Writing high quality Mendix apps

and proving it with SAT!

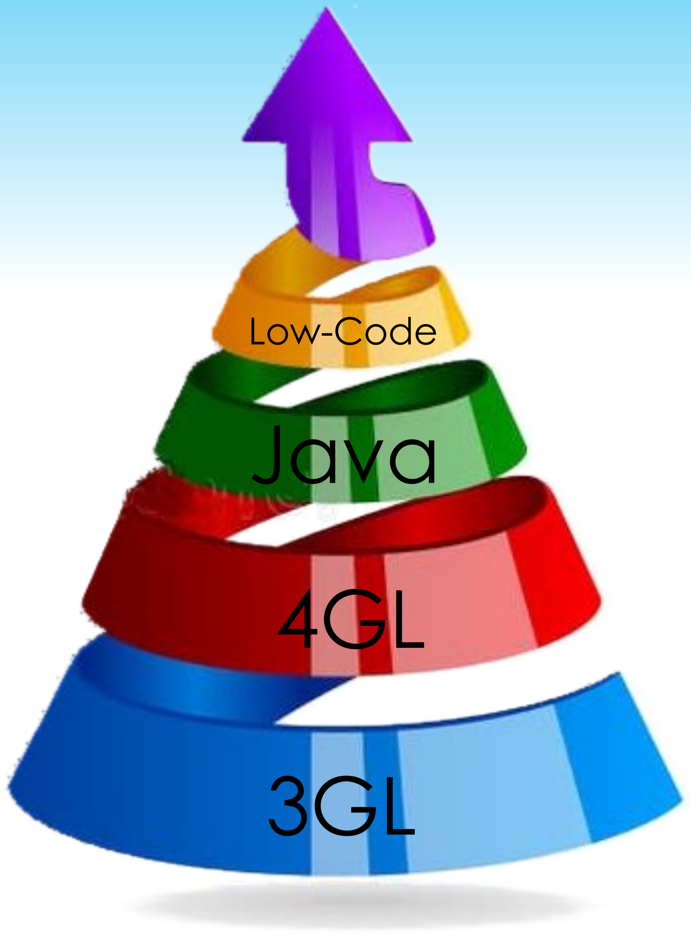
STONEXWORX

Contents

STONEXWORX

Introduction

- In IT since 1994
- 4 years of which in Mendix roles
- 1.5 year for Stoneworx
  - First months: no paid project +
  - Upskill trainee code assessments +
  - Wanted to learn about MX SDK's



Stoneworx Assessment Tool

- First take: Quality Assessment Tool
- Evolution:
  - Insight in quality issues
  - Security
  - Microflow structures

What is good quality code?

- Above all: Subjective as can be
- What you and your peers agree upon
  - Mendix community
  - Your company / the company you work for
  - Your project team

So we need a flexible set of quality controls that reflect the common ground of the project environment

Why another tool

- This one is free / open source
- it is adaptable
- It runs local: direct feedback
- It can scan both local mpr files or on team server
- It installs easy: docker or nodejs
- It will get better and better

Stoneworx Assessment Tool: Quality

Improve quality

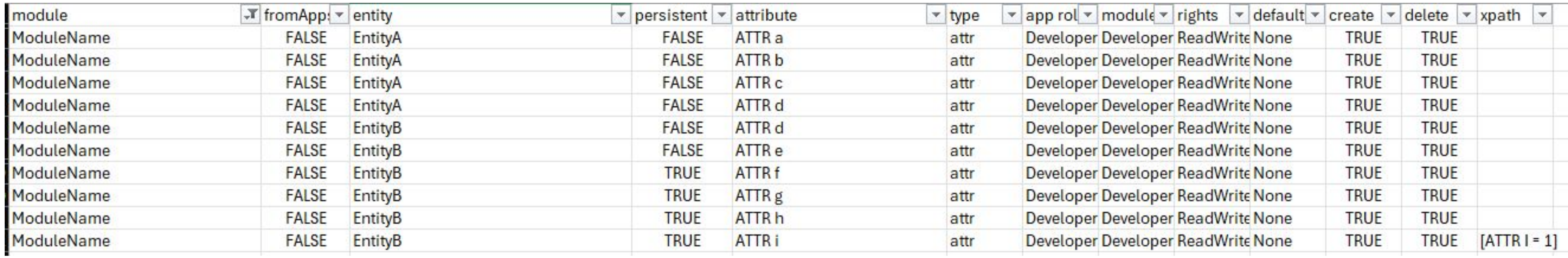
- Reliability
- Security
- Maintainability
- Project Hygiene

