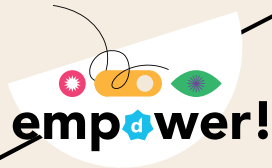


# Welc<sup>d</sup>me



Getting *ll*



-----> started

With DOM

[ YOUR TRAINER THIS SESSION ]

Principal  
DevOps  
Engineer



**Thomas**  
Ghysbrecht

thomas.ghysbrecht@skyline.be

# Agenda

Fundamentals track

## 1. Introduction

1. What is DataMiner Object Models (DOM)?
2. Options to create a model
3. Options to create instances

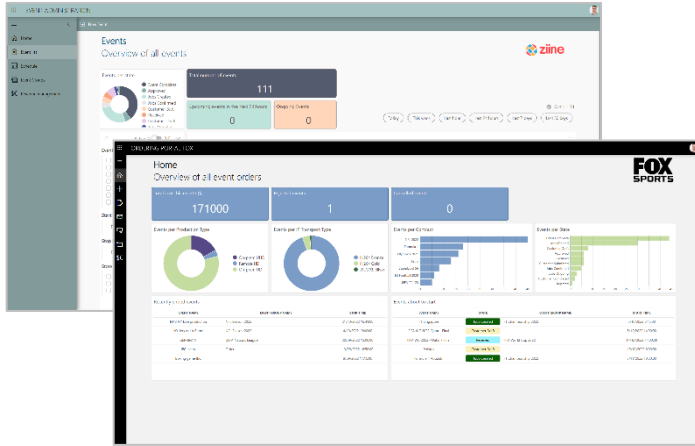
## 2. Hands-on

## 3. Best Practices

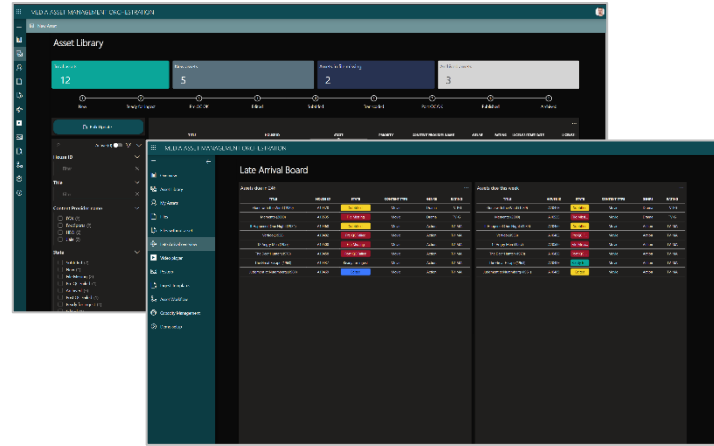


# DataMiner Object Models (DOM)

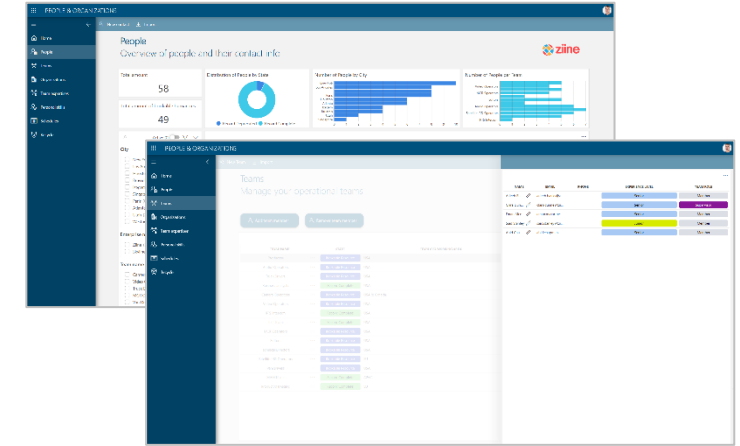
Enrich your operation with just any administrative data



- Event administration
- Event customer portals
- Vehicle resource management



- Asset management
- RPD provisioning
- Planned maintenance
- Ticketing



- People and organizations
- Human resource scheduling
- Virtual desk management

# Introduction

## Goal - Our model - Vehicle

License Plate ⓘ

ABC-123

Location ⓘ

USA <

Capabilities ⓘ

HD and 1 other option <

Size ⓘ

Medium <

Number Of Cameras ⓘ

12



# Introduction

## The 4 core DOM objects

Module Settings

1

Section Definitions  
(with FieldDescriptors)

2

DOM Definitions

3

DOM Instances

4

# Introduction

## The 4 core DOM objects

Module Settings

1

Section Definitions  
(with FieldDescriptors)

2

DOM Definitions

3

DOM Instances

4



# Introduction

## 1. Module Settings

- **Defines cross-model settings and forms a logical grouping of DOM models.**
  - Setting types:
    - Script settings (CRUD scripts)
    - Time-to-live of data (TTL)
    - History settings
    - Storage settings
    - ...
- Advanced use, defaults are fine in most cases

### Module Settings

ID:	my_vehicle_manager
ScriptSettings	...
TtlSettings	...
...	

