TRANSFORMING ENGINEERING

A Generic Cloud-based Round-trip Real-time 2D Revit BIM Editor

Jeremy Tammik
Principal Developer Consultant

About the Presenter

Jeremy Tammik
Principal Developer Consultant
Developer Technical Services
EMEA, Autodesk SARL

Jeremy is a member of the AEC workgroup of the Autodesk Developer Network ADN team, providing developer support, training, conference presentations, and blogging on the Revit API.

He joined Autodesk in 1988 as the technology evangelist responsible for European developer support to lecture, consult, and support AutoCAD application developers in Europe, the U.S., Australia, and Africa. He was a co-founder of ADGE, the AutoCAD Developer Group Europe, and a prolific author on AutoCAD application development. He left Autodesk in 1994 to work as an HVAC application developer, and then rejoined the company in 2005.

Jeremy graduated in mathematics and physics in Germany, worked as a teacher and translator, then as a C ++ programmer on early GUI and multitasking projects. He is fluent in six European languages, vegetarian, has four kids, plays the flute, likes reading, travelling, theatre improvisation, yoga, carpentry, loves mountains, oceans, sports, dancing, and especially climbing.

Introduction

Room Editor Base Functionality

- Determine simplified view of BIM rooms and furniture
- Store in cloud database
- Display simplified 2D graphics in any browser
- Edit furniture position and rotation
- Update database and BIM

Room Editor Update

- Display sheet, views and floor plan geometry
  - Revit add-in part implemented
- Improved 2D graphical editing
  - Integrated Raphaël.FreeTransform module
  - Unfortunately it ignores SVG view box transformation
- Store, edit and update non-graphical property data

Quotes on Three Fundamental Aspects

- Lazy
  - ... develop the three great virtues of a programmer: laziness, impatience, and hubris – *Larry Wall*
- Simple
  - Simplicity is the ultimate sophistication – *Leonardo da Vinci*
  - There is no greatness where there is no simplicity – *Leo Tolstoy*
  - KISS
- Perfect
  - Perfection is achieved, not when there is nothing more to add, but when there is nothing left to take away – *Antoine de Saint-Exupéry*

Data Source, Repository and Consumer Client

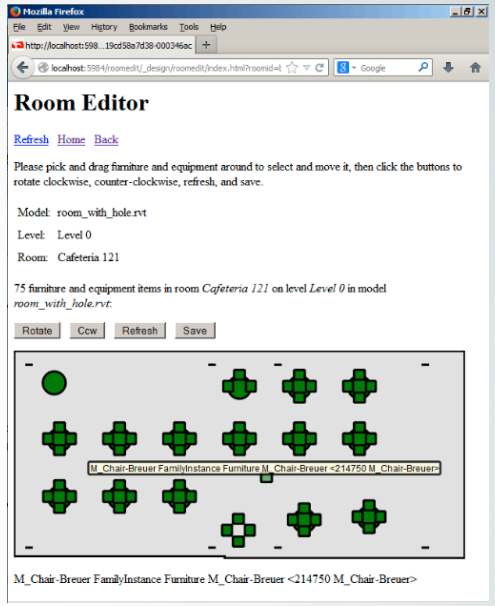
- BIM – Building Information Model
- Cloud-based data repository
- 2D rendering on mobile device

Real-time Editing Triggers Database and BIM Update

- Graphical room editor on mobile device
- Update cloud database
- Reflect real-time changes in BIM