Hygiene	NC1	format: [PRE]_[Entity(s)]_description
Hygiene	NC2	Prefix must be allowed
Hygiene	NC3	Entity must exist
Hygiene	NC4	Entity must exist in same module
Maintainability	CM1	Commit not on correct hierarchy level(ACT or one level down)
Maintainability	IP1	Show Page action outside of ACT
Maintainability	IP2	Close Page action outside of ACT
Maintainability	ND1	Nesting of subs too deep
Maintainability	CX1	Too many actions in a single microflow
Maintainability	CX2	Too complex microflow
Maintainability	CX3	Too complex expression in Create/ Change Object
Maintainability	CX4	Too complex expression in Create / Change Variable
Reliability	EH1	Java Action without custom error handling
Security	DU1	Demo users not allowed in production app
Security	PM1	Microflow of this type should contain permissions

STONEXWORX

Stoneworx Assessment Tool: Security

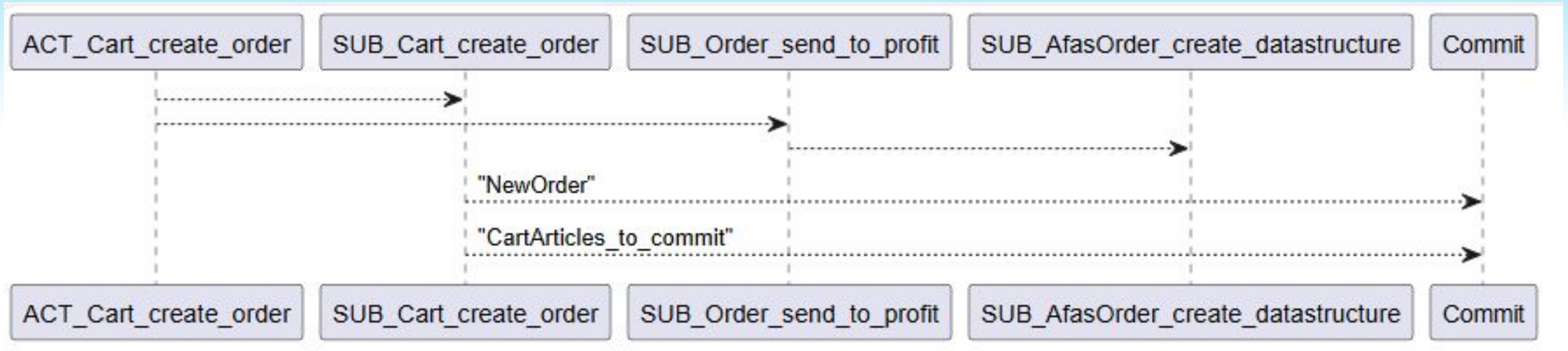
Export a csv with a access rules per Entity/Attribute. Create your own reports from there…



module	fromApp:	entity	persistent	attribute	type	app rol	module	rights	default	create	delete	xpath
ModuleName	FALSE	EntityA	FALSE	ATTR a	attr	Developer	Developer	ReadWrite	None	TRUE	TRUE	
ModuleName	FALSE	EntityA	FALSE	ATTR b	attr	Developer	Developer	ReadWrite	None	TRUE	TRUE	
ModuleName	FALSE	EntityA	FALSE	ATTR c	attr	Developer	Developer	ReadWrite	None	TRUE	TRUE	
ModuleName	FALSE	EntityA	FALSE	ATTR d	attr	Developer	Developer	ReadWrite	None	TRUE	TRUE	
ModuleName	FALSE	EntityB	FALSE	ATTR d	attr	Developer	Developer	ReadWrite	None	TRUE	TRUE	
ModuleName	FALSE	EntityB	FALSE	ATTR e	attr	Developer	Developer	ReadWrite	None	TRUE	TRUE	
ModuleName	FALSE	EntityB	TRUE	ATTR f	attr	Developer	Developer	ReadWrite	None	TRUE	TRUE	
ModuleName	FALSE	EntityB	TRUE	ATTR g	attr	Developer	Developer	ReadWrite	None	TRUE	TRUE	
ModuleName	FALSE	EntityB	TRUE	ATTR h	attr	Developer	Developer	ReadWrite	None	TRUE	TRUE	
ModuleName	FALSE	EntityB	TRUE	ATTR i	attr	Developer	Developer	ReadWrite	None	TRUE	TRUE	[ATTR I = 1]

