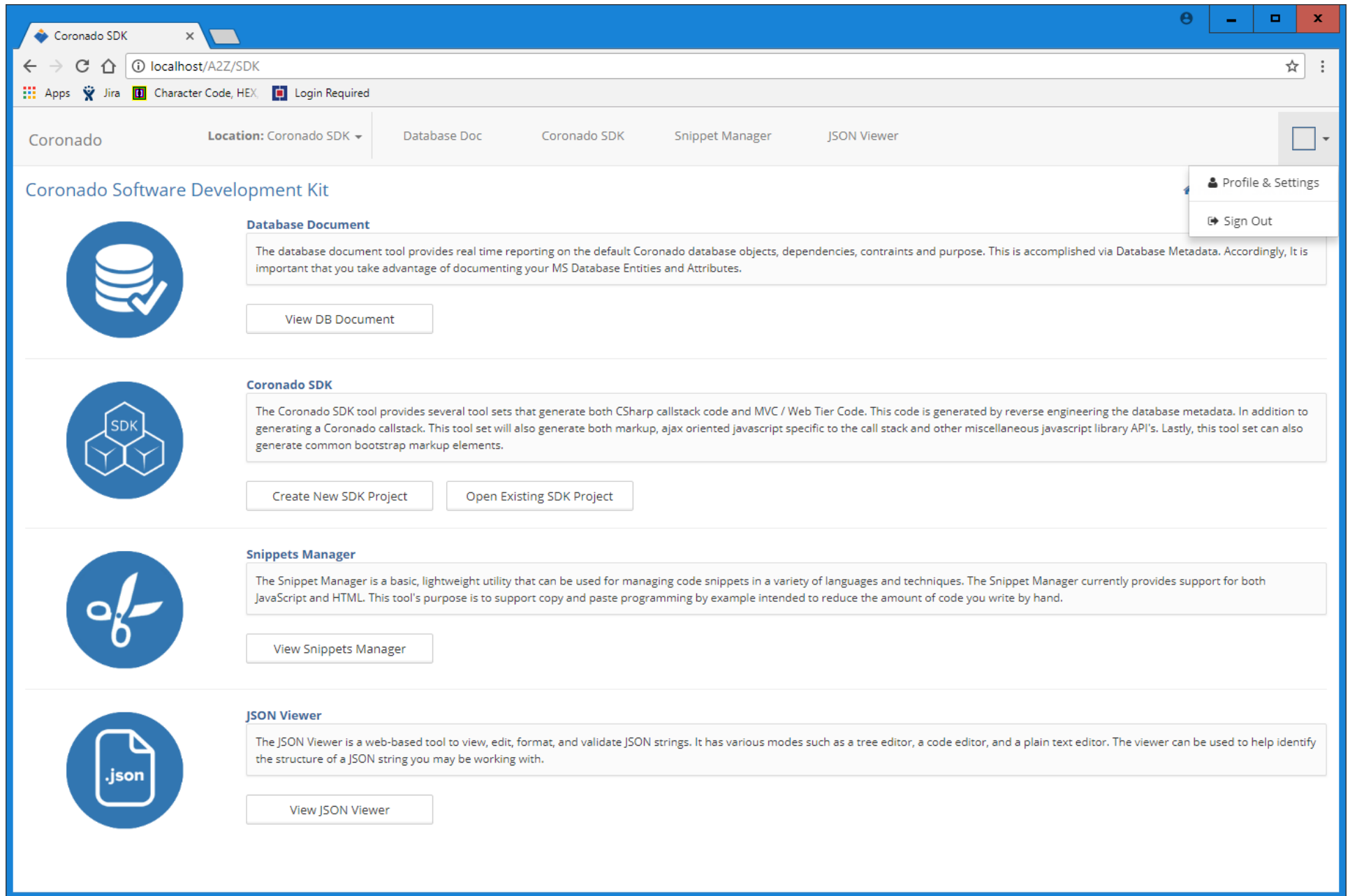Open SDK Integration

The following screen capture demonstrates a stable working copy of the SDK that is being rewritten to support other languages besides JavaScript, ASP.NET and C#. The purpose of this SDK is to provide a tool that allows you to plan, design and document your project where you can then generate coded functionality as needed. This code is then incorproated with your IDE of choice.

