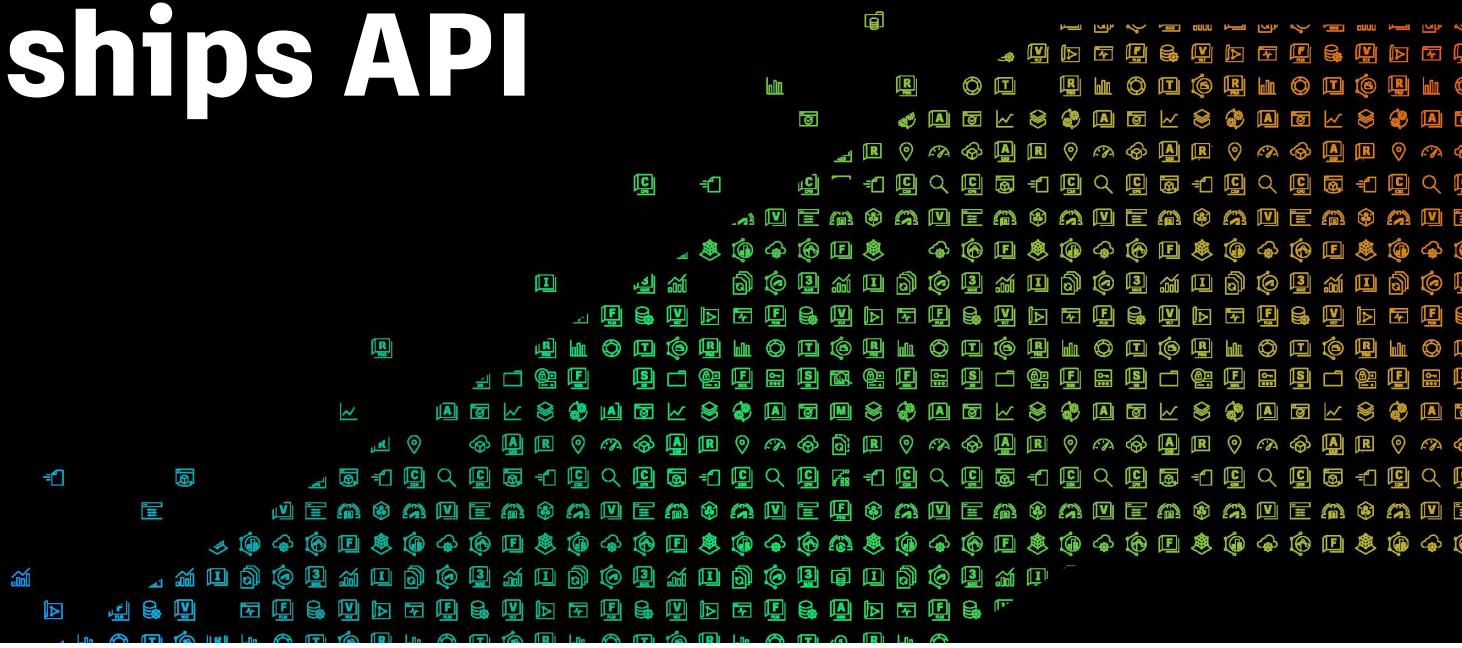
ACC/BIM 360 API Component Services & Structure



*1) ACC only, *2) BIM 360 only, *3) Not compatible or partially compatible

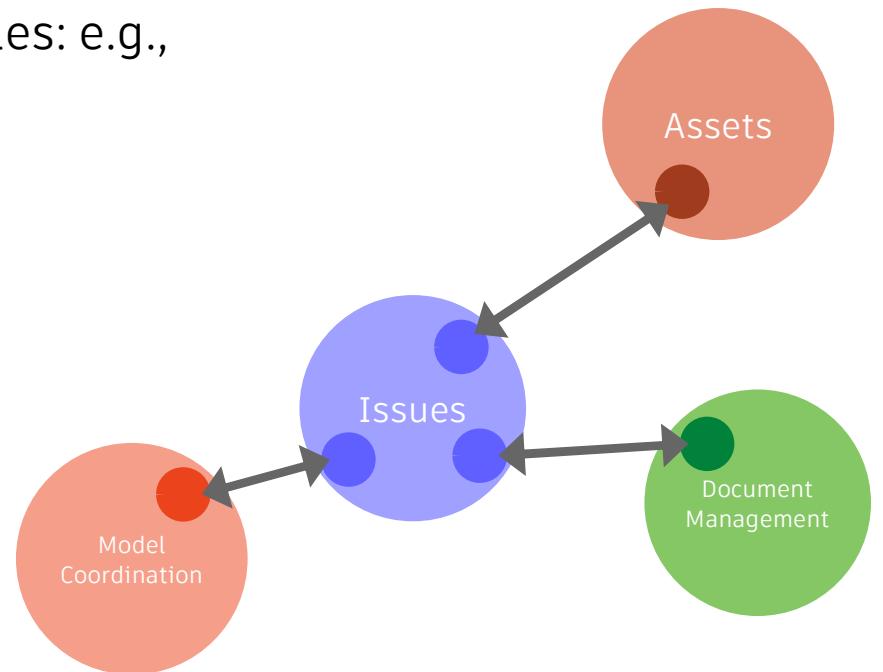


Relationships API



Relationships API

- Represent links between items (Entity) that reside in different functional components (Domain)
- Used in ACC and BIM 360 by various modules: e.g.,
 - Model Coordination, Assets, Forms, Photos, Cost
 - Older implementations may not be using it
- Read & Write
- Publicly available today



Relationships API

The screenshot shows the Autodesk Construction Cloud Assets interface. On the left, there's a sidebar with various icons and a navigation menu. The main area is titled "Assets" and shows a list of 11 assets. A red dashed box highlights a specific row in the list, which corresponds to the asset detailed on the right.

Asset Details (Right Panel):

Asset ID: 625048
Category: ... > Lighting and Ap...
Status: Specified
Linked references: -
Description: 250 A
Location: Level 1 > L

References Tab (Right Panel):

Details References Activity log

References

Add references ^

Files

O&M

Issues

#53 - Leah

Photos

Submittals

Leah

Photos

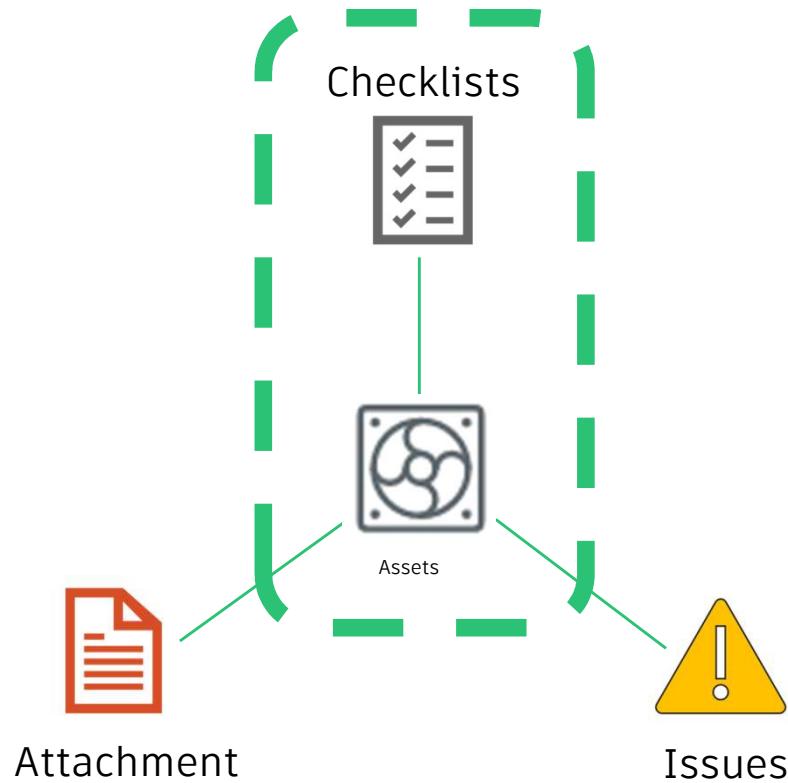
A photograph of a large electrical panel or equipment unit.

Category	Status	Linked references	Description	Location
... > Lighting and Ap...	Specified	-	250 A	Level 1 > L
... > Lighting and Ap...	Specified	-	400 A	
... > Lighting and Ap...	Pre-Start-Up	3 references	100 A	Level 1 > L
... > Lighting and Ap...	Specified	-	100 A	
... > Lighting and Ap...	Installed	3 references	400 A	Level 1
... > Lighting and Ap...	Ordered	-	400 A	
... > Lighting and Ap...	Installed	-	600 A	