# Introduction

## 1. Module Settings

- Module ID:
  - Unique in the DMS
  - Between 1 and 40 characters
  - Lowercase and not contain any special characters (see [docs](#))
- Example: **my\_vehicle\_manager**

### Module Settings

ID:	my_vehicle_manager
ScriptSettings	...
TtlSettings	...
...	

# Introduction

## The 4 core DOM objects

Module Settings

1

Section Definitions  
(with FieldDescriptors)

2

DOM Definitions

3

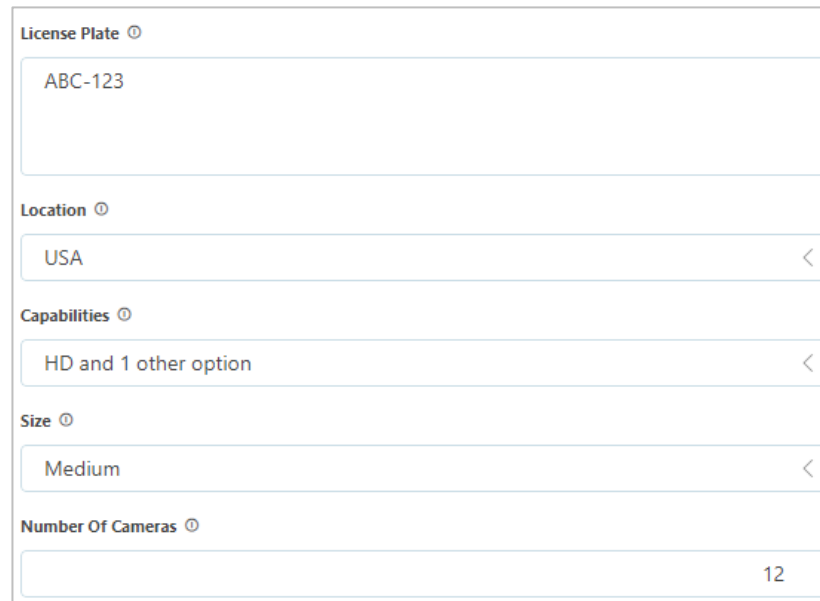
DOM Instances

4

# Introduction

## 2. Section Definition

- **Defines a section of your DOM model.**
- A section is a group of fields that belong together.
- Fields are described by **Field Descriptors**.
- These Section Definitions can be reused between models.



License Plate ⓘ  
ABC-123

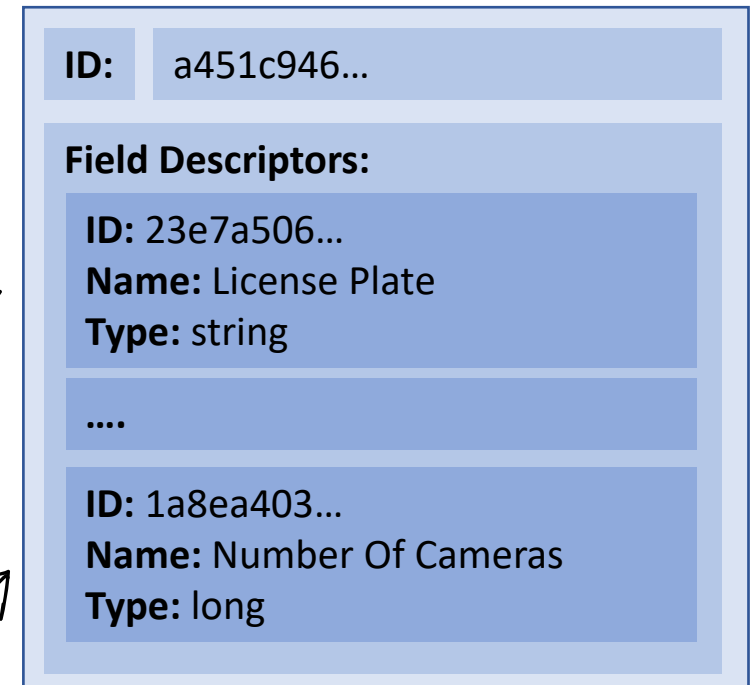
Location ⓘ  
USA <

Capabilities ⓘ  
HD and 1 other option <

Size ⓘ  
Medium <

Number Of Cameras ⓘ  
12

### Section Definition



ID: a451c946...

