



Go Pro with 2sxc
REST and JS APIs

Web Modules / Apps Today



Alle Hochbau (16) Tiefbau (19) Rückbau (15) Umbau (1) Kundenarbeit (3)

Rückbau zweier Mehrfamilienhäuser mit Garagenboxen, Carmennaweg 7/9, Chur
Rückbau in einem Wohnquartier. Daher ist ein leiser Hybrid...
Hitachi 225... [Details](#)

Baumeisterarbeiten Neubau Mehrfamilienhaus, im Bovel, Maienfeld
In Maienfeld entsteht ein Mehrfamilienhaus mit einer Naturstein-Umgebungsmauer... [Details](#)

Lehnenbrücke Cresta 1+2, Ersersstrasse, Schiers
Neubau Lehnenbrücken
Zufahrtsstrasse... [Details](#)

You can have this in minutes for free, when you leverage 2sxc

- Data in a DB
- Edit-dialog → 50%
- WYSIWYG TinyMCE
- Asset Management → 80%
- Data-Management, Import/Export, Versioning, Data-Query, Multi-Language, Permissions, → 95%
- Advanced: Custom GUIs, Data-Interfaces and Custom WebAPIs → 99%

Overview



- Developing JS in 2017
- REST-ful WebAPI
- Using REST in jQuery, Angular, React, KO
- JS API to execute CMS Actions
- GUI APIs for Toolbars & Button
- Background: \$2sxc, edit-api and more



Web in 2017 is JavaScript

This is not Your Grandpa's JS



- @-annotations
- modules
- exports
- constructors
- `let x = 7;`
- Arrows (lambda) =>
- Classes & Interfaces
- Rx & Observables
- Multi-line strings



Learn if you're a Web Dev in 2017



- node & npm
- git
- TypeScript and ES7
- Patterns and Architecture
 - Dependency Injection, SoC, SRP, ...
 - Reactive (Stream-based) programming
 - One way data flow
 - Functional programming...

Optionally learn this:

- Gulp / Grunt (so 2015)
- WebPack (so 2016)

Slowly unlearn this:

- Two-way data binding
- Promises
- jQuery
- Server rendered HTML
- MVC



The world at REST



REST

Representational State Transfer

Examples



- GET /content/Person
- GET /content/Person/17
- GET /query/All-Employees
- GET /query/Employees?team=accounting
- GET /query/everything-for-blog-details
- POST /content/Person
 - Body: { name: Daniel, partner: 0 }
 - Result: 512
- POST /content/Person/512
 - Body: { partner: 19 }

REST in Pieces



1. URL-Pattern...
2. ...targets the resource you're talking to
3. HTTP-Verbs for action
 1. GET, POST, JUMP, ...
4. Headers for more

1. ?x=y for params
2. HTTP-body for payload or complex params
3. Very open to interpretation

{JSON}

Daniel the iJungleboy – est '78



- Founded 2sic in 1999
- Architect of 2sxc since 2012
- Angular since 2014
- Blogger, daddy, nerd, ceo, checklist-freak, world-traveler, ...





2sxc and REST – since 2014

+ external Endpoint since 2016

2sxc Speaks REST since 2014



- Verbs: GET POST
- All: [root]/content/[typename]
- 1: [root]/content/[typename]/[id]
- Query: [root]/query/[queryname]