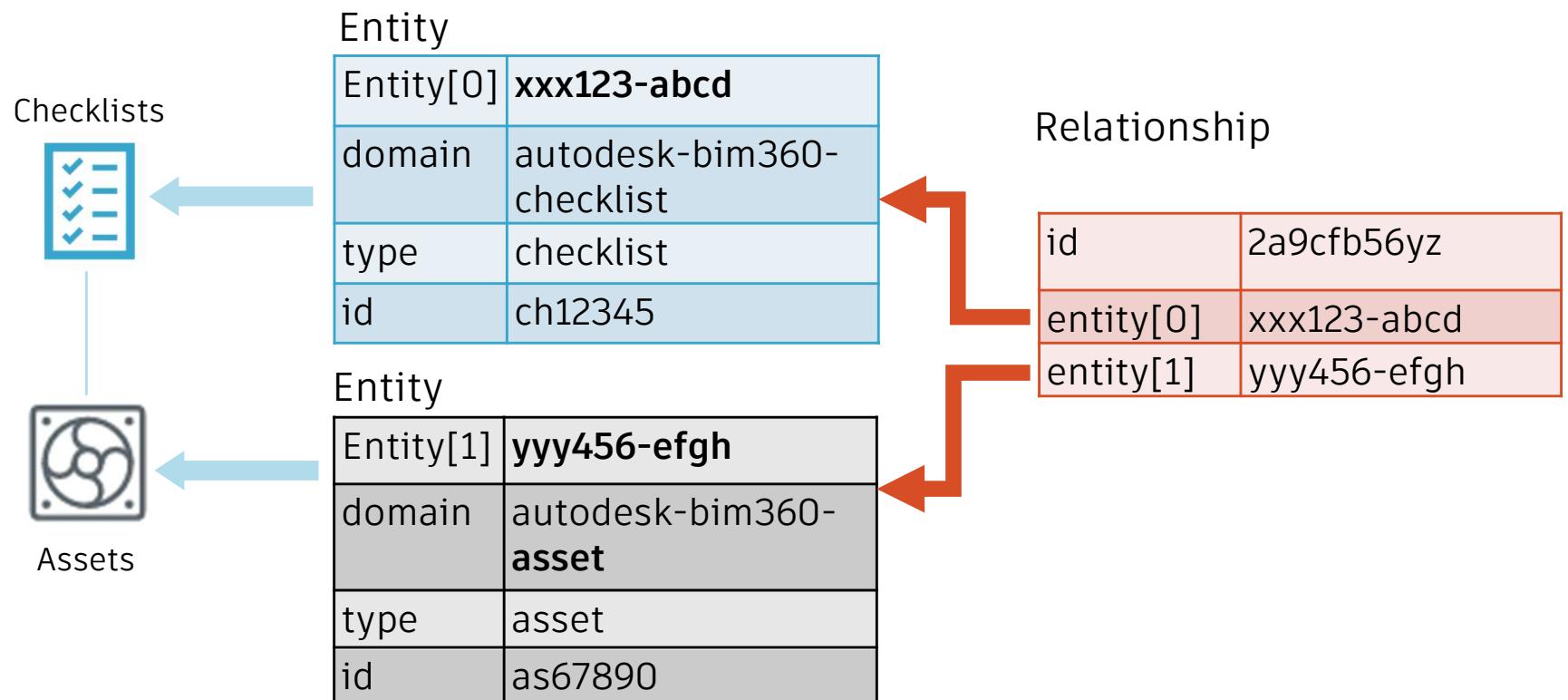
Showing 11 assets

Relationships API

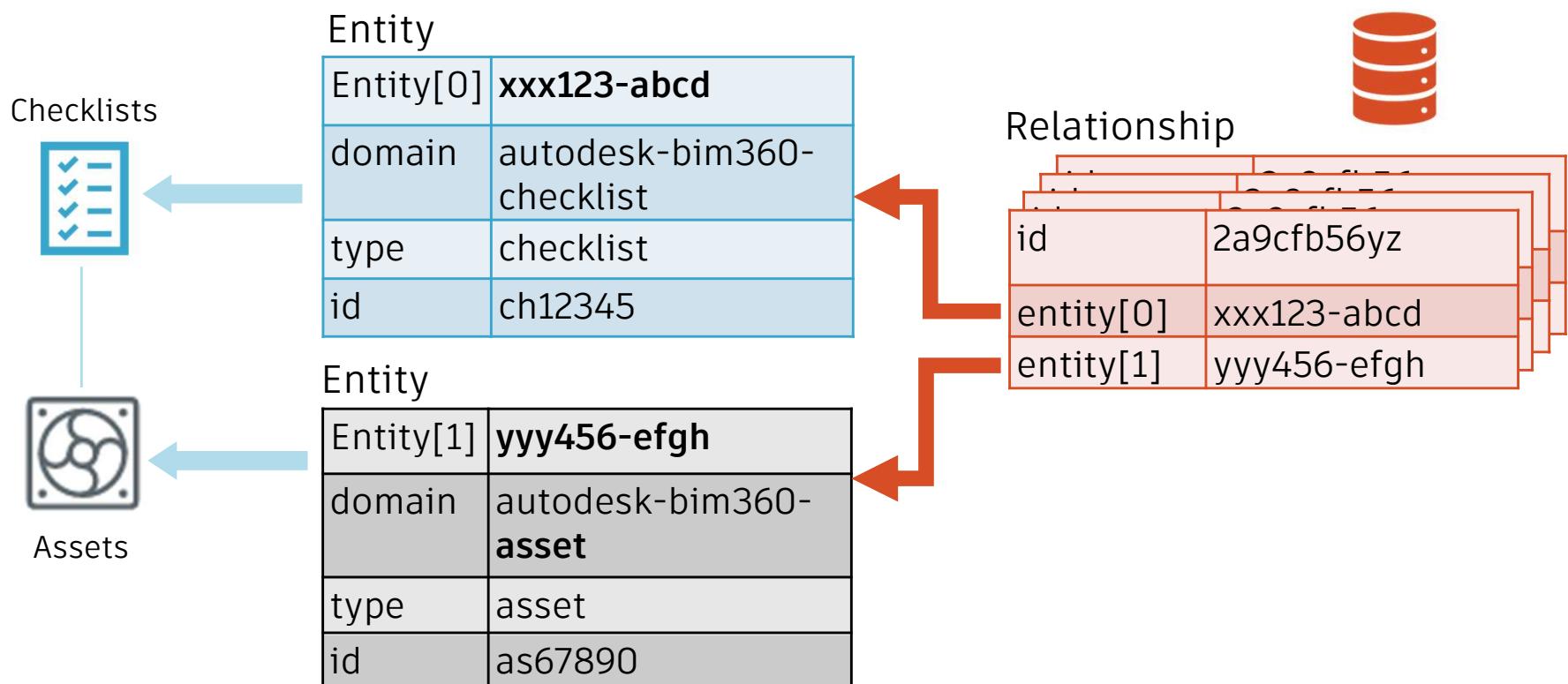
Example



Relationships API



Relationships API



Relationships API

Category	Endpoints
Search	GET relationships: :relationshipId
	POST relationships: batch
	GET relationships: search
	POST relationships: intersect
Sync	POST relationships: sync
	POST relationships: syncStatus
Utilities	POST relationships: writable
Modify	PUT relationships
	POST relationships: delete

Save and search relationships independent from the implementation detail of each components

9 endpoints

Relationships API

Category	Endpoints
Search	GET relationships: :relationshipId
	POST relationships: batch
	GET relationships: search
	POST relationships: intersect
Sync	POST relationships: sync
	POST relationships: syncStatus
Utilities	POST relationships: writable
Modify	PUT relationships
	POST relationships: delete

Get relationship(s) from id(s)

JSON

```
[{"id": "79835e9e-ab4c-4c43-b480-a7ba54196e6e", "createdOn": "2020-10-03T20:53:14.268952+00:00", "isReadOnly": true, "isService": true, "isDeleted": false}, {"entities": [{"id": "569859", "type": "checklist", "domain": "autodesk-bim360-checklist", "createdOn": "2020-10-03T20:53:14.175205+00:00"}, {"id": "f972c095-d6b2-4fd3-ae1c-70ef6061c8af", "type": "asset", "domain": "autodesk-bim360-asset", "createdOn": "2020-10-03T20:49:21.651508+00:00"}]}
```

Relationships API

Category	Endpoints
Search	GET relationships: :relationshipId
	POST relationships: batch
	GET relationships: search
	POST relationships: intersect
Sync	POST relationships: sync
	POST relationships: syncStatus
Utilities	POST relationships: writable
Modify	PUT relationships
	POST relationships: delete

Search rel's that match the given criteria (domain, type, ids, date). e.g.,

“Give me all the relationships between assets and checklists”

```
GET relationships:/search?  
domain=autodesk-bim360-asset  
&withDomain=autodesk-bim360-  
checklist
```

Relationships API

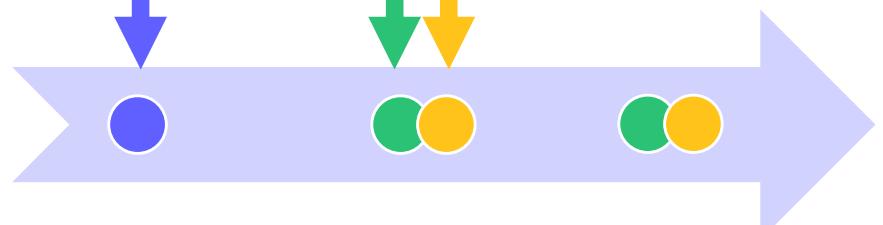
Category	Endpoints
Search	GET relationships: :relationshipId
	POST relationships: batch
	GET relationships: search
	POST relationships: intersect
Sync	POST relationships: sync
	POST relationships: syncStatus
Utilities	POST relationships: writable
Modify	PUT relationships
	POST relationships: delete

Used to sync rel. data and external repository.

sync
(initialize)

syncStatus (more
data?)

sync
(syncToken)



Relationships API - Additions

Category	Endpoints
Search	GET relationships: :relationshipId
	POST relationships: batch
	GET relationships: search
	POST relationships: intersect
Sync	POST relationships: sync
	POST relationships: syncStatus
Utilities	POST relationships: writable
Modify	PUT relationships
	POST relationships: delete

- Retrieves **compatible entity types** to create/delete relationships
- Supported domain/entity types:
 - Asset/asset category
 - Document/File lineage/version
 - Issue
 - Form/form field/form template
 - Photo
 - Etc.
- Unsupported types - may have workaround.
- Expected to be extended overtime

Relationships API

Category	Endpoints
Search	GET relationships: :relationshipId
	POST relationships: batch
	GET relationships: search
	POST relationships: intersect
Sync	POST relationships: sync
	POST relationships: syncStatus
Utilities	POST relationships: writable
Modify	PUT relationships
	POST relationships: delete

```
[  
 {  
   "domain": "autodesk-bim360-asset",  
   "entityTypes": [  
     {  
       "entityType": "asset",  
       "allow": [  
         {  
           "domain": "autodesk-bim360-issue",  
           "entityTypes": [  
             "issue"  
           ]  
         },  
         {  
           "domain": "autodesk-bim360-documentmanagement",  
           "entityTypes": [  
             "documentlineage"  
           ]  
         },  
         {  
           "domain": "autodesk-construction-photo",  
           "entityTypes": [  
             "photo"  
           ]  
         },  
         [...]  
       ]  
     }  
   ]  
 }
```

Relationships API

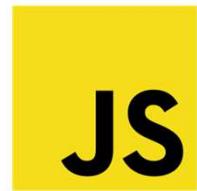
Category	Endpoints
Search	GET relationships: :relationshipId
	POST relationships: batch
	GET relationships: search
	POST relationships: intersect
Sync	POST relationships: sync
	POST relationships: syncStatus
Utilities	POST relationships: writable
Modify	PUT relationships
	POST relationships: delete

- **Create** a relationship(2) between two entities.
 - Two entities are writable
 - Entities must exist
 - must have write access to both entities
- **Delete** a given relationship
 - “Soft” delete
 - must have write access to both entities

Relationships API: Developer Resources

SDK

- .NET Core nuget packages
- JavaScript and Node.js



Relationships API: Developer Resources

Documentation

- Field Guide
- Step by Step Tutorials

The screenshot shows the Autodesk Platform Services Documentation website. The top navigation bar includes links for Solutions, Getting Started, Documentation (which is the active page), Success Stories, Community, Support, Pricing, and App Store. The left sidebar has a 'Developer's Guide' section with a dropdown menu for 'Field Guide' containing Admin, Assets (beta), AutoSpecs, Cost Management, Files, Forms, Issues, Locations, Model Coordination (with Relationships highlighted by a yellow dashed box), Model Properties, Sheets, and Submittals. The main content area displays the 'Relationships' page under the 'Containers vs. BIM 360 Projects' section. It explains that Relationship Service uses 'Containers' as primary data partitions for service data associated with specific Projects or Accounts. It also introduces the concept of 'Domain Entities' and provides two numbered points about them.

Autodesk Platform Services

Solutions ▾ Getting Started Documentation Success Stories Community ▾ Support ▾ Pricing App Store ▾

Developer's Guide

Introduction

Field Guide

- Admin
- Assets (beta)
- AutoSpecs
- Cost Management
- Files
- Forms
- Issues
- Locations
- Model Coordination
 - Relationships
 - Model Properties
- Sheets
- Submittals

Documentation / Autodesk Construction Cloud APIs / Developer's Guide

Relationships

Containers vs. BIM 360 Projects

BIM 360 Data Services e.g. the Relationship Service use “Containers” as a primary data partition for all the service data associated with a specific Project and or Account. In the case of the Relationships Service, a container is created when a new BIM 360 Project is created. The application uses the BIM 360 Docs Project ID as the container GUID for the Relationships container.