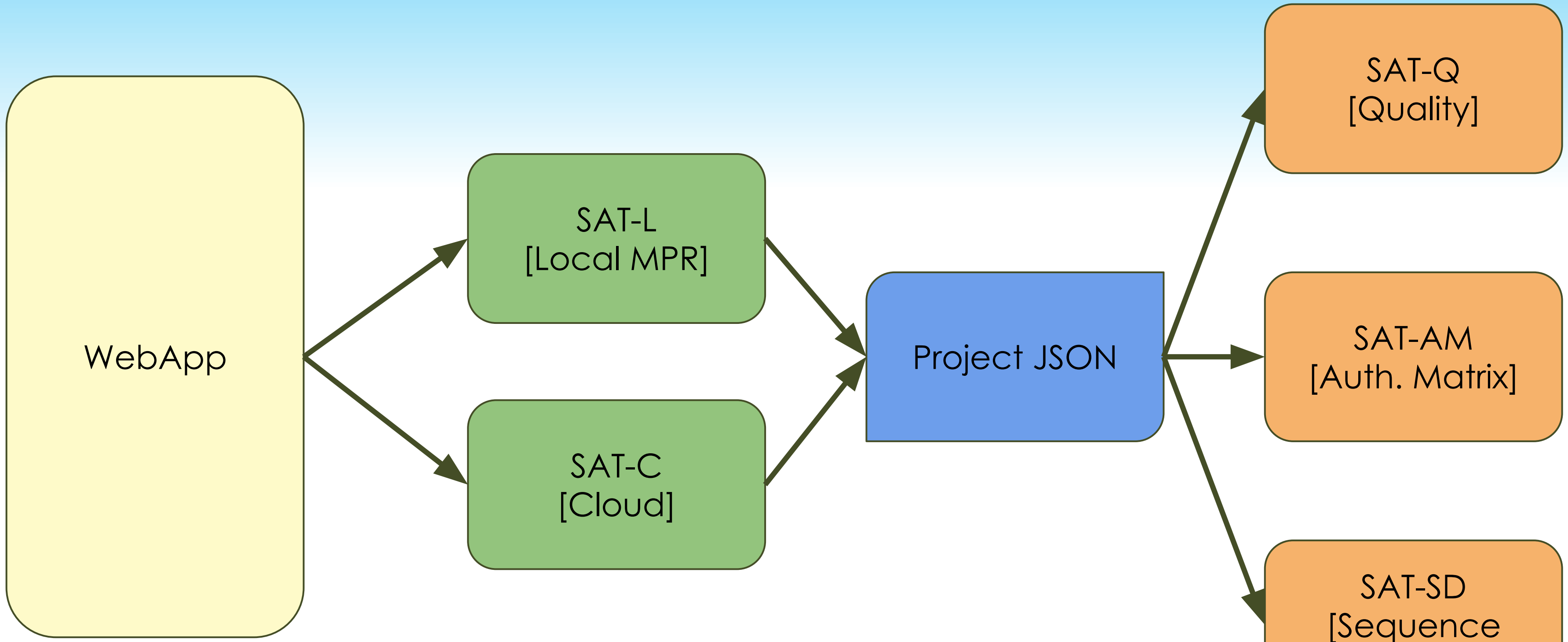
Stoneworx Assessment Tool: Microflows

- Generate a text file that can be rendered into a Sequence Diagram



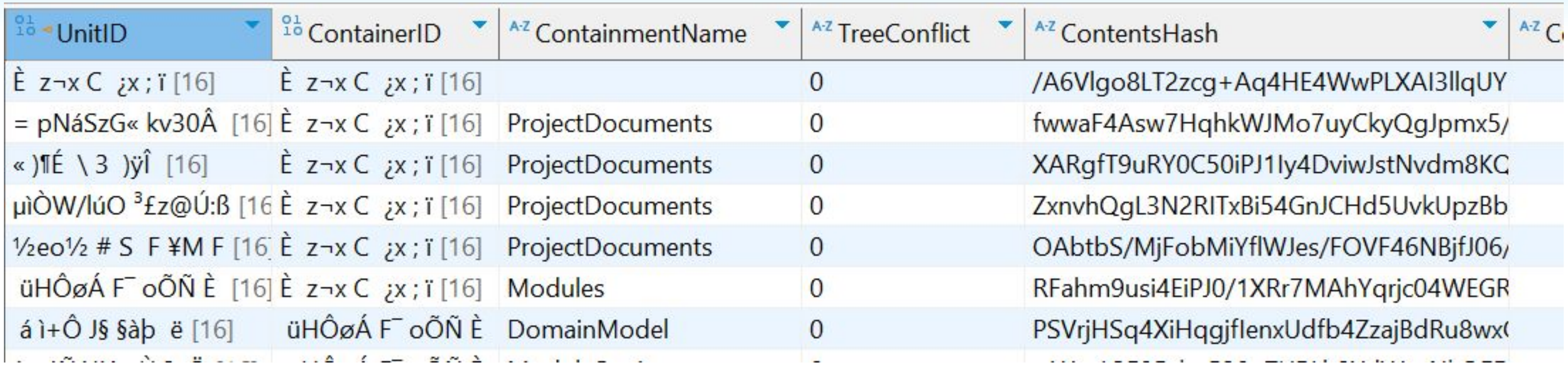
STONEXWORX

Quick overview



SAT-L

Parses the MPR which is actually a SQLite DB with a lot of bson info.



UnitID	ContainerID	ContainmentName	TreeConflict	ContentsHash	C
È z¬x C ¿x ; ï [16]	È z¬x C ¿x ; ï [16]		0	/A6Vlgo8LT2zcg+Aq4HE4WwPLXAI3llqUY	
= pNáSzG« kv30Â [16]	È z¬x C ¿x ; ï [16]	ProjectDocuments	0	fwwaF4Asw7HqhkWJMo7uyCkyQgJpmx5/	