The following screen shots illustrate real working software written by Sean McNeill

FOR YOUR EYES ONLY

Current dashboard, next version will contain key performance analytics and charts

Code generation is only part of a larger project. Managing projects can start with a simple file and be cast into object models and saved in a data source of your choice

Example of generic Snippets Manager

Example of SDK Design Screen. The SDK creates SOA API's with a service layer. This call stack can be generated via reverse engineering the database layer.

After a taxonomy is selected, you click the load objects button, a progress indicator displays while the call stack is generated (2 to 5 seconds)

After all the project objects are marshalled and serialized, a project tree is loaded allowing you to see the source code manifest

Source code is presented as a documented project where you can add additional project information.

The SDK also serves as your point of API definition and documentation

Lastly, the source code is made available for your review. Changes can be made at this stage before the code is written to disk, methods can be added, object names can be changed. Full refactor capabilities

In addition to the back end call stack, we also assemble building blocks for the web tier, specifically ASP.NET MVC w/Rest Services and jQuery libraries into the API

Here you can copy JavaScript or just review what the objects look like that you are interfacing with

The SDK will also write a service layer JS file to serve as a pointer to your rest services

Example: `url: COM.COM_CONTACT.GetContactById('XXX-000-999');`

Object sensitive JavaScript generation as demonstrated in the next few screens

Coronado   **Location:** Coronado SDK ▾   Database Doc   Coronado SDK   Snippet Manager   JSON Viewer   Sean ▾

First, solve the problem. Then, generate the code.   ⌂ Home

🔧 Designer Options

DB Model | **Project Tree**

- 🏠 Project Home
  - 💾 Save Project
  - 📂 Open Project
  - 🗒 Export Code
  - 🄲 Coronado.COM
  - 🌐 Web Tier SDK
    - ⚙ Rest Services
    - 🔤 JavaScript Service Object
    - 🔤 JavaScript Snippets
    - <> MarkUp Snippets
- 🗄 PW-IT-SEAN-W10.CORONADO_DB
- {} View SDK Object

⚙ Sandbox

🔤 JavaScript Snippets

Model Object   Member Object ▾   JavaScript Snippets   - Standard Page Ready Snippet ▾

```
1   <!-- Document Ready DOM/JavaScript Handler -->
2   @section scripts {
3   <script type='text/javascript' src='~/Areas/Rest/Scripts/COM.js'></script>
4   <script>
5       $(document).ready(function () {
6           pageConfig();
7       });
8       // Page Configuration
9       var pageConfig = function () {
10          // Add to Breadcrumb
11          addBreadCrumb('<a href='#'>XXX</a>', false);
12          addBreadCrumb('Request Details', true);
13      }
14  </script>
15  }
16
```

Select a Modeled Snippet...
- Standard Ajax GET Snippet
- Standard Ajax POST Submit Handler
- JSON Object POST Submit Handler
- Form Binding w/Bootstrap Markup
- DataTable Binding
Select Misc JS Snippet...
- Standard Page Ready Snippet
- Toastr Snippet
- SlimScroll Snippet
- Dynamic Select Options
- Setting Element Heights

Coronado SDK

① localhost/A2Z/SDK/CallStack

⚏ Apps 🏉 Jira 🔲 Character Code, HEX 🔲 Login Required

Coronado   **Location:** Coronado SDK ▾   Database Doc   Coronado SDK   Snippet Manager   JSON Viewer   Sean ▾

First, solve the problem. Then, generate the code.   🏠 Home

🔧 **Designer Options**   ⚙ **Sandbox**

DB Model	**Project Tree**

- 🏠 Project Home
  - 💾 Save Project
  - 📂 Open Project
  - 📑 Export Code
  - C# Coronado.COM
  - 🌐 Web Tier SDK
    - ⚙ Rest Services
    - �📱 JavaScript Service Object
    - T JavaScript Snippets
    - <> MarkUp Snippets
- 🗄 PW-IT-SEAN-W10.CORONADO_DB
- {} View SDK Object