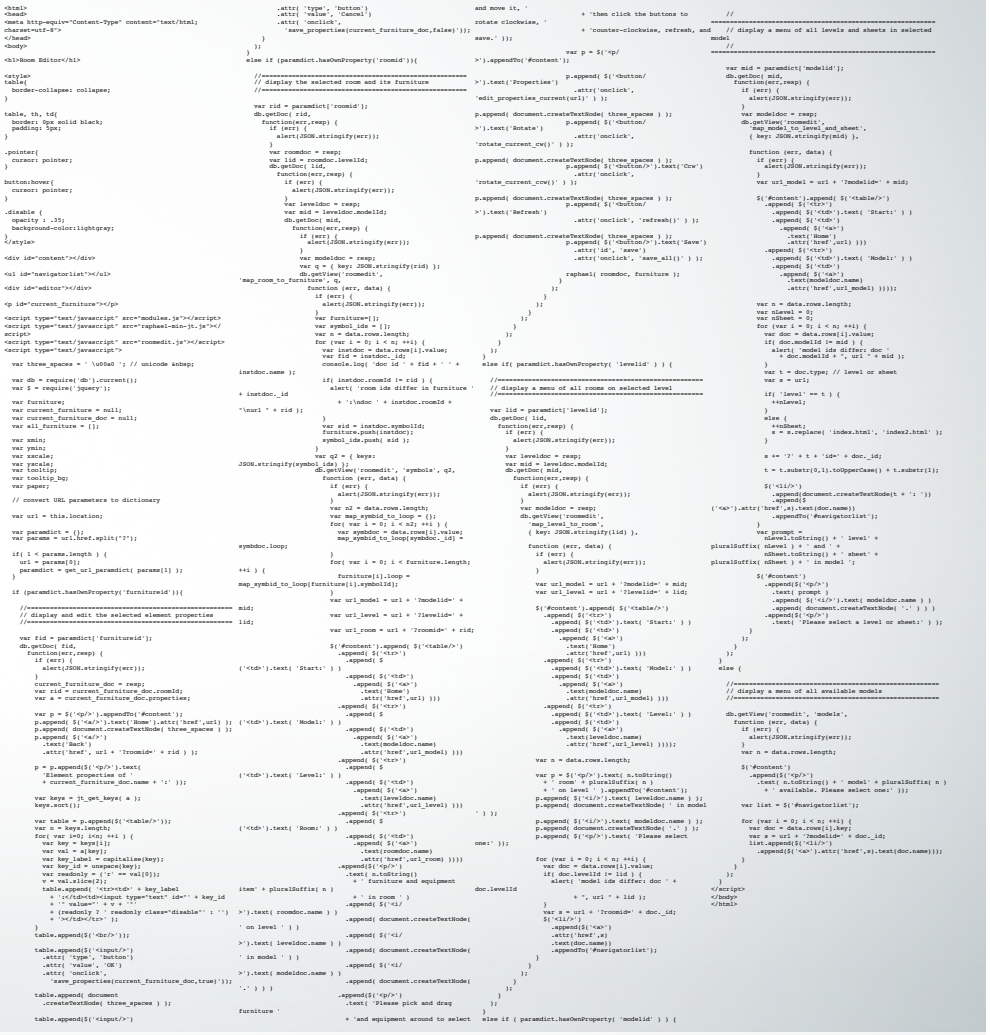
Sixty Second First Impression Demo

- Simple navigation
  - Home page: list all models, select one
  - Model selected: list all levels, select one
  - Level selected: list all rooms, select one
  - Room selected: display 2D graphical editor, click and drag furniture
  - Furniture selected: display and edit properties
  - Save: update database
- Note RESTful URLs in the address bar
  - The database interaction is RESTful as well



Free and Simple

- 100% open source
- 200 hours research
- < 8 hours to rebuild
  - Install components
  - Re-implement from scratch
  - Two implementation files
    - index.html (474 lines)
    - roomedit.js (358 lines)



Update Splits Functionality

- Index.html
- Roomedit.js
- One day to extract data from Revit
- One day to enhance database, editor, update etc.