Domain Entities

The Relationship Service is built on the concept of *Domain Entities*. Put simply, a Domain Entity is something which can be identified in one of the applications which comprise BIM 360 today. Examples of Domain Entities include Issues, Files, Meetings, Check-lists etc.. Each of these things (or Entities) has three distinct attributes :-

1. They belong to a *Domain* e.g. Budgets belong to the BIM 360 Cost domain (Application).
2. They have an *Entity Type* which allows them to be distinguished from other types of entity in the Domain, e.g. a Budget is distinct from a Contract, however both these entity types belong to the

Relationships API: Developer Resources

Documentation

- Reference Guide

The screenshot shows the Autodesk Platform Services Documentation website. A yellow dashed circle highlights the 'Relationships' section of the left sidebar, which contains links for 'Relationship: Utilities', 'Relationship: Modify', 'Relationship: Sync', and 'Relationship: Search'. The main content area shows the 'utility/relationships:writable' endpoint under 'Relationship: Utilities'. It includes a 'GET' method button, a description of the endpoint, notes about entity type domains, a 'Create a Relationship tutorial' link, and a note about compatibility with BIM 360 and Autodesk Construction Cloud (ACC) projects. Below this is a 'Resource Information' section with links for 'Method and URI', 'Authentication', and 'user context required'.

AUTODESK Platform Services

Solutions Getting Started Documentation Success Stories Community Support Pricing App Store

Sign in

Relationships

Relationship: Utilities

Relationship: Modify

Relationship: Sync

Relationship: Search

Submittals

utility/relationships:writable

PUT relationships

POST relationships:delete

POST relationships:syncStatus

POST relationships:sync

POST relationships:batch

GET relationships:search

POST relationships:intersect

GET relationships/:relationshipId

Documentation / Autodesk Construction Cloud APIs / API Reference

Relationship: Utilities

GET utility/relationships:writable

Retrieves a list of entity types that are compatible with each other, to establish whether you can create relationships between them or to delete those relationships. For example, between an asset and a document.

Note that some entity types belong to a **bim360** domain, and others to a **construction** domain.

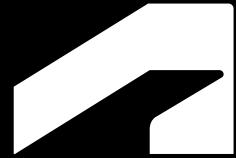
To learn how this endpoint is used, see the [Create a Relationship tutorial](#).

Note that this endpoint is compatible with both BIM 360 and Autodesk Construction Cloud (ACC) projects.

Resource Information

Method and URI **GET** <https://developer.api.autodesk.com/bim360/relationship/v2/utility/relationships:writable>

Authentication user context required



Model Properties API

Model Properties API

Design Data File

Life Cycle

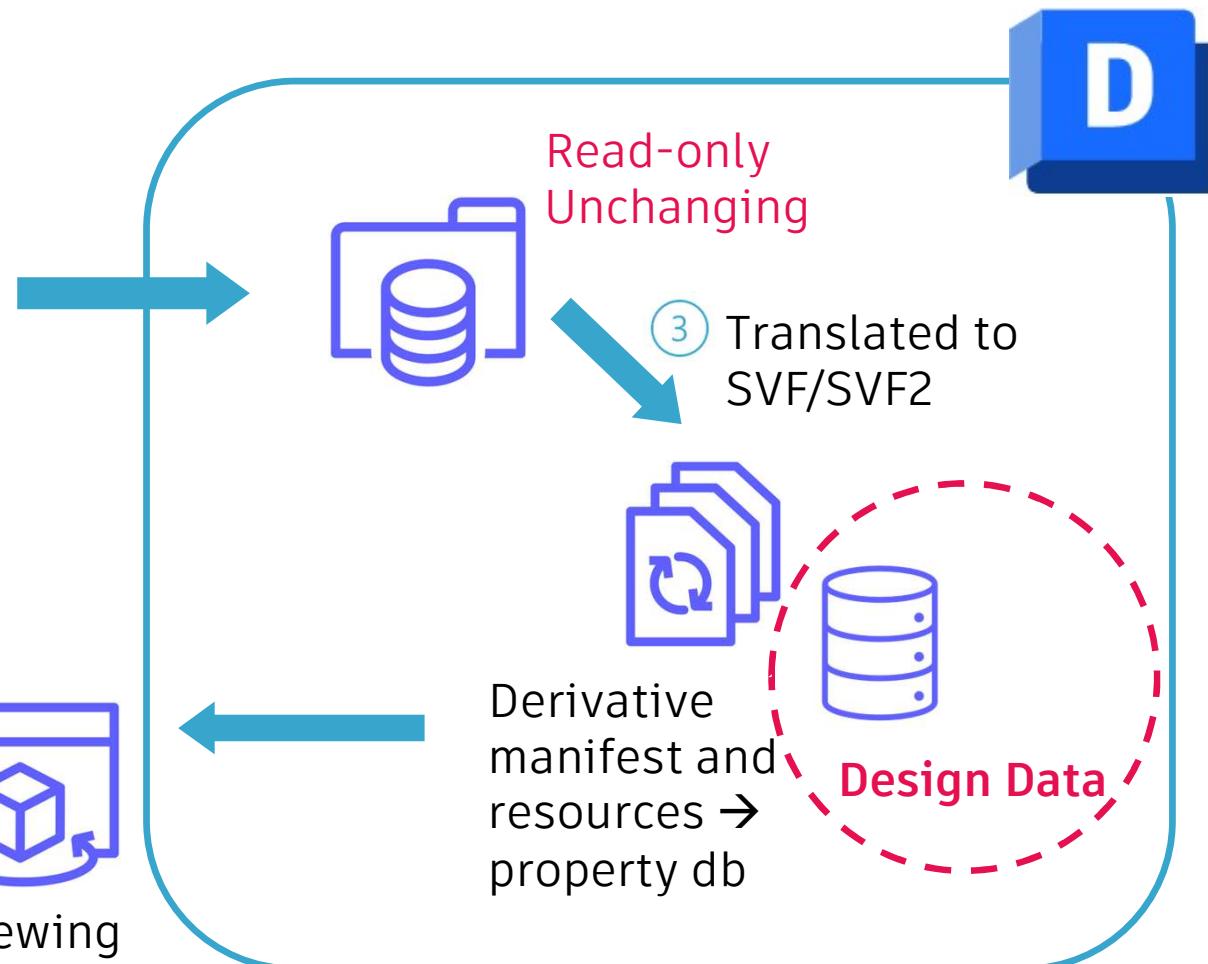
- ① Authoring Design
- ② Upload via web UI, desktop connector or API



