LINKED BUILDING DATA MEETS IFC.JS

ABOUT US

Antonio González Viegas

Mads Holten Rasmussen

IFC.JS

An open source library for IFC on the web

WHY?

BIM is hard, thus only the big players can compete in equal terms.

WHY?

WHY?

👨‍💼👨‍💼👨‍💼 Consequences of an oligopoly: expensive software that does not meet many of the needs of users and businesses.

CONCEPT

Anyone can create a videogame for free in some hours.

🌱 What if we could make a BIM tool in an afternoon and completely free?

How it works

```
1  import { IfcViewerAPI } from 'web-ifc-viewer';
2
3  // Initialize IFC.js
4  const container = document.getElementById('viewer-container
5  const viewer = new IfcViewerAPI({ container });
6
7  // Add scene elements
8  viewer.addAxes();
9  viewer.addGrid();
10
11 // Set up input element
12 const input = document.getElementById("file-input");
13
14 input.addEventListener("change",
15   async (changed) => {
```

How it works

```
1  import { IfcViewerAPI } from 'web-ifc-viewer';
2
3  // Initialize IFC.js
4  const container = document.getElementById('viewer-container
5  const viewer = new IfcViewerAPI({ container });
6
7  // Add scene elements
8  viewer.addAxes();
9  viewer.addGrid();
10
11 // Set up input element
12 const input = document.getElementById("file-input");
13
14 input.addEventListener("change",
15   async (changed) => {
```

How it works

```
1  import { IfcViewerAPI } from 'web-ifc-viewer';
2
3  // Initialize IFC.js
4  const container = document.getElementById('viewer-container
5  const viewer = new IfcViewerAPI({ container });
6
7  // Add scene elements
8  viewer.addAxes();
9  viewer.addGrid();
10
11 // Set up input element
12 const input = document.getElementById("file-input");
13
14 input.addEventListener("change",
15   async (changed) => {
```

How it works

```
1  import { IfcViewerAPI } from 'web-ifc-viewer';
2
3  // Initialize IFC.js
4  const container = document.getElementById('viewer-container
5  const viewer = new IfcViewerAPI({ container });
6
7  // Add scene elements
8  viewer.addAxes();
9  viewer.addGrid();
10
11 // Set up input element
12 const input = document.getElementById("file-input");
13
14 input.addEventListener("change",
15   async (changed) => {
```

How it works

```
viewer.addAxes();
viewer.addGrid();

// Set up input element
const input = document.getElementById("file-input");

input.addEventListener("change",
  async (changed) => {
  const file = changed.target.files[0];
  const ifcURL = URL.createObjectURL(file);
  viewer.IFC.loadIfcUrl(ifcURL);

  },
  false
);
```

Example 1: How it works

Example 2: Making a BIM tool inside Twitter

ansoni
@agviegasBIM

Everyone is prejudiced, including me. For example, if you talk to me about how your BIM processes implement Big data or AI, I'm not likely to take you very seriously anymore.

1:43 AM · Sep 25, 2021

11 Reply Share this Tweet

Read 2 replies

Example 3: BIM + GIS in 48 hours

EXAMPLE 4: DESKTOP APPLICATION

SUSTAINABILITY

1 organization is supporting IFC.js
Contribute on Open Collective

Studio 3DX, Inc.

sponsor
since January 2022

$100
Amount contributed

17 individuals are supporting IFC.js
Contribute on Open Collective

IFC-LBD

An open source converter built in IFC.js

- Includes a CLI tool
- Works with web-ifc-three

```
const rdf = await lbdParser.parseBOTTriples(ifcApi, mod
```

- Currently supports BOT, Products and FSO
- Modular and easy to extend
- Written in TypeScript

USE AS A CLI TOOL

```
npm install -g ifc-lbd
```

USE IT IN AN APP

```
1
2  import * as WebIFC from "web-ifc/web-ifc-api.js";
3  import { LBDParser } from "ifc-lbd";
4
5  import * as path from 'path';
6  import { readFile } from "fs";
7  import * as util from "util";
8  const readFileAsync = util.promisify(readFile);
9
10 async function main(){
11
12   // Read file
13   const modelPath = path.join(__dirname, './Duplex.ifc');
14   const modelData = readFileAsync(modelPath);
15
```

USE IT IN AN APP

```
import * as util from "util";
const readFileAsync = util.promisify(readFile);

async function main(){

  // Read file
  const modelPath = path.join(__dirname, './Duplex.ifc');
  const modelData = readFileAsync(modelPath);

  // Init API and load model
  const ifcApi = new WebIFC.IfcAPI();
  await ifcApi.Init();

  const modelID = ifcApi.OpenModel(ifcModelData);

  // Init IFB Parser and parse BOT
```