Real Example with a Query



Profil

Über mich

Name Hans Host

Adresse Musterstrasse 7
7070 Mustern

Geburtsdatum 19.08.1980

Telefon 000 333 30 30

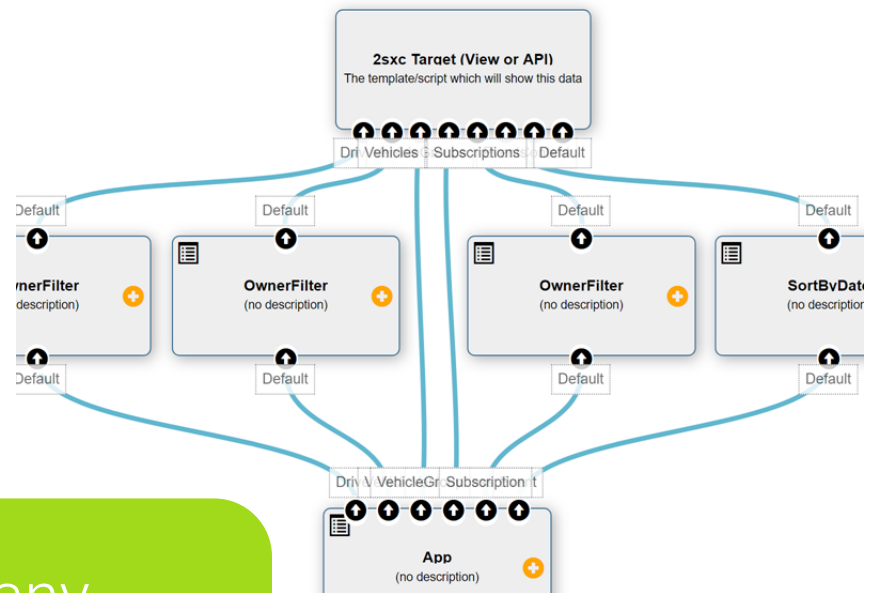
Email hans.host@muster.ch

Meine Fahrzeuge

Rufname	Modell	Gruppe	Gewicht [kg]
Demo 1	Fendt 1000 Vario	Farmstock	14500
Demo 2	MTD RF 125	Garden Standard	150
Demo3	Fendt 800 VARIO	Farmstock	5120

Meine Anmeldungen

Veranstaltung	Datum
Test Veranstaltung 2	26.04.2016 - 27.04.2016
Test Veranstaltung 1	29.03.2016 - 30.03.2016



This returns many different streams in 1 request = fast!

34

200

GET

HTTP

[www.tractorpulling.ch /de/Anmeldung](http://www.tractorpulling.ch/de/Anmeldung)