**Field Descriptors:**

ID: 23e7a506...  
**Name:** License Plate  
**Type:** string

...

ID: 1a8ea403...  
**Name:** Number Of Cameras  
**Type:** long

# Introduction

## 2. Section Definition - Field Descriptor

### • Basic:

- String
- Double
- Long
- DateTime
- TimeSpan
- Bool

### Special:

- Auto Increment
- Enum
- SRM Booking
- SRM Resource
- SRM Service Definition
- DOM Instance
- DOM Instance Value
- DataMiner Element
- DataMiner Group
- DataMiner User

### Section Definition

The diagram illustrates a 'Section Definition' structure. It consists of a main container with a light blue background. Inside, there is a top section with a light blue background containing the text 'ID: a451c946...'. Below this is a 'Field Descriptors' section, also with a light blue background. This section contains two entries, each with a light blue background. The first entry has the text 'ID: 23e7a506...', 'Name: License Plate', and 'Type: string'. The second entry has the text 'ID: 1a8ea403...', 'Name: Number Of Cameras', and 'Type: long'. Ellipses '...' are placed between the two entries in the 'Field Descriptors' section. The 'Type' field for each entry is circled in black.

```
graph TD
    subgraph Section_Definition [Section Definition]
        ID1[ID: a451c946...]
        subgraph Field_Descriptors [Field Descriptors]
            ID2[ID: 23e7a506...]
            Name1[Name: License Plate]
            Type1[Type: string]
            Dots[....]
            ID3[ID: 1a8ea403...]
            Name2[Name: Number Of Cameras]
            Type2[Type: long]
        end
    end
```

# Introduction

## The 4 core DOM objects

Module Settings

1

Section Definitions  
(with FieldDescriptors)

2

DOM Definitions

3

DOM Instances

4

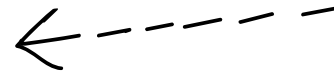
# Introduction

## 3. DOM Definition

- Defines a certain DOM model by **linking together the relevant Section Definitions**.
- Can also define overrides of the Module Settings if these would differ for a certain model.

### SectionDefinition

ID:	a451c946...
Field Descriptors:	...



### DOM Definition

ID:	eb7c180f...
Section Links:	
	ID: a451c946...
	...

# Introduction

## The 4 core DOM objects

Module Settings

1

Section Definitions  
(with FieldDescriptors)