- ⑤ Design Change
→ new file version

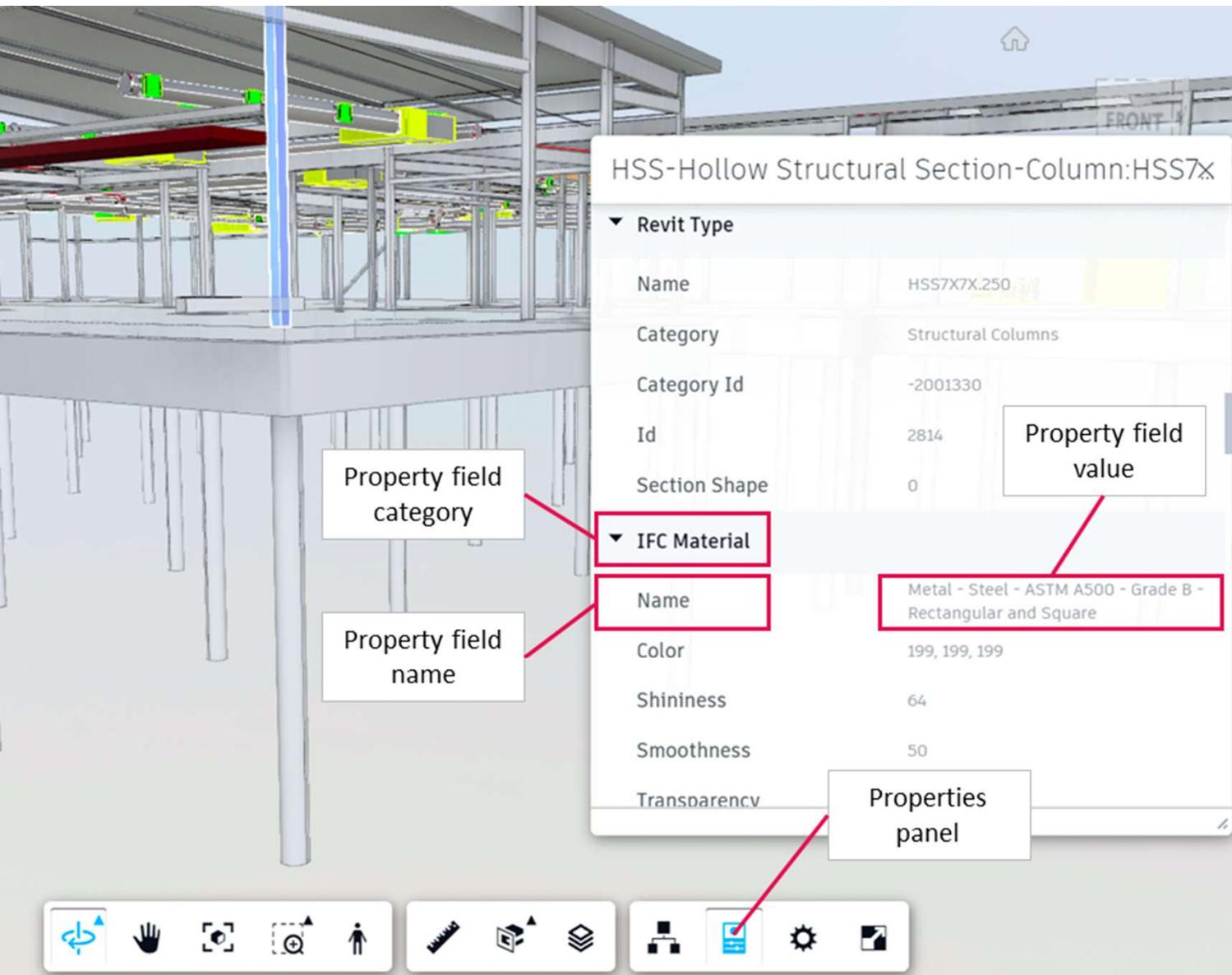


- ④ Viewing



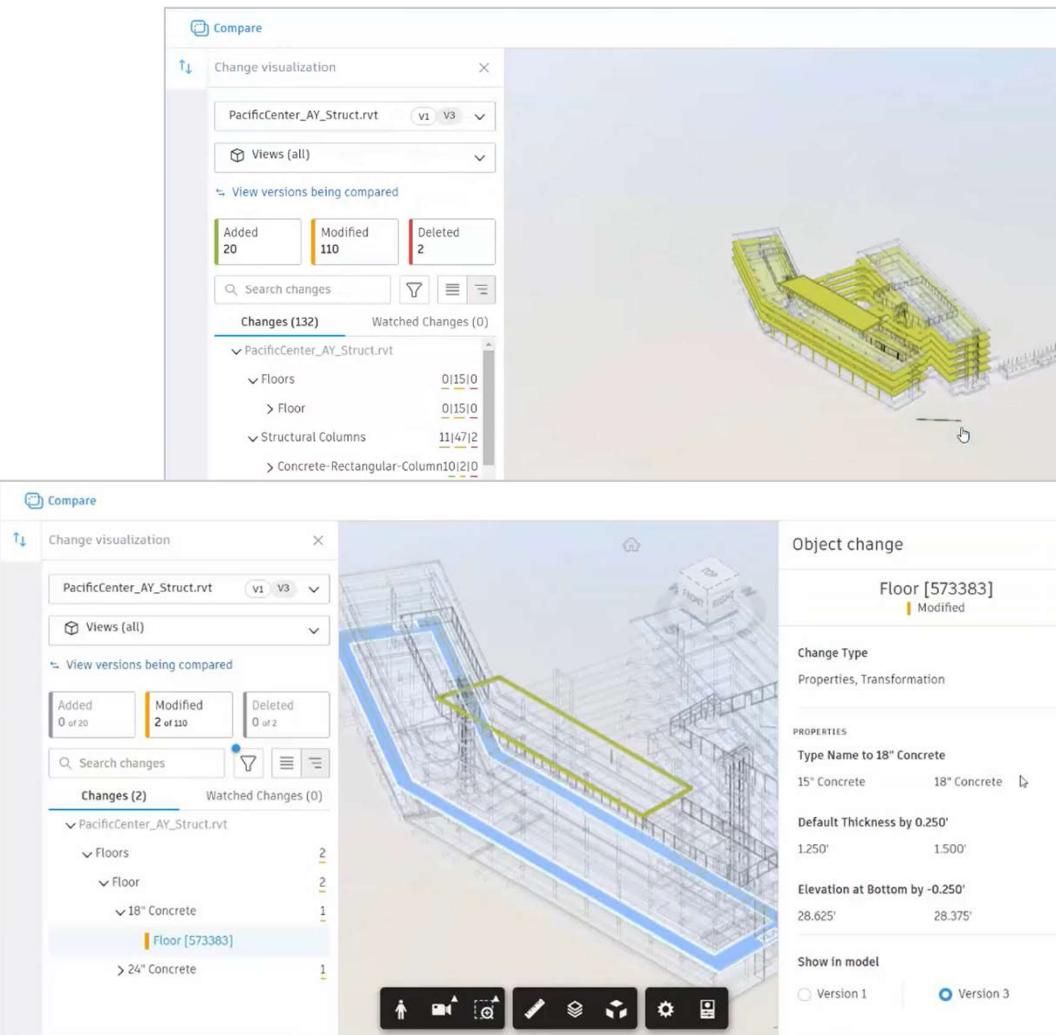
Classifying (BIM) SVF2 model properties

- Category
- Name
- Type (e.g., string, double)
- (Optional) Unit of Measure



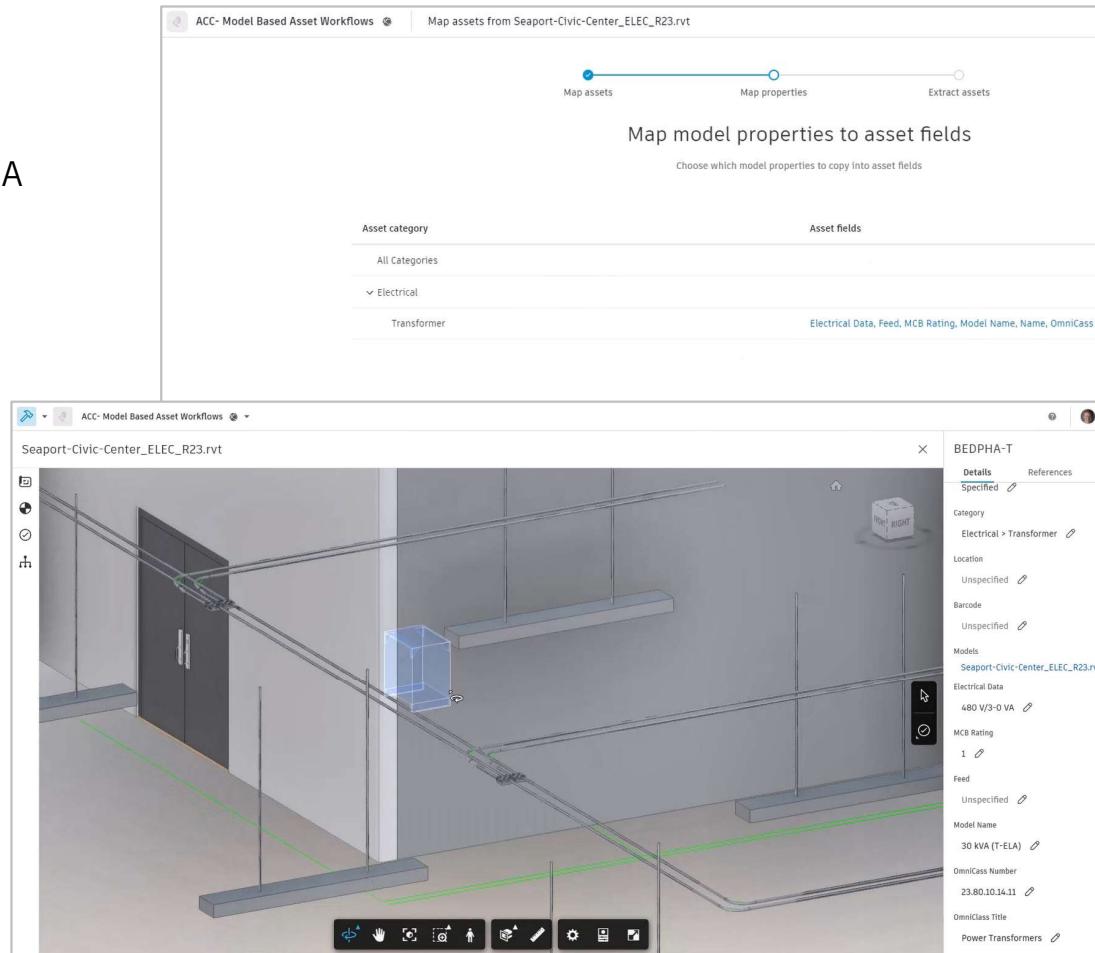
Model Properties API

- Released Feb 2022
- Autodesk/BIM 360 Docs based products
- Built on PropertyDb from Derivative Services (svf translation)
 - **Index** (Base) – query, filter properties of svf2 objects + extra (viewable bbox)
 - **Diff** – Index + compare two versions
- Used in product
 - Design Collaboration - Change analysis
 - Model Coordination
 - Model property breakdown / advanced filtering
 - Publishing views to Docs
 - Object exclusion
 - Assets – integrating with models



Model Properties API

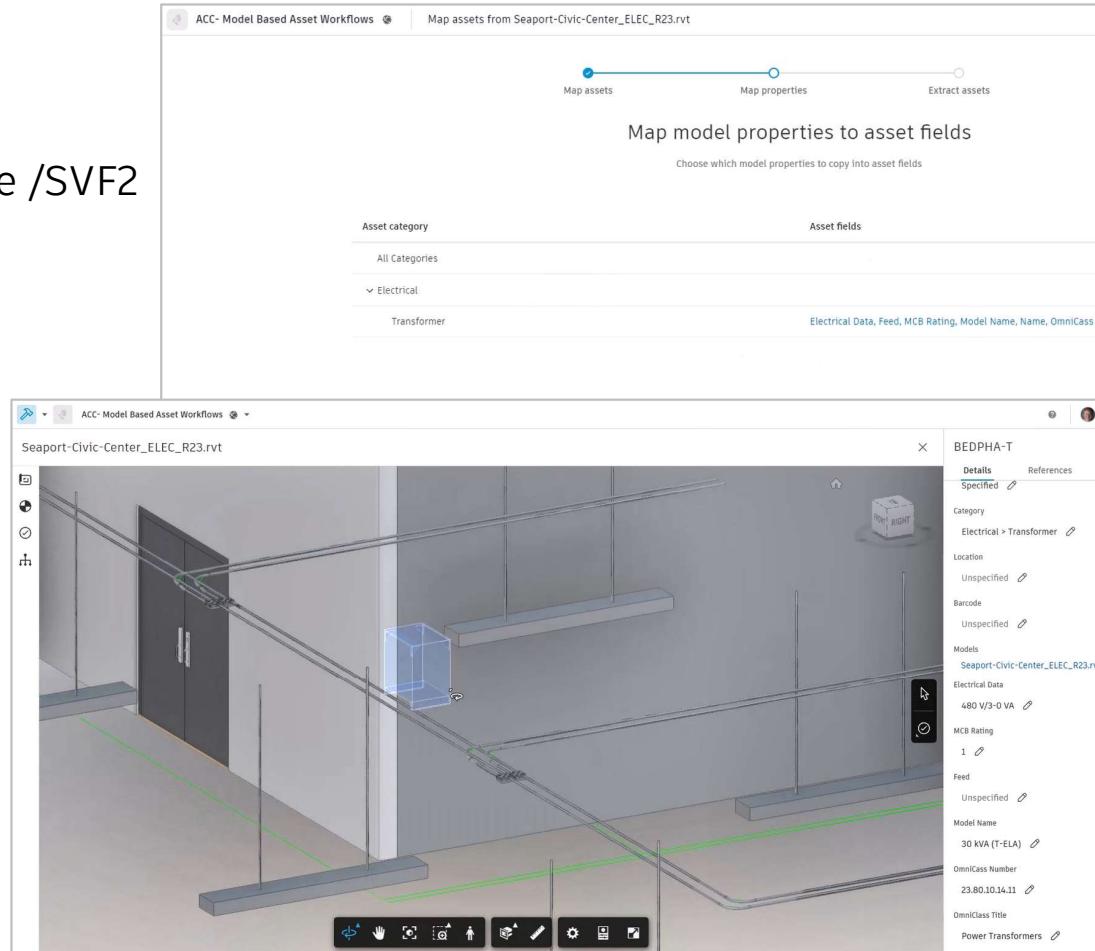
- Released Feb 2022
- Autodesk/BIM 360 Docs based products. US and EMEA
- Built on PropertyDb from Derivative Services (svf translation)
 - **Index** (Base) – query, filter properties of svf2 objects + extra (viewable bbox)
 - **Diff** – Index + compare two versions
- Used in product
 - Design Collaboration - Change analysis
 - Model Coordination
 - Model property breakdown / advanced filtering
 - Publishing views to Docs
 - Object exclusion
 - Assets – integrating with models

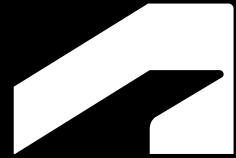


Model Properties API

Supported Files

- **Index** – any files supported Derivative Service /SVF2
- **Diff** - element id needs to be stable (uniquely identifiable)
 - **RVT**
 - **DWG**
 - **NWC** exported from:
 - Revit and
 - AutoCAD verticals
 - **IFC** exported from:
 - AutoCAD Architecture, MEP, Civil 3D 2018+
 - ARCHICAD
 - Revit
 - MigiCAD for Revit
 - Tekla Structures