USE IT IN AN APP

```
13  const modelPath = path.join(__dirname, './Duplex.ifc');
14  const modelData = readFileAsync(modelPath);
15
16  // Init API and load model
17  const ifcApi = new WebIFC.IfcAPI();
18  await ifcApi.Init();

19  const modelID = ifcApi.OpenModel(ifcModelData);
20
21  // Init LBD Parser and parse BOT
22  const lbdParser = new LBDParser();
23  const botTriples = await lbdParser.parseBOTTriples(ifcApi
24  console.log(botTriples);
25
26  // Close the model, all memory is freed
```

USE IT IN AN APP

```
18    await ifcApi.Init();

19    const modelID = ifcApi.OpenModel(ifcModelData);
20
21    // Init LBD Parser and parse BOT
22    const lbdParser = new LBDParser();
23    const botTriples = await lbdParser.parseBOTTriples(ifcApi
24    console.log(botTriples);
25
26    // Close the model, all memory is freed
27    ifcApi.CloseModel(modelID);
28
29  }
30
31  main();
```

DEEPER LOOK INTO THE CODE

PATH SEARCH HELPERS

- buildRelOneToOne()
- buildRelOneToMany()

BUILDRELONETOONE()

For example: IFCRELSPACEBOUNDARY

```
function buildRelOneToOne(
    ifcAPI: WebIFC.IfcAPI, modelID: number = 0,
    relationshipType: number, subjectRef: string,
    targetRef: string, rdfRelationship: string,
    includeInterface: boolean = false,
    biderectional: boolean = false)
```

BUILDRELONETOONE()

SPACE/ELEMENT ADJACENCY

```
1  private async buildSpaceAdjacentElementRelationships(){
2    const subjectRef = "RelatingSpace";
3    const targetRef = "RelatedBuildingElement";
4    const rdfRelationship = "bot:adjacentElement";
5    return await buildRelOneToOne(
6      this.ifcAPI, // API
7      this.modelID, // modelID
8      IFCRELSPACEBOUNDARY, // relationship type
9      subjectRef, // what rel is used to find subject of trip
10     targetRef, // what rel is used to find object of triple
11     rdfRelationship, // what's the predicate of the new tri
12     true // Add triple in both directions?
13   );
14 }
```

BUILDRELONETOMANY()

For example: IFCRELCONTAINEDINSPATIALSTRUCTURE

```
function buildRelOneToMany(
    ifcAPI: WebIFC.IfcAPI, modelID: number = 0,
    relationshipType: number, subjectRef: string,
    targetRef: string, rdfRelationship: string,
    subjectClassConstraint?: number,
    targetClassConstraint?: number)
```

BUILDRELONETOONE()

SPACE/ELEMENT ADJACENCY

```
1  private async buildSpaceContainedElementRelationships(){
2    const subjectRef = "RelatingStructure";
3    const targetRef = "RelatedElements";
4    const rdfRelationship = "bot:containsElement";
5    const subjectClass = IFCSPACE;
6    return await buildRelOneToMany(
7      this.ifcAPI, // API
8      this.modelID, // modelID
9      IFCRELCONTAINEDINSPATIALSTRUCTURE, // relationship type
10     subjectRef, // what rel is used to find subject of trip
11     targetRef, // what rel is used to find objects of tripl
12     rdfRelationship, // what's the predicate of the new tri
13     subjectClass // subjects must be of this type
14   );
15 }
```

POST-PROCESSING WITH N3 + COMUNICA

FSO uses some pattern matches that require post-processing with SPARQL update queries

```
Restored session: Sun Jan 23 11:54:45 CET 2022
mads@madss-air ~ % ifc-lbd fso -i
```

Recorded with asciinema

1. Load data into N3 store
2. Use Comunica to query the store
3. Serialize store content to file with N3 stream writer

```
1  async function componentConections(): Promise<void>{
2    const query = `
3      PREFIX fso:   <https://w3id.org/fso#>
4      INSERT{
5        ?e1 fso:connectedWith ?e2 .
6        ?e2 fso:connectedWith ?e1 .
7        ?e1 fso:feedsFluidTo ?e2 .
8        ?e2 fso:hasFluidFedBy ?e1
9      }
10     WHERE{
11       ?e1 fso:connectedPort ?p1 .
12       ?p1 fso:connectedPort ?p2 .
13       ?p2 fso:connectedComponent ?e2 .
14       ?p1 a fso:OutPort .
15       ?p2 a fso:InPort
```

```
    PREFIX fso:   <https://w3id.org/fso#>
    INSERT{
      ?e1 fso:connectedWith ?e2 .
      ?e2 fso:connectedWith ?e1 .
      ?e1 fso:feedsFluidTo ?e2 .
      ?e2 fso:hasFluidFedBy ?e1
    }
    WHERE{
      ?e1 fso:connectedPort ?p1 .
      ?p1 fso:connectedPort ?p2 .
      ?p2 fso:connectedComponent ?e2 .
      ?p1 a fso:OutPort .
      ?p2 a fso:InPort .
    }`;
  await this.executeUpdateQuery(query);
```

```
      INSERT{
        ?e1 fso:connectedWith ?e2 .
        ?e2 fso:connectedWith ?e1 .
        ?e1 fso:feedsFluidTo ?e2 .
        ?e2 fso:hasFluidFedBy ?e1
      }
      WHERE{
        ?e1 fso:connectedPort ?p1 .
        ?p1 fso:connectedPort ?p2 .
        ?p2 fso:connectedComponent ?e2 .
        ?p1 a fso:OutPort .
        ?p2 a fso:InPort .
      }`;
    await this.executeUpdateQuery(query);
  }
```