2

DOM Definitions

3

DOM Instances

4

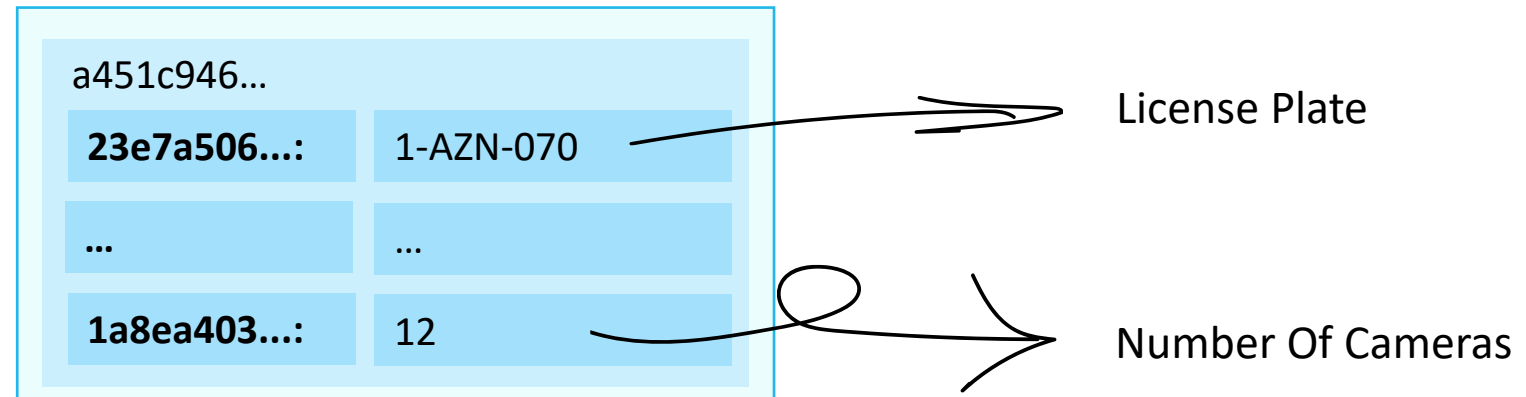


# Introduction

## 4. DOM Instance

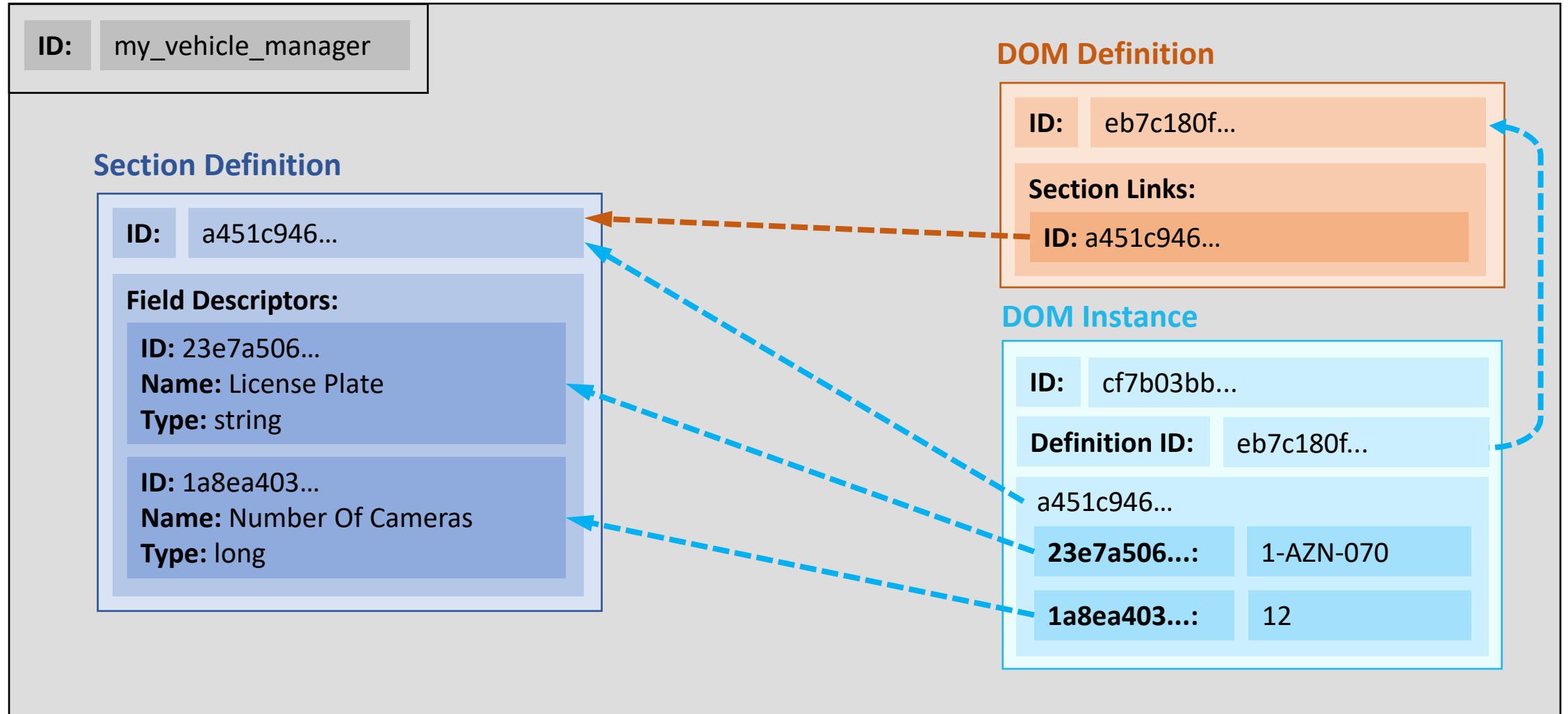
- **Represents the actual objects that adheres to the DOM model.** In our case 'Vehicles'.
- Contains a **Section** for each Section Definition.
- Sections contain a **Field Value** for each Field Descriptor.
- There could be millions of DOM instances in a DOM module.