T **JavaScript Snippets**

Model Object   [Member Object ▾]   JavaScript Snippets   [- DataTable Binding ▾]

```
1   <!-- Data Grid Markup -->
2   <div id="dataGridContainer">
3       <table id="dataGrid" class="table table-striped table-bordered"></table>
4   </div>
5
6   var gridData = [];
7
8   // Member DataTable Binding
9   var populateDataGrid = function() {
10      $('#dataGridContainer').html('<table id="dataGrid" class="table table-striped table-bordered"></table>');
11      dtGrid = $('#dataGrid').DataTable({
12          //paging: false,
13          //bFilter: false,
14          //bSort: false,
15          //bInfo: false,
16          data: gridData,
17          columns: [
18              { title: 'Member Id', data: 'MemberId' },
19              { title: 'Member Type', data: 'MemberType' },
20              { title: 'Member Class', data: 'MemberClass' },
21              { title: 'Display Name', data: 'DisplayName' },
22              { title: 'Common Name', data: 'CommonName' },
23              { title: 'Given Name', data: 'GivenName' },
24              { title: 'Other Name', data: 'OtherName' },
25              { title: 'Partition', data: 'Partition' },
26              { title: 'Source', data: 'Source' },
27              { title: 'Surrogate Key', data: 'SurrogateKey' },
28              { title: 'Read Only', data: 'ReadOnly' },
29              { title: 'Is Hidden', data: 'IsHidden' },
30              { title: 'Last Modified Uts', data: 'LastModifiedUts' },
31          ],
32          columnDefs: [{
33              targets: 0,
34              searchable: true,
35              orderable: true,
36              //className: "text-center",
```

// End of Screen Captures of Working SDK Prototype
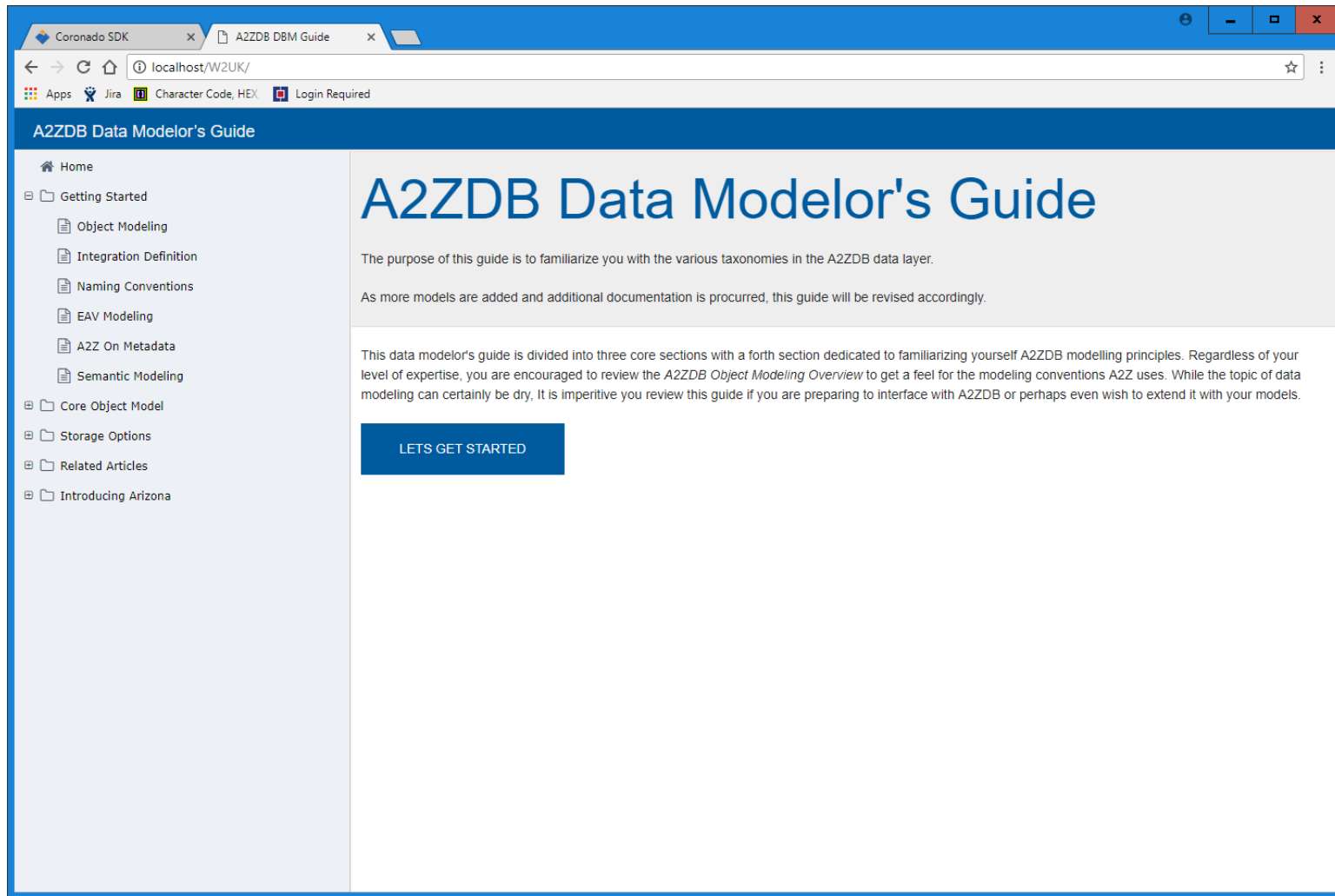
My Prototype of W2UI – Simple Documentation. I am looking into creating a state factory to manage how your containers unfold based on user preference. I am also going to switch over to css sensitive SVG iconography with png fallback as well as a theme engine that supports (MS Cobalt, Amazon, Google Material, GitHub, Bootstrap and Twitter styles) as well as design templates that are maintained in a CMS (Content Management System). Basically, eventually the CMS component will spit out docs like this with even more component options as well as serve as the basis for auto generated application interfaces using pure JSON. Even cooler is these docs will be able to stand alone with static json options or run with live db connections via rest. Everything you saw in the SDK is going into w2ui