Architecture

Base Technologies

- BIM – Revit
- Data repository – NoSQL
- Rendering and editing – HTML and SVG

Implementation Environment

- Revit BIM – Revit .NET add-in
- NoSQL Database – Apache CouchDB
- HTML, SVG – JavaScript, jquery, db, Raphaël

Simpler Still

- Same origin policy
- Server-side scripting
- Two components instead of three

The Two and Only Projects

- Revit add-in
- CouchDB database

Cloud Database

NoSQL

- “Not only SQL”
- Next generation database paradigm
- Address some of the points: non-relational, distributed, open-source, scalable, huge amounts of data
- Frequent other characteristics: schema-free, easy replication support, simple API, eventually consistent, i.e., BASE, not ACID
  - http://nosql-database.org, https://en.wikipedia.org/wiki/NoSQL, http://www.mongodb.com/nosql-explained

ACID versus Base and CAP Theorem

- ACID
  - Atomicity, Consistency, Isolation and Durability guarantee that database transactions are processed reliably
- CAP Theorem
  - The ACID paradigm cannot simultaneously guarantee consistency, availability and partition tolerance (distributed system)
- BASE
  - Basic Availability, Soft-state, Eventual consistency
  - Not guaranteed to be in a consistent state at a given moment
  - Consistency is guaranteed, eventually

http://thebuildingcoder.typepad.com/blog/2014/05/views-displaying-given-element-svg-and-nosql.html#5

CouchDB Database Implementation

- Everything is a document
- All documents are JSON
- Every document has built-in id and revision
- The database design is also a document
- The design defines views and attachments

CouchDB Database Interaction

- Relax!
- All interactions are REST
- Management console is Futon

  http://127.0.0.1:5984/_utils/
- Access all documents

  http://127.0.0.1:5984/roomedit/_all_docs
- Include full doc data, not just id and revision

  http://127.0.0.1:5984/roomedit/_all_docs?include_docs=true

BIM Model

- Model – a RVT project file
- Level
- Room
- FamilyInstance represents furniture or equipment
- FamilySymbol defines geometry

BIM Object Graphics

- Room has boundary loops and can contain holes
- FamilySymbol has a single boundary loop
- FamilyInstance has a 2D placement
  - Translation
  - Rotation

BIM Object Relationships

- CouchDB ids are Revit unique ids
- Family instance → room → level → model
- Family instance → symbol

NoSQL Database Structure

- DbObj base class
- DbModel
- DbLevel
- DbRoom
- DbFurniture
- DbSymbol

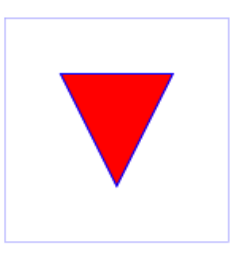
Database Object Relationships

- DbFurniture.symbolId → DbSymbol
- DbFurniture.roomId → DbRoom
- DbRoom.levelId → DbLevel
- DbLevel.modelId → DbModel

Database Object Graphics and Placement

- All graphics represented by SVG path element data

```
<svg width="4cm" height="4cm" viewBox="0 0 400 400"
     xmlns="http://www.w3.org/2000/svg" version="1.1">
  <rect x="1" y="1" width="398" height="398"
        fill="none" stroke="blue" />
  <path d="M 100 100 L 300 100 L 200 300 z"
        fill="red" stroke="blue" stroke-width="3" />
</svg>
```



JSON Symbol Database Document

- Family symbol
- Define geometry


```
{
   "_id": "11cc6e52-519e-49b2-9813-c9561b59a1fd-0005f5fc",
   "_rev": "1-d575ca095533db4ccbed9f7ab2607a12",
   "loop": "M-191 922 L190 922 216 862 190 859 -191 859 -216 862 -216 919Z",
   "type": "symbol",
   "description": "FamilySymbol Furniture <390652 Table ronde a chaises>",
   "name": "Table ronde avec chaises - 01"
}
```