### DOM Instance



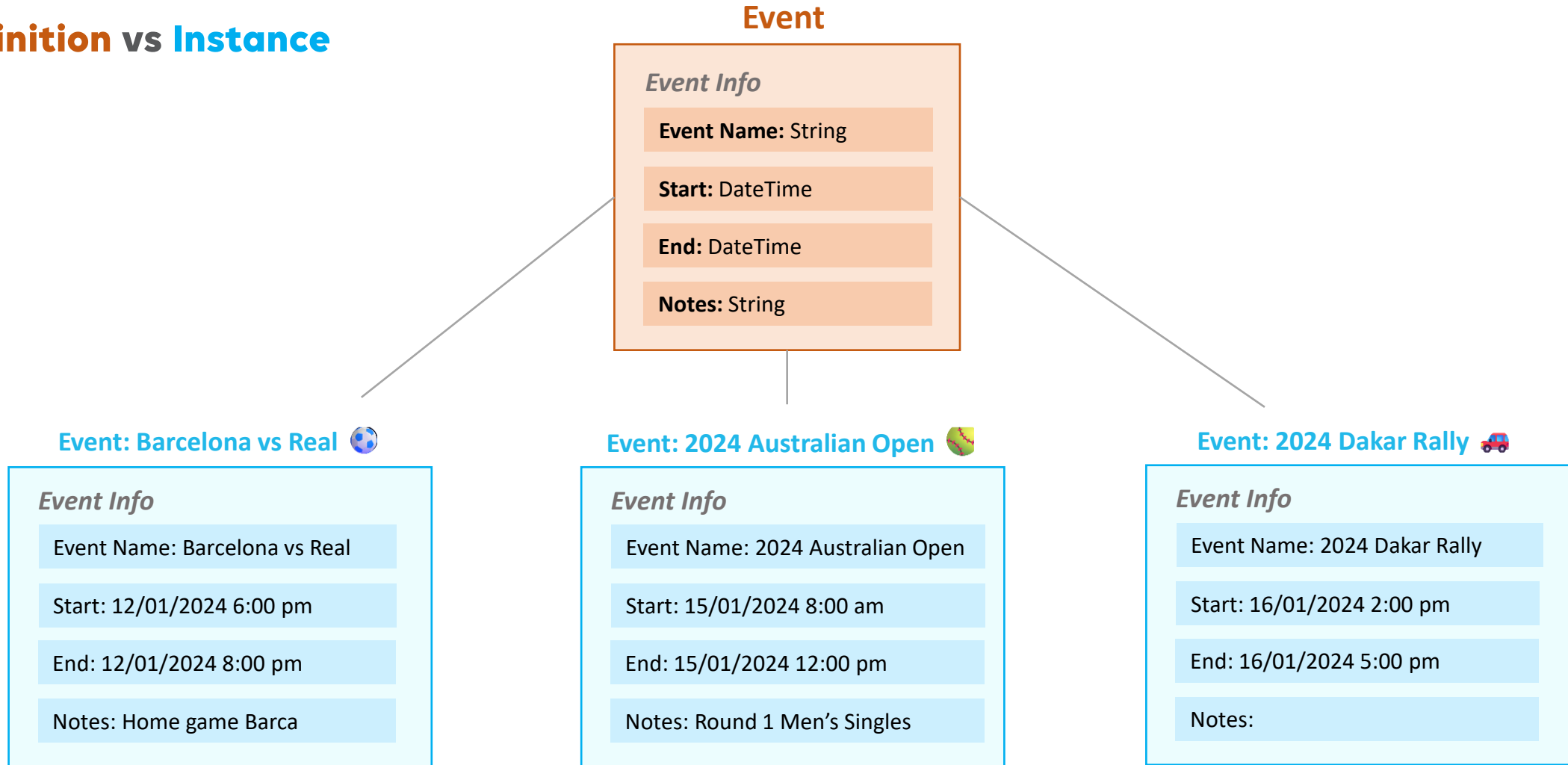
# Introduction

## Module Settings



# Introduction

## Definition vs Instance



# Introduction

## Creating the model

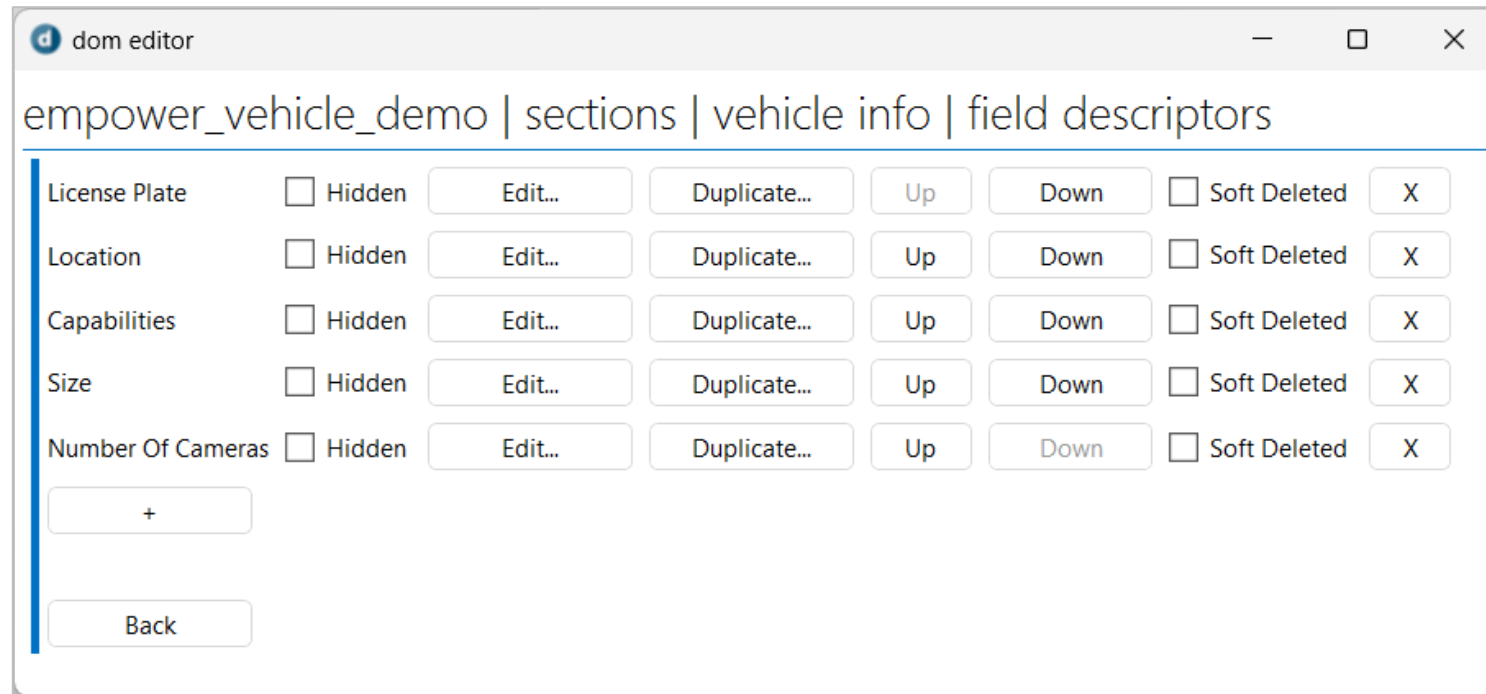
- **Option 1:** C# Code ← ---
- **Option 2:** DOM Editor script
- **Option 3:** DOM Designer script + Excel

```
// Create DomDefinition
_domDefinition = new DomDefinition
{
    SectionDefinitionLinks = new List<SectionDefinitionLink>()
    {
        new SectionDefinitionLink(_sectionDefinition.GetID())
    }
};
_domDefinition = _domHelper.DomDefinitions.Create(_domDefinition);
```

# Introduction

## Creating the model

- **Option 1:** C# Code
- **Option 2:** DOM Editor script ← ---
- **Option 3:** DOM Designer script + Excel



# Introduction

## Creating the model

- **Option 1:** C# Code
- **Option 2:** DOM Editor script
- **Option 3:** DOM Designer script + Excel ← ---

	A	B	C	D	E
1	SectionName	Name	Type	Default	Values
2	Order details	Order id	String		
3	Order details	Description	String		
4	Order details	Start time	DateTime		
5	Order details	End time	DateTime		
6	Order details	Server location	Enum	Europe	Europe/US
7					