Sample Applications

Model Properties API



Filter Elements & Partial Model Load

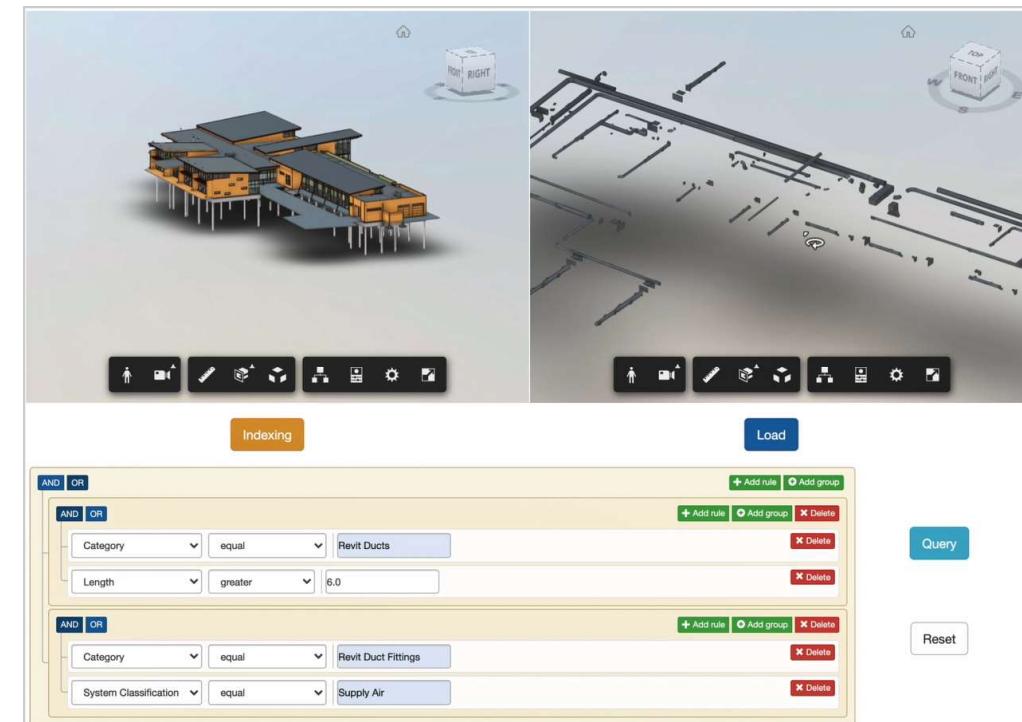
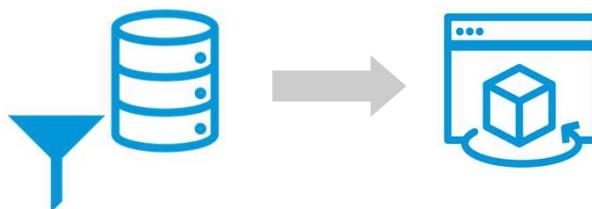
Index

What it does: Filters elements by properties, which can be geometric properties such as length and height. A filter condition is defined as a binary expression form and can be combined by AND/OR. The results are visualized in the Forge viewer.

Code:

<https://github.com/autodesk-platform-services/aps-model.properties-elements.filtering>

Author: Xiaodong Liang, Autodesk



Compare Two Versions

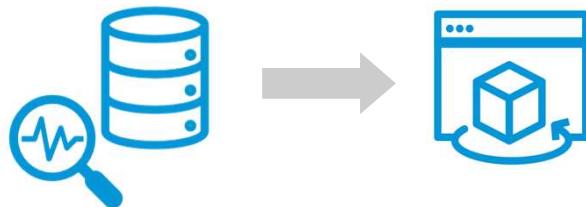
Diff

What it does: Compares two versions of a model and visualizes the differences found in the Forge Viewer. Differences can be in geometries and properties. Elements may be added, modified and removed.

Code:

<https://github.com/autodesk-platform-services/aps-model.properties-versions.difference>

Author: Xiaodong Liang, Autodesk



Added Items (25)					Removed Items (30)					Changed Items (284)		
name	category	level	RC	Free	name	category	level	RC	Free	name	geometry changed?	property changed?
Round Elbow [1187817]	Revit Duct Fittings	Arch-FIRST FLOOR	Duct Fittings	16"ø-	Round Elbow [853203]	Revit Duct Fittings	Arch-FIRST FLOOR	Duct Fittings	4"ø-	Round Elbow [839567]	yes	yes
Round Elbow [1187825]	Revit Duct Fittings	Arch-FIRST FLOOR	Duct Fittings	16"ø-	Round Elbow [914477]	Revit Duct Fittings	Arch-FIRST FLOOR	Duct Fittings	4"ø-	Round Elbow [839569]	yes	yes
Round Elbow [1187827]	Revit Duct Fittings	Arch-FIRST FLOOR	Duct Fittings	16"ø-	Round Elbow [914480]	Revit Duct Fittings	Arch-FIRST FLOOR	Duct Fittings	4"ø-	Round Elbow [839571]	yes	yes

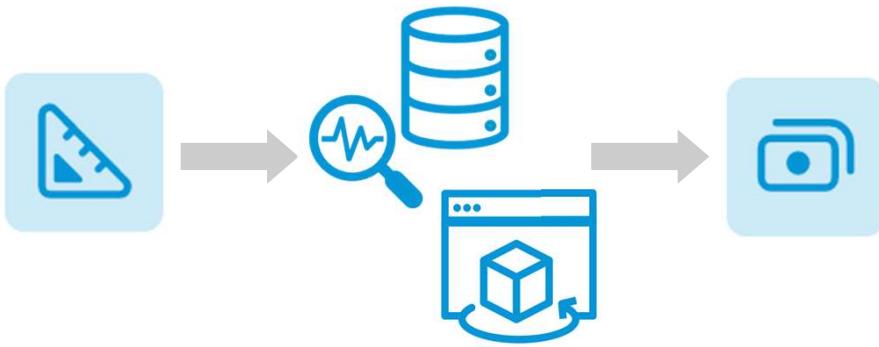
Change Analysis in Takeoff and Cost

Application of **Diff** in Estimate

What it does: Identifies changes in takeoff items, compares current and previous versions of a model and visualizes the differences in the Viewer. Update the budgets data in Cost module.

Code: <https://github.com/autodesk-platform-services/aps-acc-takeoff-versions-compare-cost>

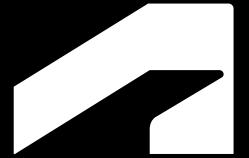
Author: Zhong Wu, Autodesk



The screenshot displays the Autodesk Construction Cloud interface with several windows open:

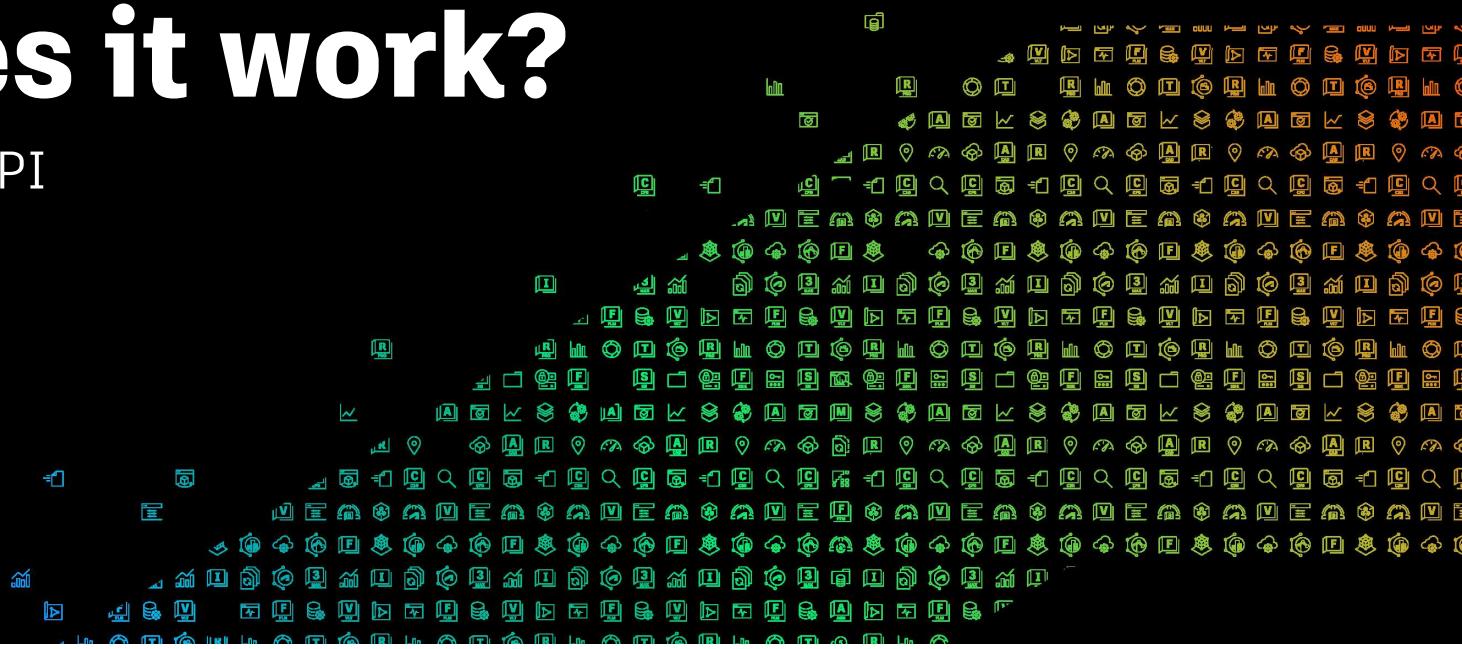
- Sheets & Models:** Shows a list of sheets and models, including "V1 Revit Basic House" and "V1 Revit Basic House".
- Viewer:** Shows a 3D rendering of a modern house with a red roof and a balcony.
- Cost Module:** Shows the "Income" tab with a table of budget items:

Budget Code	Budget Name	Qty	Unit	Unit Cost	Amount	Internal Budget Transfer	Main Contract	Linked to Main Contract SOV
01651 600	Glass	6	nr	230.00	1,380.00	0.00	No	No
09715 997	Window	8	nr	553.00	4,424.00			
- ACC Project List:** Shows a tree view of project components, including "Developer Advocacy Support" and "Project A1-V2".
- 3D View:** Shows a 3D view of a building's interior with certain elements highlighted in blue.
- Takeoff Package Budget:** Shows a table of unit prices for various items like Door, Floor, Glass, Wall, and Window.
- Cost Estimation of takeoff items for the latest version:** Shows a table comparing current, latest, quantity, cost, and sv2id for items like Basic Wall, Single Window, and Piping.



How does it work?

Model Properties API



How does it work?