40

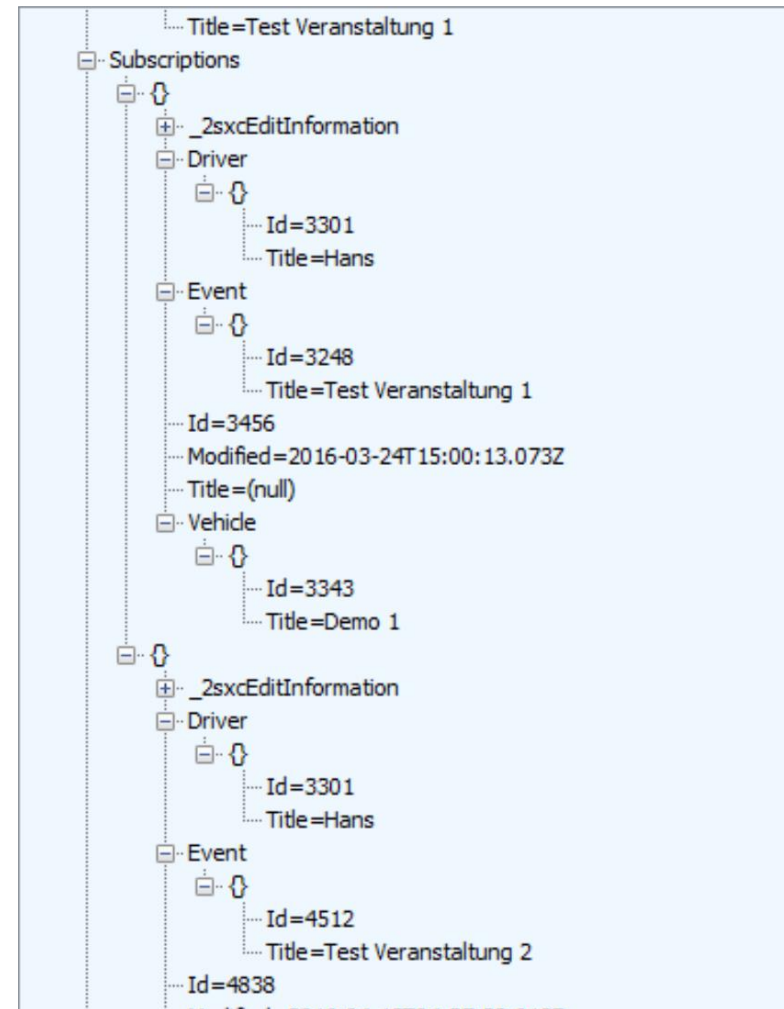
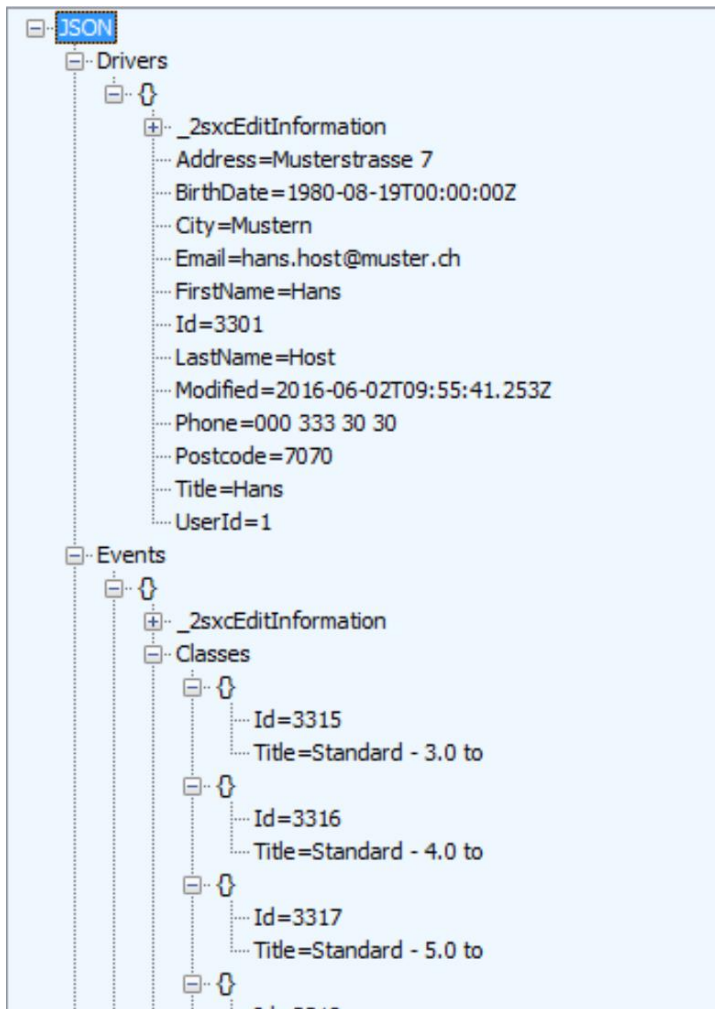
200

GET

HTTP

[www.tractorpulling.ch /de/DesktopModules/2SIC/API/app-query/MyProfile](http://www.tractorpulling.ch/de/DesktopModules/2SIC/API/app-query/MyProfile)

Resulting JSON w/Streams of Driver, Events, Subscriptions, ...



[root] Path



- The DNN API-Root
 - this changes by
 - portal
 - language
- + /app/auto/
- Auto-Resolve w/\$2sxc



```
var sxc = $2sxc(tag|mid);
```

```
var promise = sxc.webApi.get("content/Person"); // easy
```

/app/**auto**/etc.