«)¶É \ 3)ÿÎ [16]	È z¬x C ¿x ; ï [16]	ProjectDocuments	0	XARgfT9uRY0C50iPJ1Iy4DviwJstNvdm8KC	
µìÒW/lúO ³£z@Ú:ß [16	È z¬x C ¿x ; ï [16]	ProjectDocuments	0	ZxnvhQgL3N2RITxBi54GnJCHd5UvkUpzBb	
½eo½ # S F ¥M F [16	È z¬x C ¿x ; ï [16]	ProjectDocuments	0	OAbtbS/MjFobMiYflWJes/FOVF46NBjfJ06/	
üHÔøÁ F¯ oÕÑ È [16]	È z¬x C ¿x ; ï [16]	Modules	0	RFahm9usi4EiPJ0/1XRr7MAhYqrjc04WEGR	
á ì+Ô J§ §àþ ë [16]	üHÔøÁ F¯ oÕÑ È	DomainModel	0	PSVrjHSq4XiHqgjflenxUdfb4ZzajBdRu8wx	

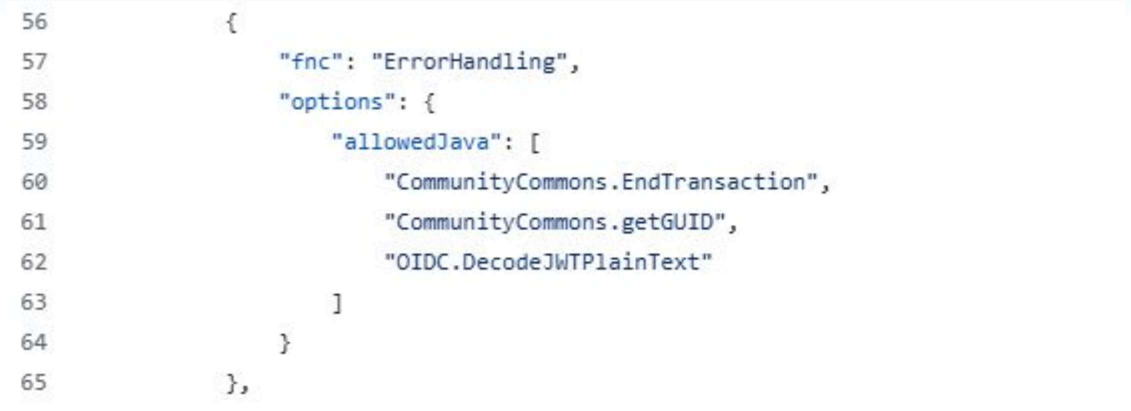
SAT-C

Connects to working copy in cloud with Mendix SDK’s.