JSON Instance Database Document

- Furniture doc represents family instance and defines
- Relationship to room and family symbol
- Placement = transform = translation + rotation


```
{
   "_id": "11cc6e52-519e-49b2-9813-c9561b59a1fd-0005f65b",
   "_rev": "1-c1b4fc969181267b55dab4c6857fc5d7",
   "roomId": "cbe571b0-0593-4350-a8e6-abf3c9239325-00061210",
   "symbolId": "11cc6e52-519e-49b2-9813-c9561b59a1fd-0005fe6d",
   "transform": "R-90T-10429,1020",
   "type": "furniture",
   "description": "FamilyInstance Furniture <390747 Canapé 3 places>",
   "name": "Canapé 3 places"
}
```

CouchDB Views

- A view defines a map and optional reduce function
- The map produces key-value pairs
- Reduce produces an accumulation

```
exports.models = {
  map: function (doc) {
    if( 'model' == doc.type ) {
      emit(doc, null);
    }
  },
  reduce: function (key, values, rereduce) {
    return sum(values);
  }
}
```

Room Editor Views

- models
- levels
- rooms
- furniture
- symbols
- map_room_to_furniture
- map_level_to_room
- map_model_to_level

```
rooms = {
  map: function (doc) {
    if( 'room' == doc.type ) {
      emit(doc, null);
    }
  }
};
```

```
map_level_to_room = {
  map: function (doc) {
    if( 'room' == doc.type ) {
      emit(doc.levelId, doc);
    }
  }
};
```

RESTful CouchDB Document and View Access URLs

- Specific document

  http://127.0.0.1:5984/roomedit/11cc6e52-519e-49b2-9813-c9561b59a1fd-0005f5fc

- Specific view

  http://127.0.0.1:5984/roomedit/_design/roomedit/_view/models

- Specific key in view

  http://127.0.0.1:5984/roomedit/_design/roomedit/_view/map_level_to_room?key=%22933c4a06-93b8-11d3-80f8-00c04f8efc32-0000001e%22

Entire CouchDB Application Definition

- Kanso loads CouchDB

```
roomedit/data/room_model_9.json
roomedit/index.html
roomedit/kanso.json
roomedit/lib/app.js
roomedit/lib/views.js
roomedit/raphael-min-jt.js
roomedit/roomedit.js
```

http://kan.so

- kanso.json

```
{
  "name": "roomedit",
  "attachments": ["index.html",
    "raphael-min-jt.js"],
  "modules": ["lib"],
  "load": "lib/app",
  "dependencies": {
    "attachments": null,
    "modules": null,
    "properties": null,
    "db": null,
    "jquery": null
  }
}
```

Index.html

- HTML scaffolding
- JavaScript query section
- Raphaël SVG generation and interaction
- RESTful database updates
- Called with
  - No argument: list all models
  - Model id: list all levels in model
  - Level id: list all rooms on level
  - Room id: display graphical editor and enable database updates

Minimal Predefined HTML Scaffolding

```
<h1>Room Editor</h1>

<div id="content"></div>

<ul id="navigatorlist"></ul>

<div id="editor"></div>

<p id="current_furniture"></p>

<script type="text/javascript" src="modules.js"></script>
<script type="text/javascript" src="raphael-min-jt.js"></script>
```

- Populated entirely using JavaScript adding HTML and SVG nodes using jquery, raphael and db for CouchDB queries

List all Models

- No input parameter
- View ‘models’, no key
- Populate navigator list

```
db.getView('roomedit', 'models',
  function (err, data) {
    if (err) {
      return alert(err);
    }
    var n = data.rows.length;
    for (var i = 0; i < n; ++i) {
      var doc = data.rows[i].key;
      var s = url + '?modelid=' + doc._id;
      $('<li/>').append($('<a>')
        .attr('href',s)
        .text(doc.name))
      .appendTo('#navigatorlist');
    }
    var p = $('<p/>').appendTo('#content');
    p.append( $('<a/>').text('Home').attr('href',url) );
    p.append( document.createTextNode( three_spaces ) );
    p.append( $('<a/>').text('Back').attr('href',url) );

    $('<p/>')
      .text( 'Please select a model in the list below.' )
      .appendTo('#content');

    $('<p/>')
      .text(n.toString() + ' model’ + pluralSuffix( n ) + dotOrColon( n ) )
      .appendTo('#content');
  }
);
```