Basic Workflow

Model Derivative



DM/Docs

Files uploaded to Docs via UI, Desktop Connector and API

Model Derivative

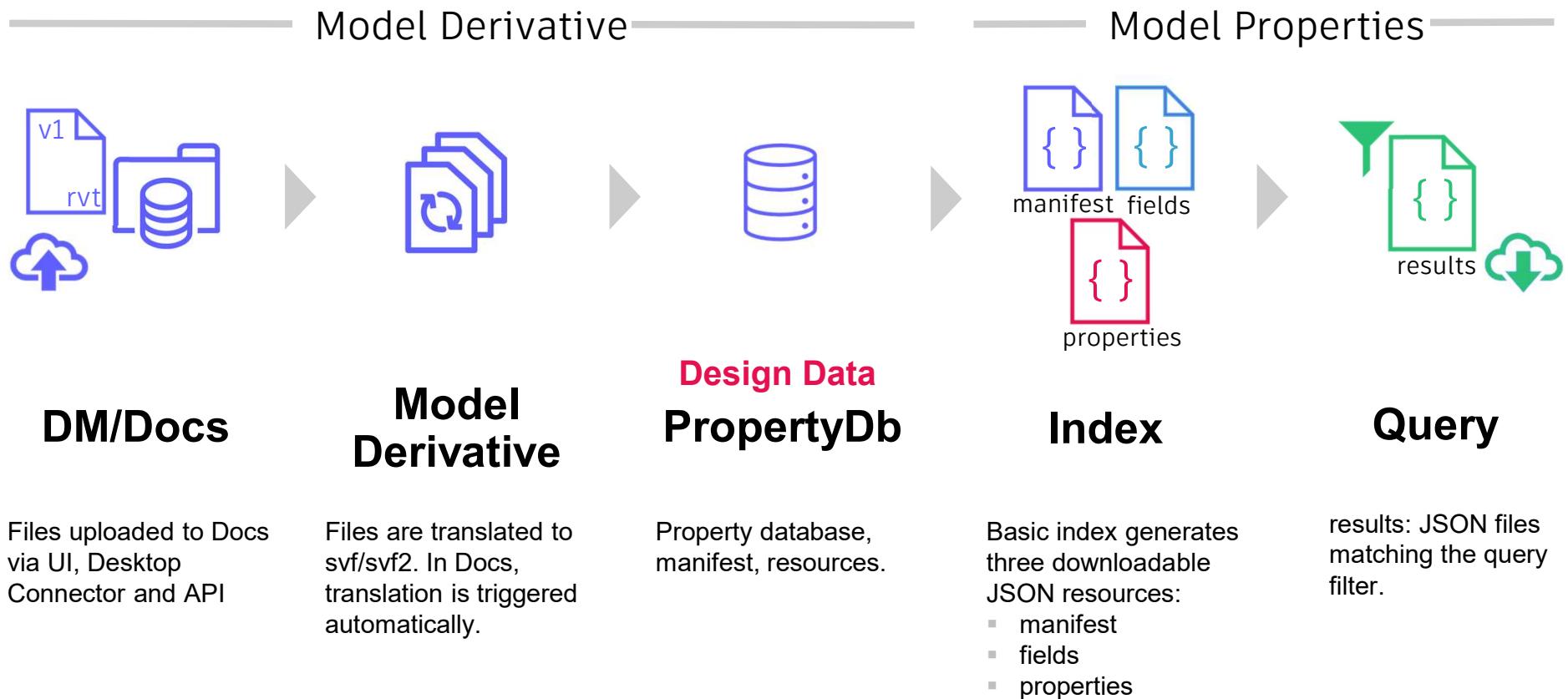
Files are translated to svf/svf2. In Docs, translation is triggered automatically.

Design Data PropertyDb

Property database, manifest, resources.

How does it work?

Basic Workflow



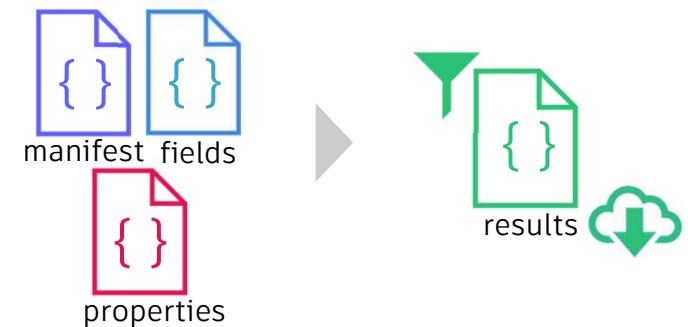
How does it work?

Resources Generated

Resource	Type	Description
manifest	JSON	Manifest for index or query detailing the seed files, svf2 propertyDb used to generate index rows
fields	NDJSON	The set of unique fields (property types) extracted for an index or query
properties	NDJSON	The raw objects property values for the index.
results	NDJSON	The object property results from a query executed.

NDJSON = new-line delimited JSON

Model Properties



Index

Basic index generates three downloadable JSON resources:

- manifest
- fields
- properties

Query

results: JSON files matching the query filter.

How does it work?

Index Endpoints

	Endpoints		
Index	POST	indexes:batch-status	
	GET	indexes/:indexId	
	GET	indexes/:indexId/manifest	
	GET	indexes/:indexId/fields	
	GET	indexes/:indexId/properties	
Query	POST	indexes/:indexId/queries	
	GET	indexes/:indexId/queries/:queryId	
	GET	indexes/:indexId/queries/:queryId/properties	

- 8 endpoints

How does it work?

Index Endpoints for Creation

	Endpoints	
Index	POST	indexes:batch-status
	GET	indexes/:indexId
	GET	indexes/:indexId/manifest 
	GET	indexes/:indexId/fields 
	GET	indexes/:indexId/properties 
Query	POST	indexes/:indexId/queries
	GET	indexes/:indexId/queries/:queryId 
	GET	indexes/:indexId/queries/:queryId/properties

- Create basic index – “lazy”
 - First time – start the indexing job and cache the results
 - Once executed – use the cache
 - Cached 30 days since the last used
- Poll for progress
 - state: PROCESSING, FINISHED, FAILED
- Response JSON is identical
- stats: objects (# of object)
- Create 3 downloadable json.gz resources
 - Manifest, fields, properties

Ex. Create basic index – POST indexes:batch-status

```
curl --request POST 'https://developer.api.autodesk.com/construction/index/v2/projects/f83c... /indexes:batch-status' \
--header 'Authorization: Bearer ****' \
--header 'Content-Type: application/json' \
--data-raw '{
    "versions": [
        {
            "versionUrn": "urn:adsk.wipprod:fs.file:vf.DyTwutcvTcOLUNUARxcTzQ?version=4"
        }
    ],
    [
        {
            "projectId": "f83cef12-deef-4771-9feb-4f85643e3c46",
            "indexId": "qTmPiKJZ7siqxkTNpWGAnw",
            "type": "INDEX",
            "state": "PROCESSING",
            "selfUrl": "https://developer.api.autodesk.com/construction/index/v2/projects/f83cef12-deef-4771-9feb-4f85643e3c46",
            "versionUrns": [
                "urn:adsk.wipprod:fs.file:vf.DyTwutcvTcOLUNUARxcTzQ?version=4"
            ],
            "updatedAt": "2021-08-19T08:21:13.8771187+00:00",
            "retrvAt": "2021-08-27T14:28:28.8382067+00:00",
            "stats": null,
            "manifestUrl": null,
            "fieldsUrl": null,
            "propertiesUrl": null
        }
    ]
}'
```

request

response

Ex. Polling for Progress - GET indexes/:indexId

```
curl --request GET 'https://developer.api.autodesk.com/construction/index/v2/projects/... /indexes/qTmPiKJZ7siqxkTNpWGAnw'  
--header 'Authorization: Bearer ****'
```

request

response

```
{  
    "projectId": "f83cef12-deef-4771-9feb-4f85643e3c46",  
    "indexId": "qTmPiKJZ7siqxkTNpWGAnw",  
    "type": "INDEX",  
    "state": "FINISHED",  
    "selfUrl": "https://developer.api.autodesk.com/construction/index/v2/proje... /indexes/qTmPiKJZ7siqxkTNpWGAnw",  
    "versionUrns": [  
        "urn:adsk.wipprod:fs.file:vf.DyTwutcvTcOLUNUARxcTzQ?version=4"  
    ],  
    "updatedAt": "2021-08-19T08:21:13.8771187+00:00",  
    "retryAt": "2021-08-27T14:31:55.1444684+00:00",  
    "stats": {  
        "objects": 33097  
    },  
    "manifestUrl": "https://developer.api.autodesk.com/construction/index/v2/pro... 46/indexes/qTmPiKJZ7siqxkTNpWGAnw/manifest",  
    "fieldsUrl": "https://developer.api.autodesk.com/construction/index/v2/proje... /indexes/qTmPiKJZ7siqxkTNpWGAnw/fields",  
    "propertiesUrl": "https://developer.api.autodesk.com/construction/index/v2/p... 3c46/indexes/qTmPiKJZ7siqxkTNpWGAnw/properties"  
}
```

How does it work?

Index Endpoints for Download

Endpoints	
Index	POST indexes:batch-status
	GET indexes/:indexId
	GET indexes/:indexId/manifest 
	GET indexes/:indexId/fields 
	GET indexes/:indexId/properties 
Query	POST indexes/:indexId/queries
	GET indexes/:indexId/queries/:queryId 
	GET indexes/:indexId/queries/:queryId/properties

- (Optional) download
 - manifest
 - fields
 - properties

```
{
  "schema": "2.0.0",
  "projectId": "f83cef12-deef-4771-9feb-4f85643e3c46",
  "status": "Succeeded",
  "createdAt": "2021-07-23T08:56:07.0868303+00:00",
  "seedFiles": [
    {
      "lineageId": "a19f7db",
      "lineageUrn": "urn:adsk.wipprod:dm.lineage:DyTwutcvTcOLUNUARxcTzQ",
      "versionUrn": "urn:adsk.wipprod:fs.file:vf.DyTwutcvTcOLUNUARxcTzQ?version=4",
      "databases": [
        {
          "id": "3747dccf",
          "offsets": "urn:adsk.viewing:fs.file:dXJuOmFkc2sud2l...yc21vbj04/output/Resource/objects_offs.json.gz",
          "attributes": "urn:adsk.viewing:fs.file:dXJuOmFkc2sud2l...dmVyc21vbj04/output/Resource/objects_attrs.json.gz",
          "values": "urn:adsk.viewing:fs.file:dXJuOmFkc2sud2l...c21vbj04/output/Resource/objects_vals.json.gz",
          "mapping": "urn:adsk.viewing:fs.file:dXJuOmFkc2sud2l...yc21vbj04/output/Resource/objects_avs.json.gz",
          "ids": "urn:adsk.viewing:fs.file:dXJuOmFkc2sud2lwchJ...vbj04/output/Resource/objects_ids.json.gz"
        }
      ],
      "views": [
        {
          "id": "e7fda9d5",
          "urn": "urn:adsk.wipprod:fs.file:vf.DyTwutcvTcOLUNUARxcTzQ?version=4",
          "is3d": true,
          "viewableName": "{3D}",
          "viewableId": "0935d8b2-149b-4a0d-b816-863f0d595a20-000bcd64",
          "viewableGuid": "00cd2da3-fbfa-44a9-7a33-cad0bc4720cb"
        },
        {
          "id": "12fcb372",
          "urn": "urn:adsk.wipprod:fs.file:vf.DyTwutcvTcOLUNUARxcTzQ?version=4",
          "is3d": true,
          "viewableName": "New Construction",
          "viewableId": "c884ae1b-61e7-4f9d-0001-719e20b22d0b-00120bb2",
          "viewableGuid": "4a966c2a-ead6-65c3-4f98-273dd7543047"
        }
      ]
    },
    "errors": [],
    "stats": {
      "objects": 33097,
      "contentlength": 1881318
    }
  }
}
```

Lineage & Version

Manifest (.json)