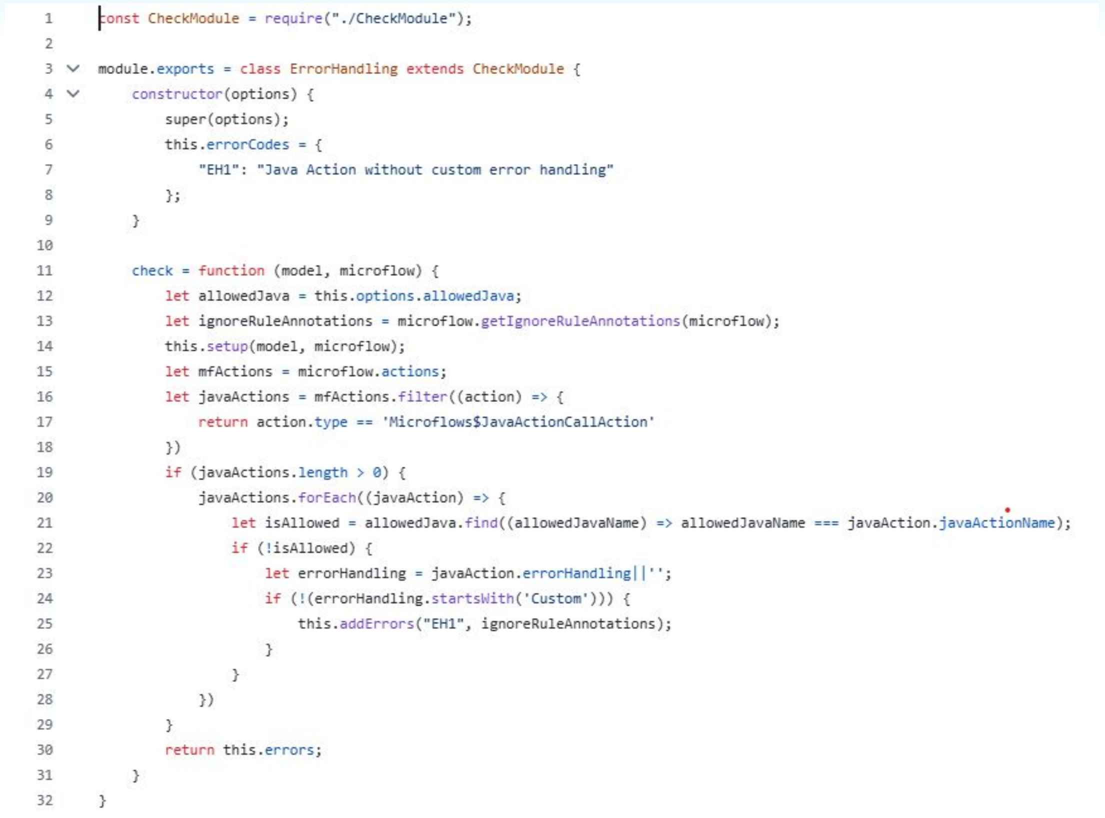
```
    workingCopyFile = branch + '.workingcopy'; //beetje ugly: global variabele keertje fixen
    return new Promise((resolve, reject) => {
        const client = new mendixplatformsdk_1.MendixPlatformClient();
        if (clear) {
            wcFile = "";
            wcID = "";
        } else readWorkingCopyFile(appID, workingCopyFile, branch);
        console.log(`GET APP: ${appID}-${branch}`);
        app = client.getApp(appID);
        console.log(`LOADING: ${appID}-${branch}`);
```