List all Levels in Selected Model

- Input parameter ‘modelId’
- Get model document itself
  - Display current selection
- View ‘map_model_to_level’ with model id key
  - Populate navigator list

List all Rooms on Selected Level

- Input parameter ‘levelId’
- Get level document itself → model id
- Get model document
  - Display current selection
- View ‘map_level_to_room’ with level id key
  - Populate navigator list

Display and Edit a Selected Room

- Input parameter ‘roomId’
- Get room document itself → level id
- Get level document → model id
- Get model document
  - Display current selection
- View ‘map_room_to_furniture’ with key room id
- Retrieve family instances → symbol ids
- View ‘symbols’ with list of symbol keys
  - Populate and display SVG editor

Nested Database Queries and Callback Functions

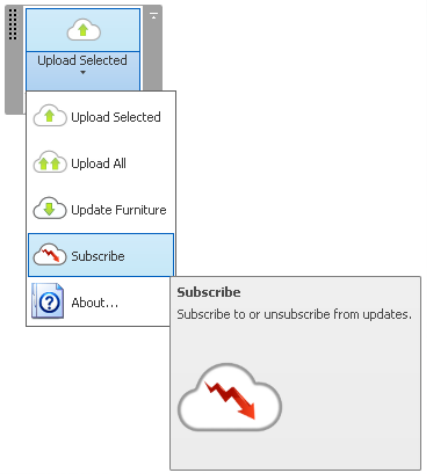
- Nested asynchronous database queries
- Each nested query depends on previous results
- Page cannot be displayed until all complete
- Nested callback functions

SVG Room Editor

- Using jquery and Raphaël
- Input room + furniture populated with symbol SVG path
- Determine size and aspect ratio
- Instantiate paper = canvas
- Place room and attach tooltip events mouseover + out
- Place furniture and attach identification, drag and tooltip events

Revit Add-in

Revit Add-in



- Determine 2D boundary polygon loops
- Upload model to database
- Download changes from database
- Subscribe to real-time updates
- Timer and external event

Revit Add-in Interaction with CouchDB

- How does the Revit add-in access CouchDB?
  - Upload model data
  - Retrieve database changes
  - Subscribe to changes
- DreamSeat

  https://github.com/vdaron/DreamSeat
  - Easy!
  - Good job!

Conclusion

Challenges

- Many!
- NoSQL, CouchDB, views, CouchApp and Kanso
- JavaScript, SVG, event handling, mobile devices
- Mapping to negative Y axis and restore for update
- Handle nested database callback functions
- Retrieving database changes
- Idling versus external event
- This is a simple learning sample
- Infinite possible real-life applications

Latest News – On the Road

- PouchDB – put it in your pocket
- Open-source JavaScript database
- Inspired by Apache CouchDB
- Designed to run in the browser
- Work offline as well as online
- Store data locally while offline
- Synchronize with CouchDB when online
- Keep data in sync no matter where
  - http://pouchdb.com

Conclusion

- NoSQL is great!
- Open source is free and effective
- REST and JSON is powerful and simple to work with
- Server-side scripting pays off
  - Avoid programming on the mobile device itself
- KISS!

Room Editor Documentation and Materials

- The Building Coder Revit API blog and categories
  - http://thebuildingcoder.typepad.com
    - Cloud
    - Desktop
    - Mobile
- GitHub repositories
  - https://github.com/jeremytammik
    - RoomEditorApp – Revit add-in
    - Roomedit – CouchDB definition
- Room editor playground
  - http://jt.iriscouch.com/roomedit/_design/roomedit/index.html

Autodesk is a registered trademark 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 and services 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. © 2014 Autodesk, Inc. All rights reserved.