A2ZDB Data Modelor's Guide

A2ZDB Object Modeling Principles

Document Revision: 1.0 - 03/30/2018
Author(s): Sean McNeill

The A2Z object model is first and foremost designed for an enterprise perspective that can be described as a holistic top-down logical model commonly refered to as *Entity Attribute Value* or EAV model. What this means is the model is created with a different, perhaps more abstract approach than one would likely encounter with a traditional *View Integration Model*. As a result, it allows you to sustain and build upon your domain taxonomy from a holistic perspective that will provide maximum scalability and effeciency into your problem domain through natural software accreation and data federation.

The benefit of using this top-down approach is how well the model itself flexes toward your domain of interest without the need for refactoring callstacks into data storage mediums because a column, table, property or new document was added or changed. We have all experienced this problem one time or another. The reason this benefit can be realized is because the physical objects and attributes that transport the data do not describe the data the same way one might find in traditional *View Integration Models*.

The following example illustrates this EAV principle at it's core.

Top-Down Logical Model

EAV: Father of All Object Models

- Everything is an "ENTITY"
- All "ENTITY" data has an "ENTITY_TYPE"
- An "ENTITY" can have a "RELATIONSHIP" with another "ENTITY"
- All "RELATIONSHIPS" have a "RELATIONSHIP_TYPE"
- An "ENTITY" can have one or more "ATTRIBUTES"
- An "ATTRIBUTE" has an "ATTRIBUTE_TYPE"
- An "ATTRIBUTE" can have one or more "ATTRIBUTE_VALUES"

Reference: EAV Modeling for a broader understanding and appeal.

A2ZDB Data Modelor's Guide

- 🏠 Home
- ⊞ 📁 Getting Started
- ⊟ 📁 Core Object Model
  - ▦ A2Z - Arizona Object State Model
  - ▦ CMS - Content Management System Model
  - ▦ COM - Community Object Model
  - ▦ DEV - DevOps Object Model
  - ▦ DNA - Dashboard Normilization and Archive
  - ▦ EAV - Entity Attribute Value Object Model
  - ▦ GEO - Geographic Data Object Model
  - ▦ MDR - Metadata Repository Object Model
  - ▦ MSG - Message Delivery Object Model
  - ▦ OPS - Operations & Management Object Model
- ⊞ 📁 Storage Options
- ⊞ 📁 Related Articles
- ⊞ 📁 Introducing Arizona

COM: Community Object Model

The Community Object Model is a core taxonomy schema as this is where member's of the community are identified, archetyped and extended. The Community Object Model is the who's who of A2ZDB taxonomy. For an in-depth review of this model, please refer to the article titled "Anatomy of a Taxonomy Schema". Like metadata, Member data is often exdended into just about every other taxonomy schema A2ZDB supports.

Integration Definition

Member Metadata Repository

MDR_MEMBER

MDR_MEMBER_TAG

Community Object Model

COM_MEMBER

COM_CONTACT

COM_LOCATION

COM_RELATIONSHIP

COM_ROLE

COM_MEMBER_DOCUMENT

COM_OBJECT_NOTATION

Related Definitions

Metadata Taxonomy Definitions

Member Entity Taxonomy

Contact Entity Taxonomy

Location Entity Taxonomy

A2Z Relational Database Support

Commentary required