SAT-Q: Configuration & Extensions

- Config in a json file: default.json
- Modules in nodejs modules in SAT-Q:



```json
            {
                "fnc": "ErrorHandling",
                "options": {
                    "allowedJava": [
                        "CommunityCommons.EndTransaction",
                        "CommunityCommons.getGUID",
                        "OIDC.DecodeJWTPlainText"
                    ]
                }
            },
```



```javascript
const CheckModule = require("./CheckModule");

module.exports = class ErrorHandling extends CheckModule {
    constructor(options) {
        super(options);
        this.errorCodes = {
            "EH1": "Java Action without custom error handling"
        };
    }

    check = function (model, microflow) {
        let allowedJava = this.options.allowedJava;
        let ignoreRuleAnnotations = microflow.getIgnoreRuleAnnotations(microflow);
        this.setup(model, microflow);
        let mfActions = microflow.actions;
        let javaActions = mfActions.filter((action) => {
            return action.type == 'Microflows$JavaActionCallAction'
        })
        if (javaActions.length > 0) {
            javaActions.forEach((javaAction) => {
                let isAllowed = allowedJava.find((allowedJavaName) => allowedJavaName === javaAction.javaActionName);
                if (!isAllowed) {
                    let errorHandling = javaAction.errorHandling||'';
                    if (!(errorHandling.startsWith('Custom'))) {
                        this.addErrors("EH1", ignoreRuleAnnotations);
                    }
                }
            })
        }
        return this.errors;
    }
}
```

Demo!

Further development

Add more checks (get inspired by other commercial tools)

Add more diagrams (Class diagram)