SVF2 Prop DB Resource URNs

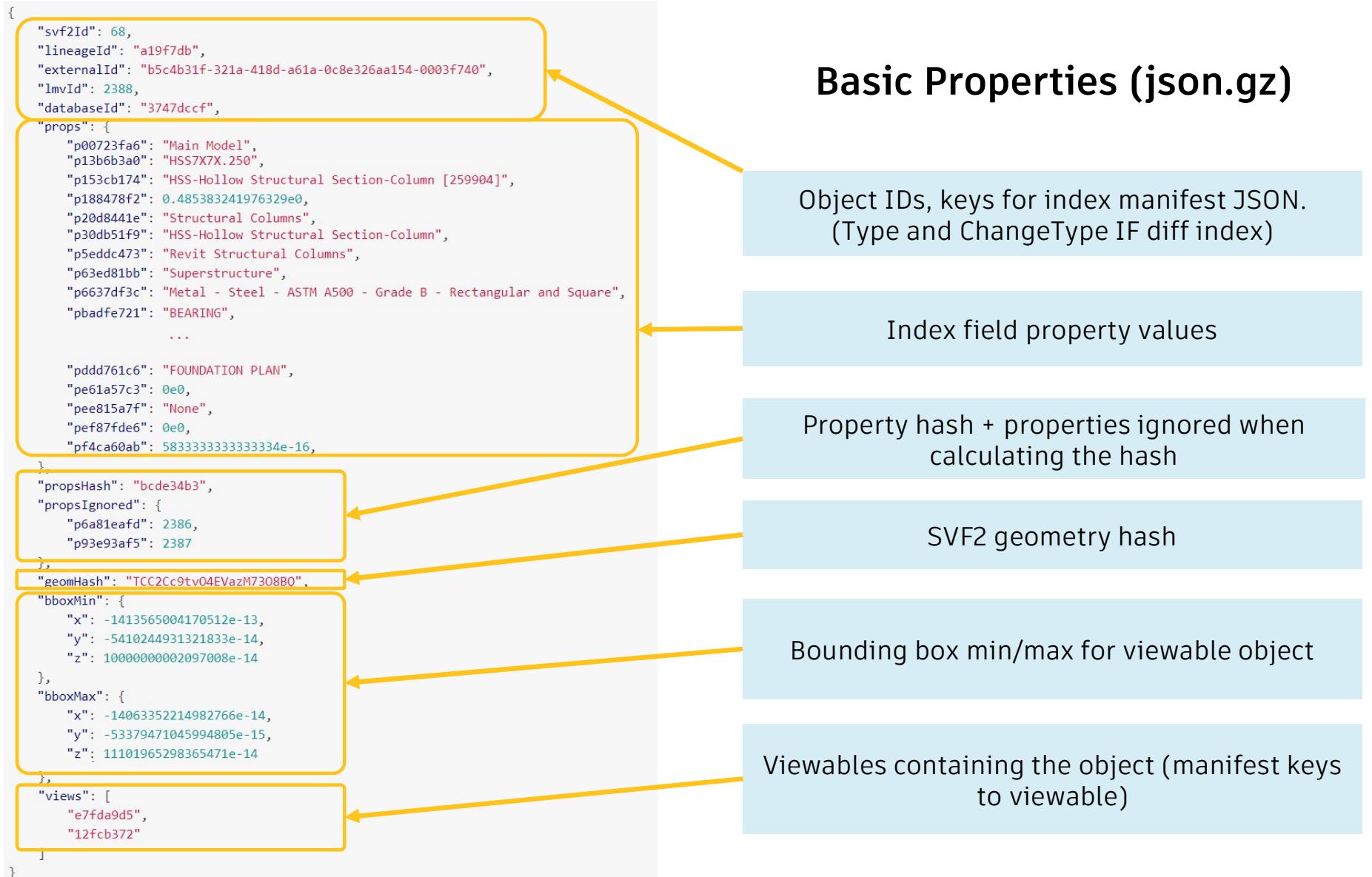
Viewables

Index object count and byte size

Index Fields (json.gz)

```
{
  "key": "p153cb174", "category": "__name__", "type": "String", "name": "name", "uom": null}
  {"key": "p74a9a490", "category": "__document__", "type": "String", "name": "schema_name", "uom": null}
  {"key": "p137c14f2", "category": "__document__", "type": "String", "name": "schema_version", "uom": null}
  {"key": "p1490bcea", "category": "__document__", "type": "Boolean", "name": "is_doc_property", "uom": null}
  {"key": "p5eddc473", "category": "__category__", "type": "String", "name": "Category", "uom": null}
  {"key": "p00723fa6", "category": "Identity Data", "type": "String", "name": "Design Option", "uom": null}
  {"key": "pe8094f29", "category": "Other", "type": "String", "name": "Project Issue Date", "uom": null}
  {"key": "p50756a0d", "category": "Other", "type": "String", "name": "Client Name", "uom": null}
  {"key": "p32791eb0", "category": "Other", "type": "String", "name": "Project Address", "uom": null}
  {"key": "pb75ced9", "category": "Other", "type": "String", "name": "Project Name", "uom": null}
  {"key": "p8213f1ad", "category": "Other", "type": "String", "name": "Project Number", "uom": null}
  {"key": "pa7275c45", "category": "__categoryId__", "type": "Integer", "name": "CategoryId", "uom": null}
  {"key": "p93e93af5", "category": "parent", "type": "DbKey", "name": "parent", "uom": null}
  {"key": "p1d45bc4f", "category": "Dimensions", "type": "Double", "name": "Computation Height", "uom": "ft"}
  {"key": "pe01bd7ef", "category": "Extents", "type": "String", "name": "Scope Box", "uom": null}
  {"key": "p9ffb245", "category": "Materials and Finishes", "type": "Integer", "name": "Color", "uom": null}
  {"key": "p1b3b6224", "category": "Materials and Finishes", "type": "String", "name": "Transparency", "uom": null}
  {"key": "pd9fcab30", "category": "Materials and Finishes", "type": "Boolean", "name": "Glow", "uom": null}
  {"key": "pf62e5a3c", "category": "Structural", "type": "Double", "name": "Structural Framing Length Roundoff", "uom": "ft"}}
```

Field Key → SQL column name	Category	Type	Name	UOM (Unit of Measurement)
--------------------------------	----------	------	------	------------------------------



How does it work?

Index Endpoints for Query

	Endpoints		
Index	POST	indexes:batch-status	
	GET	indexes/:indexId	
	GET	indexes/:indexId/manifest	
	GET	indexes/:indexId/fields	
	GET	indexes/:indexId/properties	
Query	POST	indexes/:indexId/queries	
	GET	indexes/:indexId/queries/:queryId	
	GET	indexes/:indexId/queries/:queryId/properties	

- Build and run query.
 - Index queries are described using custom JSON schema, (which is converted to a filter expression. AWS S3 Select)
 - Columns can be restricted. Can use alias (have different header)
- Poll for progress.
 - state: PROCESSING, FINISHED, FAILED

```
// Forge viewer element display name field
{"key": "p153cb174", "category": "__name__", "type": "String", "name": "name", "uom": null}
// Revit category name field
{"key": "p20d8441e", "category": "__category__", "type": "String", "name": "__RC", "uom": null}
// Revit family name field
{"key": "p30db51f9", "category": "__category__", "type": "String", "name": "__RFN", "uom": null}
// Revit type name field
{"key": "p13b6b3a0", "category": "__category__", "type": "String", "name": "__RFT", "uom": null}
```

Index Fields

```
{
  "Query": {
    "query": {
      "$and": [
        { "$notnull": "s.props.p20d8441e" },
        { "$notnull": "s.props.p30db51f9" },
        { "$notnull": "s.props.p13b6b3a0" },
        { "$gt": [{ "$count": "s.views" }, 0] }
      ]
    },
    "columns": {
      "svf2Id": true,
      "lmvName": "s.props.p153cb174",
      "revitCategory": "s.props.p20d8441e",
      "revitFamily": "s.props.p30db51f9",
      "revitType": "s.props.p13b6b3a0",
      "s.views": true
    }
  }
}
```

Row has Revit classification

Views array has count more than 0

Columns define alias

Sample Query: Get Revit Classification with Column Transform

Equivalent in S3 SQL

```
select
  svf2Id,
  props.p153cb174 as lmvName,
  props.p20d8441e as revitCategory,
  props.p30db51f9 as revitFamily,
  props.p13b6b3a0 as revitType,
  views
from S3Object[*] s
where
  props.p20d8441e is not null and
  props.p30db51f9 is not null and
  props.p13b6b3a0 is not null and
  count(views) > 0
```

How does it work?

Index Endpoint to Download Query Results

		Endpoints
Index	POST	<code>indexes:batch-status</code>
	GET	<code>indexes/:indexId</code>
	GET	<code>indexes/:indexId/manifest</code> 
	GET	<code>indexes/:indexId/fields</code> 
	GET	<code>indexes/:indexId/properties</code> 
Query	POST	<code>indexes/:indexId/queries</code>
	GET	<code>indexes/:indexId/queries/:queryId</code> 
	GET	<code>indexes/:indexId/queries/:queryId/properties</code>

- Download the query results
 - use the queryResultsUrl in query call or query id to download the index rows which match the submitted query expression.
 - Result: line delimited JSON
 - a sub-set of the property index rows
 - Format is the exactly the same as properties we saw earlier.

How does it work?

Diff Endpoints

Endpoints	
Diff	POST diffs:batch-status
	GET diffs/:diffId
	GET diffs/:diffId/manifest 
	GET diffs/:diffId/fields 
	GET diffs/:diffId/properties 
Query	POST diffs/:diffId/queries
	GET diffs/:diffId/queries/:queryId
	GET diffs/:diffId/queries/:queryId/properties 

- Diff - The steps are the same as Index
- Specify two version urn's to compare:

```
{  
  "diffs": [  
    {  
      "prevVersionUrn": "urn:adsk.wi...  
      "curVersionUrn": "urn:adsk.wi...  
    }  
  ]  
}
```

- stats: add, removed, modified

```
{
  "type": "OBJECT_CHANGED",
  "svf2Id": 160,
  "externalId": "552d2a83-4642-4d5c-8e7f-5de799129097-000d047a",
  "lmvId": 2699,
  "lineageId": "2b856593",
  "databaseId": "3d0bd846",
  "props": {
    "p002932a2": 0.0,
    "p01bbdcf2": "Arch-FIRST FLOOR",
    ...
  },
  "views": [
    "f109b687",
    "f24d458"
  ],
  "prev": {
    "lmvId": 2699,
    "lineageId": "b28c3429",
    "databaseId": "936acb06",
    "props": {
      "p1b2aabef": 10.5
    },
    "propsHash": "ad9828df",
    "propsIgnored": {
      "p6a81eaf9": 2545,
      "p93e93af5": 2546
    },
    "geomHash": "4s1yfJZd0hnBu2DdFL4HEw",
    "bboxMin": {
      "x": -1413565004170512e-13,
      "y": -5410244931321833e-14,
      "z": 10000000002097008e-14
    },
    "bboxMax": {
      "x": -14063352214982766e-14,
      "y": -53379471045994805e-15,
      "z": 11101965298365471e-14
    },
    "views": [
      "f109b687",
      "8e525582"
    ]
  }
}
```

Type if diff index

Previous (prev) object embedded in current row. Lineage manifest key & viewer id.