SOME QUERIES USE EXTENSION FUNCTIONS

```
1  async function segmentLengths(): Promise<void>{
2    const query = `INSERT{
3      ?seg ex:length ?d
4    }
5    WHERE{
6      ?seg a fso:Segment ;
7        fso:connectedPort ?port1 , ?port2 .
8      FILTER(?port1 != ?port2)
9      ?port1 omg:hasGeometry/fog::asSfa_v2-wkt ?p1 .
10     ?port2 omg:hasGeometry/fog::asSfa_v2-wkt ?p2 .
11     BIND(geosf:distance(?p1, ?p2, 3) AS ?d)
12   }`;
13   await this.executeUpdateQuery(query);
14 }
```

SOME QUERIES USE EXTENSION FUNCTIONS

```
1  async function segmentLengths(): Promise<void>{
2    const query = `INSERT{
3      ?seg ex:length ?d
4    }
5    WHERE{
6      ?seg a fso:Segment ;
7        fso:connectedPort ?port1 , ?port2 .
8      FILTER(?port1 != ?port2)
9      ?port1 omg:hasGeometry/fog::asSfa_v2-wkt ?p1 .
10     ?port2 omg:hasGeometry/fog::asSfa_v2-wkt ?p2 .
11     BIND(geosf:distance(?p1, ?p2, 3) AS ?d)
12   }`;
13   await this.executeUpdateQuery(query);
14 }
```

SOME QUERIES USE EXTENSION FUNCTIONS

```
1  // GEOJSON
2  // geosf:distance(p1, p2, decimals)
3  'http://www.opengis.net/def/function/geosparql/distance'(ar
4
5    // Number of decimals? (default = 8)
6    const decimals = args[2] != undefined ? parseFloat(args[2
7
8    // Make sure both args are literal values
9    if (args[0].termType === 'Literal' && args[1].termType ==
10
11     const p1 = parseWKT(args[0].value);
12     const p2 = parseWKT(args[1].value);
13
14     const a = p1[0] - p2[0];
15     const b = p1[1] - p2[1];
```

SOME QUERIES USE EXTENSION FUNCTIONS

```
1  // GEOJSON
2  // geosf:distance(p1, p2, decimals)
3  'http://www.opengis.net/def/function/geosparql/distance'(ar
4
5    // Number of decimals? (default = 8)
6    const decimals = args[2] != undefined ? parseFloat(args[2
7
8    // Make sure both args are literal values
9    if (args[0].termType === 'Literal' && args[1].termType ==
10
11     const p1 = parseWKT(args[0].value);
12     const p2 = parseWKT(args[1].value);
13
14     const a = p1[0] - p2[0];
15     const b = p1[1] - p2[1];
```

SOME QUERIES USE EXTENSION FUNCTIONS

```
9    if (args[0].termType === 'Literal' && args[1].termType ==
10
11     const p1 = parseWKT(args[0].value);
12     const p2 = parseWKT(args[1].value);
13
14     const a = p1[0] - p2[0];
15     const b = p1[1] - p2[1];
16     const c = p1[2] - p2[2];
17
18     const d: number = round(Math.sqrt(a * a + b * b + c * c
19
20     return DF.literal(d.toString(), DF.namedNode('http://ww
21
22   }
23
```

SOME QUERIES USE EXTENSION FUNCTIONS

```
13
14      const a = p1[0] - p2[0];
15      const b = p1[1] - p2[1];
16      const c = p1[2] - p2[2];
17
18      const d: number = round(Math.sqrt(a * a + b * b + c * c
19
20      return DF.literal(d.toString(), DF.namedNode('http://ww
21
22    }
23
24    // Just return literal "ERROR" if something fails
25    return DF.literal("ERROR");
26  }
```

SOME QUERIES USE EXTENSION FUNCTIONS

```
13
14      const a = p1[0] - p2[0];
15      const b = p1[1] - p2[1];
16      const c = p1[2] - p2[2];
17
18      const d: number = round(Math.sqrt(a * a + b * b + c * c
19
20      return DF.literal(d.toString(), DF.namedNode('http://ww
21
22    }
23
24    // Just return literal "ERROR" if something fails
25    return DF.literal("ERROR");
26  }
```

PLEASE JOIN THE EFFORT IN FURTHER DEVELOPING THESE OPEN SOURCE PROJECTS!

- IFC-LBD
- Comunica geoSPARQL