Suggestions? □ Mail me ☺

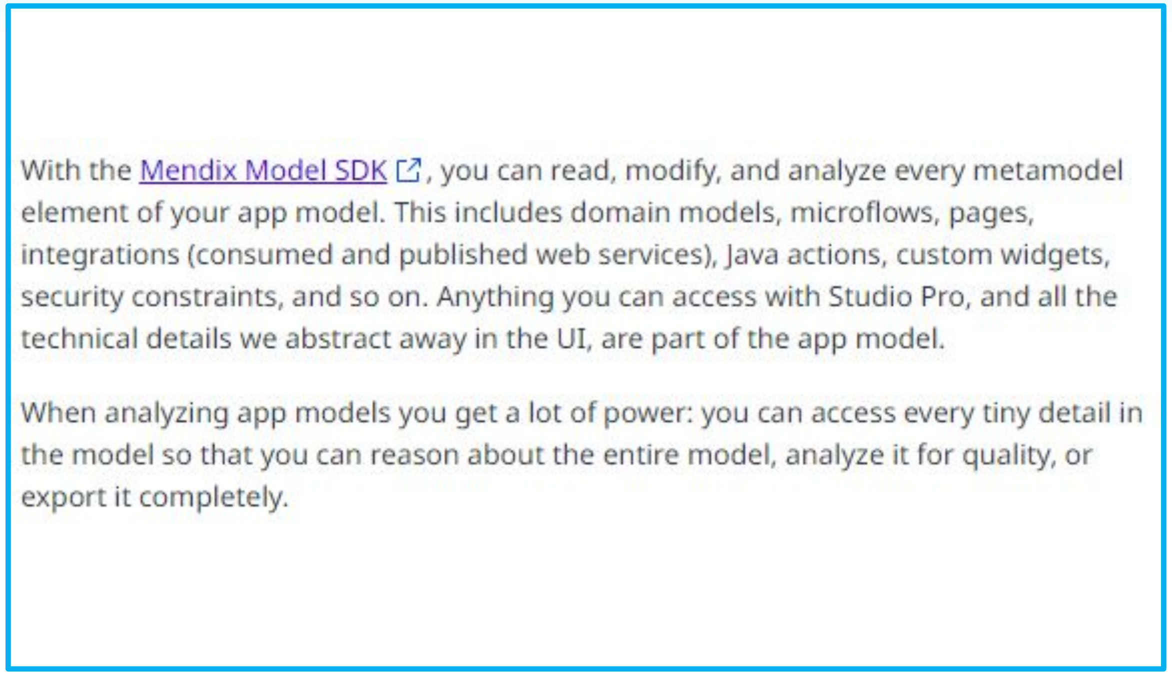
STONE WORX

Slide deck available in Github Repo:
https://github.com/StoneworxNL/SAT

Mendix Model & Platform SDK

Model SDK

Mendix model SDK documentation

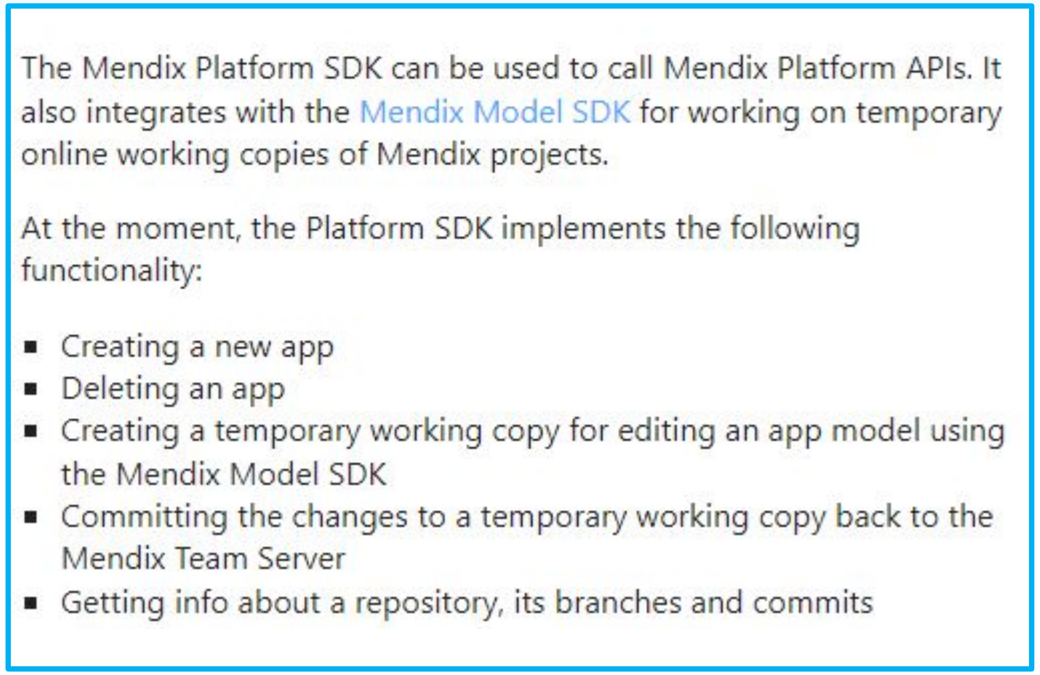


With the Mendix Model SDK, you can read, modify, and analyze every metamodel element of your app model. This includes domain models, microflows, pages, integrations (consumed and published web services), Java actions, custom widgets, security constraints, and so on. Anything you can access with Studio Pro, and all the technical details we abstract away in the UI, are part of the app model.

When analyzing app models you get a lot of power: you can access every tiny detail in the model so that you can reason about the entire model, analyze it for quality, or export it completely.

Platform SDK

Mendix platform SDK documentation



The Mendix Platform SDK can be used to call Mendix Platform APIs. It also integrates with the Mendix Model SDK for working on temporary online working copies of Mendix projects.

At the moment, the Platform SDK implements the following functionality:

- Creating a new app
- Deleting an app
- Creating a temporary working copy for editing an app model using the Mendix Model SDK
- Committing the changes to a temporary working copy back to the Mendix Team Server
- Getting info about a repository, its branches and commits

STONE WORX

Mendix Model & Platform SDK: howto

- Import the modules:

```
const mendixplatformsdk_1 = require("mendixplatformsdk");
const { microflows, Annotation} = require("mendixmodelsdk"); //, .. Depending on what you want to do
```

- Create a client:

```
const client = new mendixplatformsdk_1.MendixPlatformClient();
```

- Load an existing app:

```
app = client.getApp(appID);  // with the AppID that you can find in the portal under the App / Settings
```

STONEXWORX

Mendix Model & Platform SDK: howto

- Create working copy

```
app.createTemporaryWorkingCopy(branch)
      .then((workingCopy) => {
      // return or work with new workingcopy here
            })
      .catch((err) => {console.log(err)});
```

- Load a working copy:

```
 workingCopy = app.getOnlineWorkingCopy(wcID); // no promises here
```

- Open the model:

```
workingCopy.openModel()
      .then((model) => {
            // return or work with model here
            })
      .catch((err) => {console.log(err)});
```