- Use “auto” to auto-detect based on the current module
 - Practical
 - Security-aware for this module (view/write permissions)
 - Context-aware – can deliver items assigned to current module (private content)
 - Side note: old code uses /app-content/

.../app/auto/content/[type]/[id]



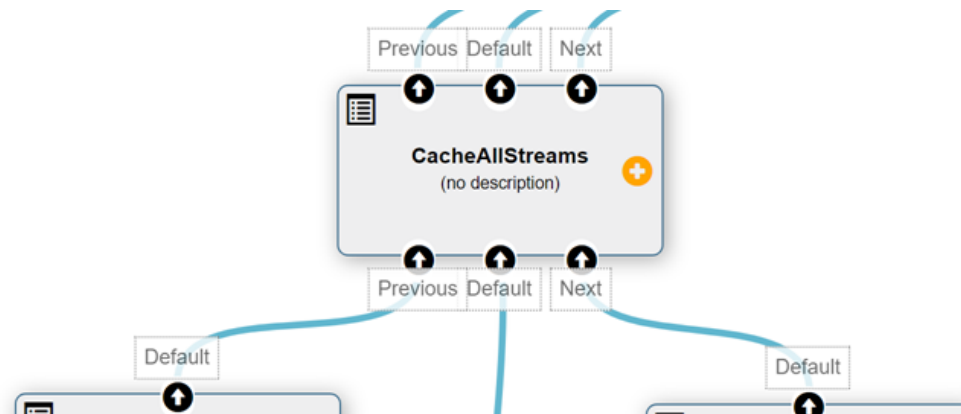
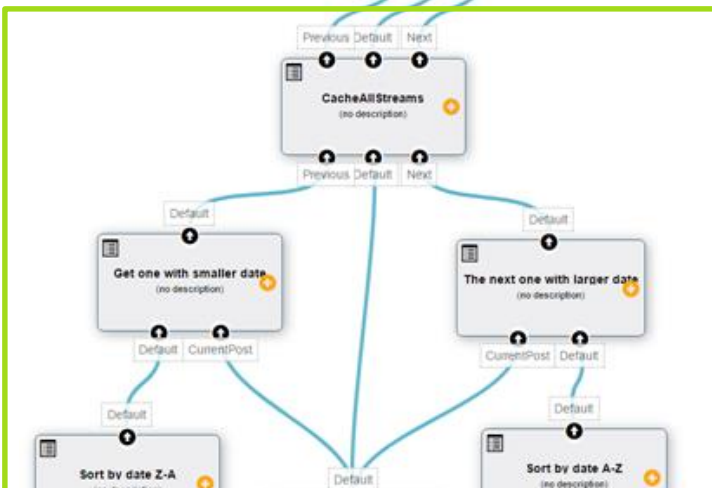
- Access all items of a type
- Access one item of a type
- Create a new item
- Modify an existing item

→ if permissions allow...

.../app/auto/query/query-name



- Access pre-built queries
- .../app/auto/query/
- Provide parameters after query-name
 - .../blog-items?category=recipes



← Read previous



Hardcore: 2sxc
9.0 with Entity
Framework Core
1.1

New: DNN Instant Theme / Skin with Bootstrap 3 & SASS

5/19/2017

Daniel Mettler



Bootstrap INSTANT



Want to create high-quality DNN themes within minutes? Introducing the DNN Instant Theme with Bootstrap 3, SASS and many features.

Read next →



New App:
Courses / Events
with Registration



Default

REST Security / Permissions



- Default is no-access
- Typical settings for...
 - Visualizers: Enable “read” for view-users
 - Forms: Enable “create” for view-users
 - Owned data:
 - Enable “c” for viewers
 - Enable “r/u/d” for owners
 - Public API: Enable “read” for anonymous

A screenshot of a web application interface for managing permissions. The title is 'Berechtigungen' (Permissions). It features a table with columns: ID, Name, Bedingung (Condition), and Gewährung (Grant). There are two rows of security rules. Below the table is a list of data entities with their respective counts and permissions. A red arrow points to the 'Subscription' row.