◀ ▶

fields

form\_New

states

buttons

events

+

# Introduction

## Creating the instances

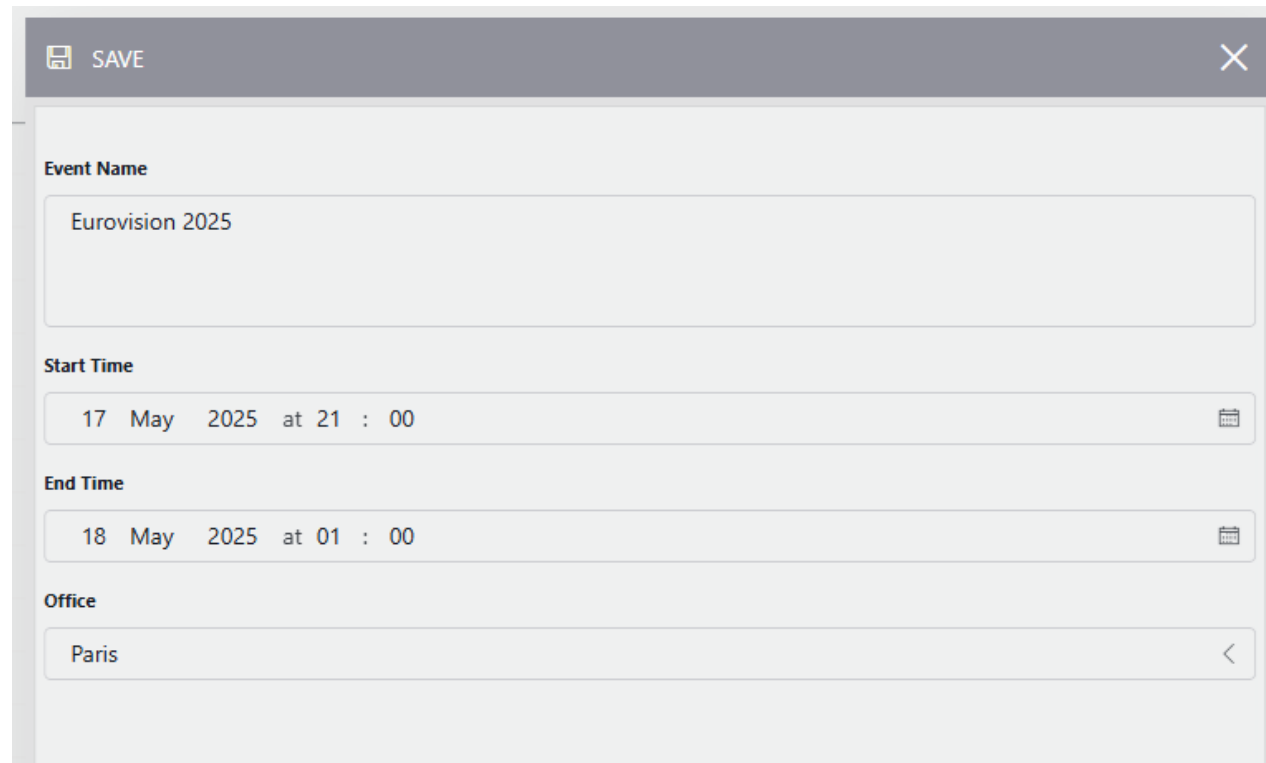
- **Option 1:** C# Code ← ---
- **Option 2:** Low-code App Form

```
// John was grabbing a coffee in the kitchen when he noticed that one of the lights is not working
var domInstance = new DomInstance()
{
    ID = new DomInstanceId(Guid.NewGuid()),
    DomDefinitionId = domDefinition.ID,
};
domInstance.AddOrUpdateFieldValue(sectionDefinition, reporterNameFieldDescriptor, "John");
var description = "One of the lights in the kitchen on the 2nd floor does not work";
domInstance.AddOrUpdateFieldValue(sectionDefinition, issueTextFieldDescriptor, description);
domInstance = _domHelper.DomInstances.Create(domInstance);
```

# Introduction

## Creating the instances

- **Option 1:** C# Code
- **Option 2:** Low-code App Form ← ---



A screenshot of a low-code application form titled "SAVE" with a close button (X) in the top right corner. The form contains four input fields:

- Event Name:** A text field containing "Eurovision 2025".
- Start Time:** A date and time picker showing "17 May 2025 at 21 : 00" with a calendar icon on the right.
- End Time:** A date and time picker showing "18 May 2025 at 01 : 00" with a calendar icon on the right.
- Office:** A dropdown menu showing "Paris" with a left arrow icon on the right.



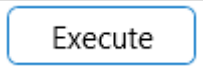


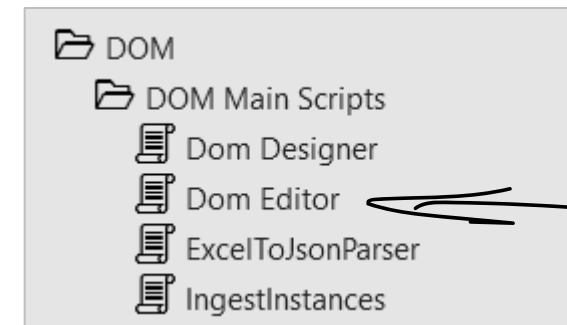
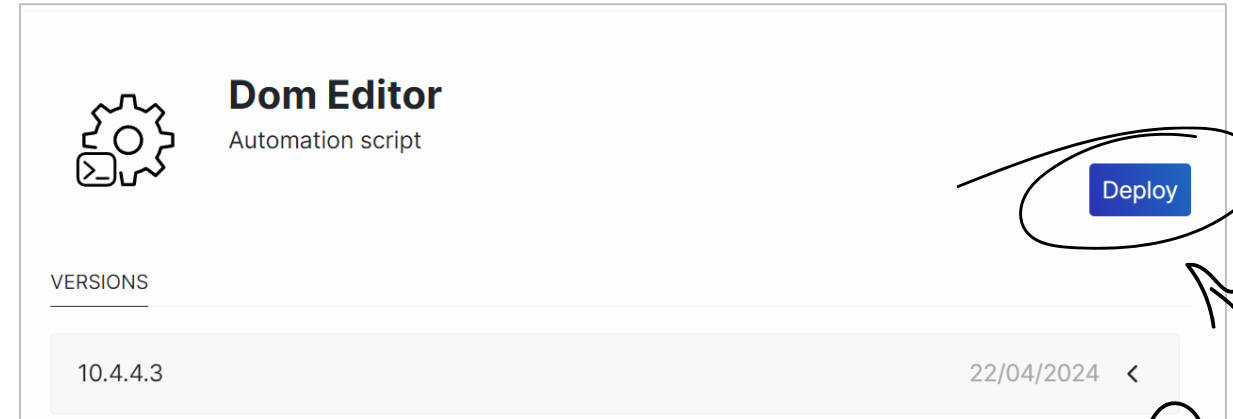
# Hands-on