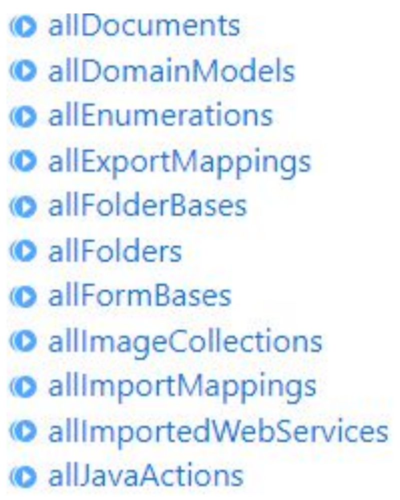
STONEXWORX

Mendix Model & Platform SDK: howto

- Work with the model

```
let securities = model.allProjectSecurities(); // load project security settings
Let microflows = model.allMicroflows();         //load all microflows
```

- You get the idea:


allDocuments
allDomainModels
allEnumerations
allExportMappings
allFolderBases
allFolders
allFormBases
allImageCollections
allImportMappings
allImportedWebServices
allJavaActions

Mendix Model & Platform SDK: howto

- The allXXXX methods return an array of Interfaces, to work with the real thing, you have to load it first:

```
microflowIF.load()
      .then((microflow) => {
            // work with microflow
       })
            .catch((err)=>{console.log(err)})
```

- Works for other objects as well

- For detailed information on the API: check documentation mentioned above

SAT / Installing & GIT

Nodejs & NPM: Required to run the tool

- Javascript without a browser
- Download here (including NPM, node package manager)
- Test from CMD with: \
  - node –v give something like “v20.10.0”
  - npm –v gives something like “10.4.0”

SAT Code

- Download the tool from GIT here
- via green button download zip and unzip to working directory

Run “npm install”

For Sequence Diagrams:

- Download PlantUML here
- Or use online tool

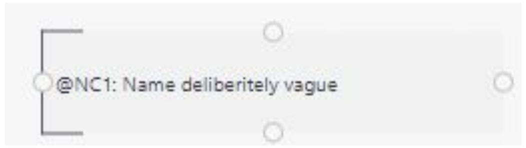
Set Personal Access Token in envrironment

- Access User Settings / Developer settings via top right menu in Portal
- Create a PAT for the app with at least:
  - mx:modelrepository:repo:read
  - mx:app:metadata:read
- Create environment variable with cmd line: set MENDIX_TOKEN=[YOUR PAT HERE]

SAT / Microflow Quality Result (snippet)

```
Microflow;Code;Description
security;DU1;Demo users not allowed in production app
CarInsurance.SUB_TEMPSellers_AddIndex;NC3;entity must exist
BackOffice.ACT_Plan_SaveVAL;CX2;Too complex microflow
CarInsurance.DS_TempSellers_Pos1;NC3;entity must exist
```

NB: If you want to accept a rule 'violation': just add an annotation in your microflow like:



@NC1: Name deliberitely vague

SAT / Microflow Quality Configuration

- default.json: file describing all checks to execute
- Format:
  - checksFolder: location of pluggable check modules
  - checks
    - fnc: a CheckModule implementation
    - options: specific options per module

```
{
    "checksFolder": "../qualityChecks/",
    "checks": [{
            "fnc": "NamingConvention",
            "options": {
                "allowedPrefixes": ["ACO","ACT","SUB","CRS","SCH","CTL",….]}
        },
        {
            "fnc": "ErrorHandling"
        },
```

STONEXWORX

SAT / Microflow Quality Extending

- Add modules to the Check modules folder, include them in the default.json
- Should look like this:

```
const CheckModule = require("./CheckModule");

module.exports = class ErrorHandling extends CheckModule {
    constructor(options) {
        super(options);
        this.errorCodes = {
            "EH1": "Java Action without custom error handling"
        };
    }

    check = function (mfQuality, microflow) {
        let errors = [];
        let mfActions = mfQuality.hierarchy[microflow].actions;
        CHECK LOGIC HERE
        return errors;
    }
}
```

STONEXWORX

SAT / Complementary Tooling

- Bizzomate Mendix Dev Tools (documentation)
  - Gives insight in security
  - Chrome extention that gives access to backend (bypassing mendix frontend)
  - Thus revealing all access rules for each role
- Menditect Microflow Call Hierarchy (documentation)
  - Gives insight in call hierarchy of nested microflows
  - Marketplace widget
  - Requires MX 10.6 and up