ID	Name	Bedingung	Gewährung
3452	Security Rule	SecurityAccessLevel.View	
5273	Security Rule	Owner	crud

Name	Count	Description	Permissions
Driver	266	Master data for a driver.	9
Event	6	Master data for an event.	23
<u>Subscription</u>	521	<u>Master data for subscription.</u>	3
Title	3	Just a title.	1
Vehicle	379	Master data for vehicle.	7
VehicleClass	19	Group and weight.	3
VehicleGroup	11	Group of vehicle.	1



REST in JS Frameworks

“External” Access



- Plain REST with URLs
- No helpers available / necessary
- Needs the app-name in the URL, as there is no module-context
- Req. permissions on “anonymous”

/desk.../2sxc/api/app/blog/content/blogpost

AngularJS 1.x and Angular



- Use 2sxc4ng to bootstrap your application
- \$http is automatically configured to
 - Include DNN headers and security headers
 - Auto-resolve paths like “content/...” etc.
- `var promise = $http.get(“content/blogpost”);`
- You can also use these services
 - Content
 - Query
 - Toolbar
- Angular 4+ see next session 😊

Any other JS in 2sxc views



- Get controller using `$2sxc(tag | mid)`
- Then resolve path using `resolveServiceUrl`

```
var url = sxc.resolveServiceUrl("content/Person");
```

- *please help by creating libraries for KO, React, Ember, Vue*



Custom REST Endpoints

With C# WebAPIs

Just create your own WebAPI



- Place C# files in folder [appname]/api/
- Accessible as [root]/app/auto/api/...
- Eg.
 - MailController.cs
 - bool Send(...)
 - [root]/app/auto/api/Mail/Send

```
FormController.cs x  _Support Request.cshtml  _Generic Smart Form Example.cs
1  using DotNetNuke.Security;
2  using DotNetNuke.Web.Api;
3  using System.Web.Http;
4  using ToSic.SexyContent.WebApi;
5  using System.Collections.Generic;
6  using System;
7  using System.Linq;
8  using System.Web.Compilation;
9  using System.Runtime.CompilerServices;
10 using DotNetNuke.Services.Mail;
11 using Newtonsoft.Json;
12
13 public class FormController : SxcApiController
14 {
15
16     [HttpPost]
17     [DnnModuleAuthorize(AccessLevel = SecurityAccessLevel.Anonymous)]
18     [ValidateAntiForgeryToken]
19     public void ProcessForm([FromBody]Dictionary<string,object> contactFormRequest)
20     {
21
22         // Pre-work: help the dictionary with the values uses case-insensitive
23         contactFormRequest = new Dictionary<string, object>(contactFormRequest);
24
25         // test exception to see how the js-side behaves on errors
26         // throw new Exception();
27
28
29         // 0. Pre-Check - validate recaptcha if used
30         if(Content.Recaptcha ?? false) {
31             var recap = contactFormRequest["Recaptcha"];
32             if(!(recap is string) || String.IsNullOrEmpty(recap as string))
33                 throw new Exception("recaptcha is empty");
34
35         }
```



CMS Actions & Toolbars

Overview

- Action:
 - Name: "new", "edit", ...
 - Params:
 - id: 27
 - prefill={ link: "page:27" }
- Toolbar
 - Button Group
 - Button
 - will run Action
 - Settings
 - Visibility, float, ...