Array of property keys which have values different to current

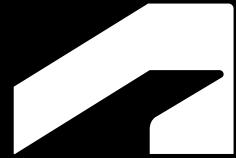
Previous bounding boxes, hashes and viewable keys in manifest

Basic Index Row vs. Diff Index Row

Description	Current version	Previous version
IDs	s. <code>svf2Id</code> s. <code>externalId</code>	
Change type, previous vs. current		s. <code>type</code> s. <code>changeType</code>
lineage version info, SVF2 database URNs	s.lmvId s.lineageId s.databaseId	s. <code>prev.lmvId</code> s. <code>prev.lineageId</code> s. <code>prev.databaseId</code>
Property values	s.props.* s.propsHash s.propsIgnored.*	s.prev.props.* s.prev.propsHash s.prev.propsIgnored.*
Geometry hash and bounding box values IF viewable	s.geomHash s.bboxMin.x s.bboxMin.y s.bboxMin.z s.bboxMax.x s.bboxMax.y s.bboxMax.z	s.prev.geomHash s.prev.bboxMin.x s.prev.bboxMin.y s.prev.bboxMin.z s.prev.bboxMax.x s.prev.bboxMax.y s.prev.bboxMax.z
Viewable keys IF viewable	s.views s.views[i]	s.prev.views s.prev.views[i]

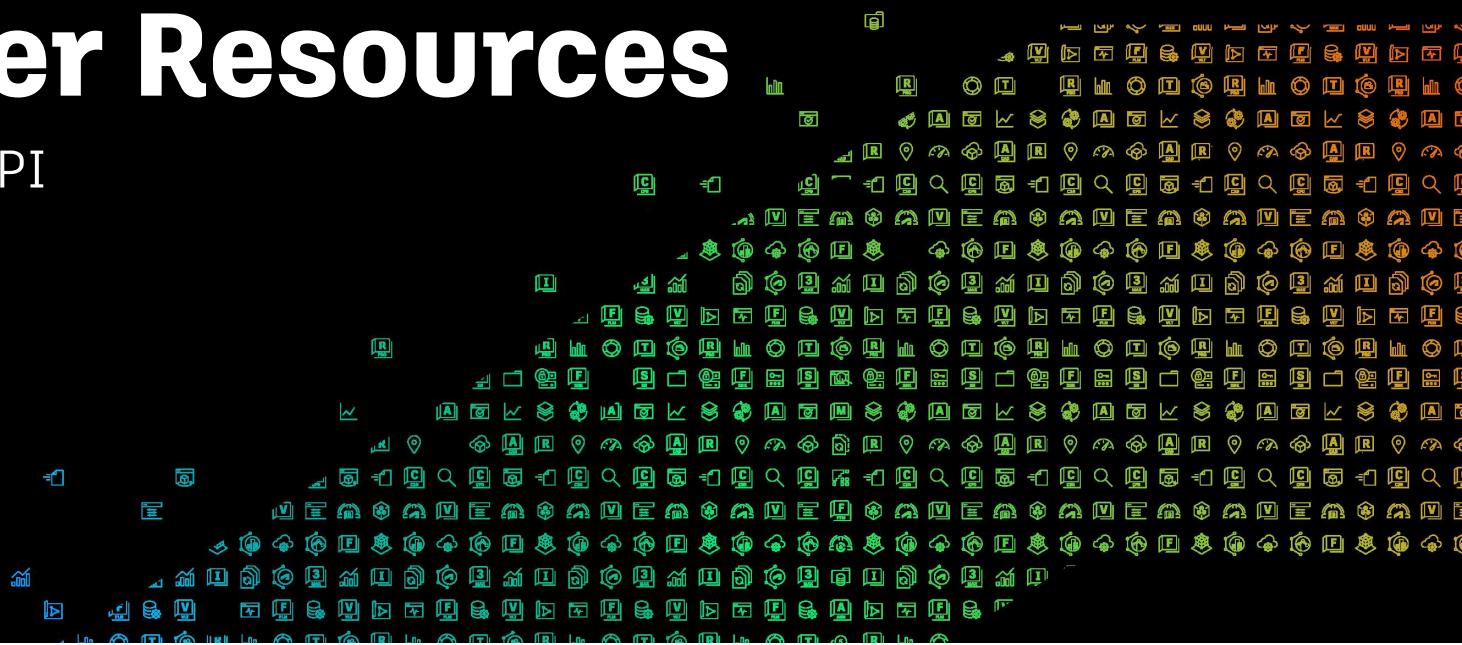
JSON Abstract Syntax Tree → S3 Select(AWS)

\$not	\$like	\$cat	\$char_length
\$and	\$between	\$coalesce	\$lower
\$or	\$in	\$mod	\$upper
\$gt	\$contains	\$cast	\$count
\$lt	\$isnull	\$nullif	\$sum
\$eq	\$notnull	\$date_add	\$avg
\$le	\$add	\$date_diff	\$min
\$ge	\$sub	\$extract	\$max
	\$mul	\$substring	\$trim
	\$div	\$to_string	\$utcnow
		\$to_timestamp	\$case



Developer Resources

Model Properties API



Developer Resources

Documentation

- Field Guide
 - [Introduction to Model Properties](#)
- Step-by-Step Tutorials
 - [Index Querying](#)
 - [Tracking Changes](#)
 - [Query Language Reference](#)
- Reference Guide
 - [Index](#)
 - [Diff](#)

The screenshot shows the Autodesk Platform Services Documentation website. The top navigation bar includes links for Solutions, Getting Started, Documentation (which is highlighted), Success Stories, Community, Support, and Pricing. Below the navigation is a search bar. The main content area has a sidebar on the left titled 'Developer's Guide' with sections for Introduction, Field Guide (Admin, Assets (beta), AutoSpecs, Cost Management, Files, Forms, Issues, Locations), Model Coordination, Relationships, Model Properties (which is highlighted with a yellow box), Sheets, Submittals, and Regions. The main content area displays the 'Introduction to Model Properties' page under 'Model Properties Service'. The page content discusses the model properties service allowing callers to build and query indexes for Autodesk and BIM 360 Docs properties using PDB files extracted from SVF files. It also mentions support for backwards compatibility and the inclusion of both SVF object IDs and rows generated by the service. A separate section on 'Diff index file type support' is mentioned.

AUTODESK Platform Services

Solutions Getting Started Documentation Success Stories Community Support Pricing

Developer's Guide

Introduction

Field Guide

- Admin
- Assets (beta)
- AutoSpecs
- Cost Management
- Files
- Forms
- Issues
- Locations

Model Coordination

Relationships

Model Properties

Sheets

Submittals

Regions

Documentation / Autodesk Construction Cloud APIs / Developer's Guide

Introduction to Model Properties

Model Properties Service

The model properties service allows callers to build and query indexes b...
to Autodesk and BIM 360 Docs. These are the properties that can be disp...
using the PDB (Property Database) files extracted as part of the SVF tra...
added to these indexes from the SVF2 files generated from SVF. For a pro...
be supported in the SVF2 translation process. For backwards compatibili...
rows generated by the service contain both the SVF object IDs (lmvId) a...

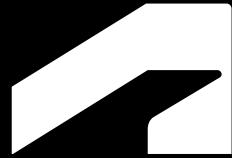
In addition to property data, the indexes built by the service contain the...
objects that are viewable via the Viewer, allowing callers to view the res...
to trigger the service to calculate changes (diffs) that have occurred bet...
calculate a diff, callers specify a previous and current file version, and th...
changes to their property values and bounding-box geometry.

Diff index file type support

Unlike basic property indexing, which only depends on SVF translation, c...

Developer Resources

- Code Samples on GitHub
 - [Postman Collection](#) (correspond to three Step-by-Step tutorials)
 - [Model Properties API Walkthrough in PowerShell Core](#) (scripting to explore query language)
 - [Element Filtering and Partial Model Load](#) (Integration with Viewer)
 - [Compare Two Versions](#) (Integration with Viewer)
- Blog Post
 - “BIM 360/ACC Model Properties API”
<https://aps.autodesk.com/blog/bim-360acc-model-properties-api>
includes links to the resources
 - Search more with “Model Properties”, e.g.,
<https://aps.autodesk.com/blog/model-properties-api-vs-model-derivative-api>



Common Questions

Q: What is the difference between a standard and a premium plan?
A: Premium plans offer additional features such as priority support, advanced reporting, and access to exclusive resources.

Q: Can I cancel my subscription at any time?
A: Yes, you can cancel your subscription at any time through your account settings.

Q: How do I renew my subscription?
A: You can renew your subscription by logging into your account and selecting the renewal option.

Q: What is the refund policy for my purchase?
A: Our refund policy varies depending on the plan and the reason for cancellation. Please refer to our terms and conditions for more information.

Q: How do I change my payment method?
A: You can change your payment method by logging into your account and updating your payment information.

Q: What is the difference between a standard and a premium plan?
A: Premium plans offer additional features such as priority support, advanced reporting, and access to exclusive resources.

Q: Can I cancel my subscription at any time?
A: Yes, you can cancel your subscription at any time through your account settings.

Q: How do I renew my subscription?
A: You can renew your subscription by logging into your account and selecting the renewal option.

Q: What is the refund policy for my purchase?
A: Our refund policy varies depending on the plan and the reason for cancellation. Please refer to our terms and conditions for more information.

Q: How do I change my payment method?
A: You can change your payment method by logging into your account and updating your payment information.

Methods to Access Design Data

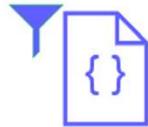
Model
Derivative

Model
Properties

Design
Automation

Data Exchange

AEC Data Model



Platform	ACC/BIM 360	Platform	Autodesk Docs	Autodesk Docs* ²
File based	File based	File based/RCM	Cloud hosted	Cloud hosted
Entire model	Entire model	Entire model Revit	Partial model Revit	Entire model Revit
Light weight query	Full query/filter	Control same as desktop add-in	Flexible query (GraphQL)	Flexible query (GraphQL)
Read	Read	Read/write	Read & evolving limited write	Read & longer term evolving limited write
Today	Today	Today	Public beta/Future	Public beta/Future

ACC API Roadmap



Added '24

- Submittals read GA
- Docs file description read
- Automatic region routing for all ACC specific API
- Data Connector schema
- Parameters sharing collections
- Cost batch budgets-contracts link
- BuildingConnected write
- TradeTapp read



Currently Active

- Submittals write
- Optional region header change from APAC to AUS
- BuildingConnected (BC) Webhooks for opportunities and bids
- BC opportunity comments
- BC adding bidders to bid packages



Near Future

- Review public (currently private beta)
- Markup a.k.a. Issues pushpin
- Model Aggregate
- Project level custom attributes
- Data Connector-multi projects
- Cost - cost item sub items, tax, workflow



Beyond

- Secure Service Account
- Alignment
- Design Collaboration packages/settings
- Cost – budget transfers, documents, payment, activities

Summary

- 1 Introduction: Shared Services in ACC
- 2 Relationships API
- 3 Model Properties API
- 4 Common Questions
- 5 What's Next





Make Anything

<https://aps.autodesk.com/get-help>
aps.help@autodesk.com

Autodesk and the Autodesk logo are registered trademarks or trademarks of Autodesk, Inc., and/or its subsidiaries and/or affiliates in the USA and/or other countries. All other brand names, product names, or trademarks belong to their respective holders. Autodesk reserves the right to alter product offerings and specifications, and pricing at any time without notice, and is not responsible for typographical or graphical errors that may appear in this document. © 2024 Autodesk. All rights reserved.