Profil






Über mich

Name	Hans Host	
Adresse	Musterstrasse 7 7070 Mustern	
Geburtsdatum	19.08.1980	
Telefon	000 333 30 30	
Email	hans.host@muster.ch	

Meine Fahrzeuge

Rufname	Modell	Gruppe	Gewicht [kg]	
Demo 1	Fendt 1000 Vario	Farmstock	14500	
Demo 2	MTD RF 125	Garden Standard	150	
Demo3	Fendt 800 VARIO	Farmstock	5120	
				

Meine Anmeldungen

Veranstaltung	Datum	Ort	Fahrzeug		
Test Veranstaltung 2	26.04.2016 - 27.04.2016	Goidegg	Demo3 Fendt 800 VARIO		
Test Veranstaltung 1	29.03.2016 - 30.03.2016	Hemberg	Demo 1 Fendt 1000 Vario		
					



STPV

de fr  2sichost

Info

Zum ersten mal hier? » [Wegleitung_STPV-Registrierungstool](#)

Pour la première fois ici? » [Instructions_pour_lenregistrement](#)



Actions and Params



- Can be called from any JS code
 - Like text-links
` + `
 - from any js-app like a manage-GUI in React
- Ca. 25 Actions
- Run-API (if user is logged in) using
`sxc.manage.run(verb);`
`sxc.manage.run(verb, params);`
- Documented in Wiki

Toolbars



- Toolbar Configuration with JSON
 - Simple like “add,edit”
 - Very detailed possible with a lot of json/js
 - Custom buttons and everything is possible
- Settings
 - float-behavior
 - show-behavior
 - ore-behavior (“...” button, right/left)

Toolbar Configuration JSON



- Config JSON is consistent
 - JS
 - C#
 - AngularJS & Angular
 - Please help with KO, React & Ember
- Multi-level configuration
 - Trivial – just button names
 - Sophisticated – full control of everything

Examples



Mobius: Admin-List filtered by current id

```
@Edit.Toolbar(toolbar: new object[] {  
    new { showCondition = true,  
        command = new { action = "contentitems",  
            contentType = config.ContentType,  
            filters = new { ModuleId = Dnn.Module.ModuleID }  
        }  
    }, settings: new { hover="left", show = "hover" })
```

@Edit.Toolbar only
appears if user has edit-
permissions

In JS you'll need to check
that in your code.

Example in Blog for new & manage

```
<div class="sc-element">  
    @* toolbar for add / manage posts *@  
    @Edit.Toolbar(toolbar: new object[] {  
        new {  
            command = new {  
                action = "new",  
                contentType = "BlogPost"  
            }  
        },  
        new {  
            command = new {  
                action = "contentitems",  
                contentType = "BlogPost"  
            },  
            showCondition = true  
        }  
    }, settings: new { hover="left", show = "hover" })  
  
    @* toolbar to manage list/view settings *@  
    @Edit.Toolbar(ListContent)
```



\$2sxc and sxc-instances

Background Know-How \$2sxc



- \$2sxc is a tiny js which is the foundation of everything
- Auto included when users are logged in and have edit-permissions
- Manually included for custom JS stuff
- Never redistribute yourself, as it adapts to changes in the 2sxc server
- Don't worry about duplicate inclusions

Background know how \$2sxc



- All actions are tied to a context
 - the context is a content-block
 - which is usually the mod. Instance
- Best way to get the context controller:
`var sxc = $2sxc([some-html-tag]);`
 - this will auto detect the context
- Alternatives
 - `var sxc = $2sxc(moduleId);`
 - `var sxc = $2sxc(moduleId, contentBlockId);`



Quick Recap

Recap Go Pro with 2sxc JS APIs



- 2sxc REST APIs
 - all basic operations
 - queries
- 2sxc JS helpers for
 - jQuery using
 - `sxc.WebApi.get...`
 - AngularJS by auto-configuring `$http`
 - ...more services
 - AngularJS framework libraries, especially React, Knockout, Ember, Vue
- \$2sxc & context
- Actions run CMS commands
 - With params like prefill, filter, etc.
- Toolbar is ultra-configurable in JSON
 - Custom buttons / icons grouping
 - Hover, align, more,...
 - Toolbar consistent across JS/C#/angular



Up Next: Angular & DNN

With more code 😊



Enjoy 2sxc and REST

Questions?