## Let's create a DOM model!

# Hands-on

## Creating our DOM model using the DOM editor

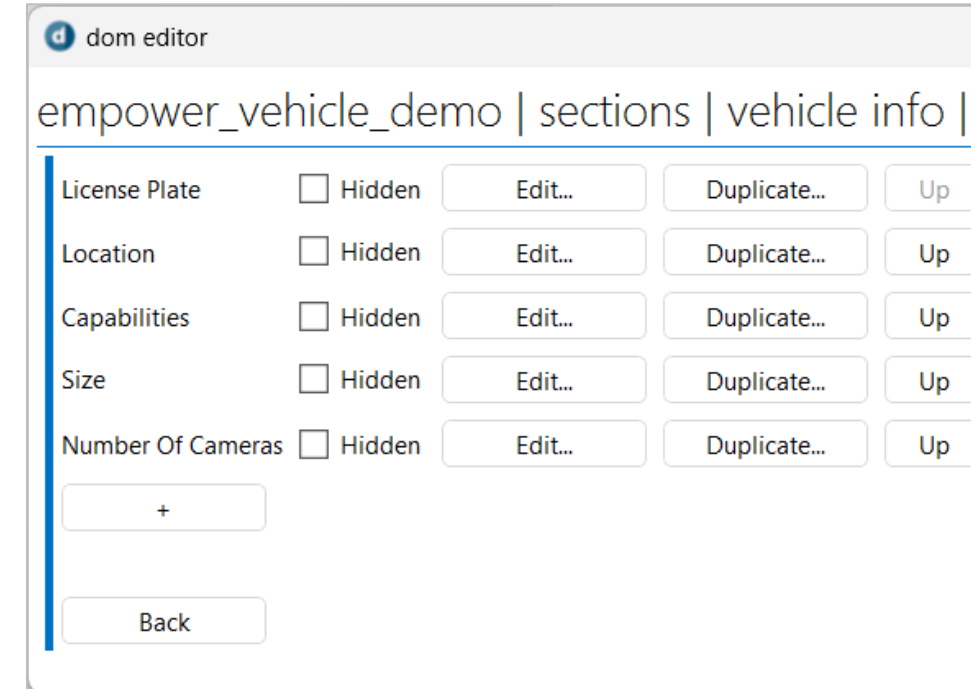
1. Go to <https://catalog.dataminer.services/>
2. Search for '**Dom Editor**'
3. Deploy this package to your DaaS agent
4. Open up Cube 
5. Go to the **Automation module** 
6. Execute the '**Dom Editor**' script 



# Hands-on

## Creating our DOM model using the DOM editor

- Create a DOM model for a vehicle with fields:
  1. **License Plate** – Text like (e.g. '1-AZN-070')
  2. **Location** – List of values (e.g. 'UK', 'Belgium', etc.)
  3. **Size** – List of values (e.g. 'Small', 'Medium', 'Large', etc.)
  4. **Number of Cameras** – Number (e.g. '12')



The screenshot shows the 'dom editor' window with the title 'empower\_vehicle\_demo | sections | vehicle info |'. It displays a list of fields for a vehicle model:

Field Name	Hidden	Edit...	Duplicate...	Up
License Plate	<input type="checkbox"/>	Edit...	Duplicate...	Up
Location	<input type="checkbox"/>	Edit...	Duplicate...	Up
Capabilities	<input type="checkbox"/>	Edit...	Duplicate...	Up
Size	<input type="checkbox"/>	Edit...	Duplicate...	Up
Number Of Cameras	<input type="checkbox"/>	Edit...	Duplicate...	Up

Below the list, there is a '+' button to add new fields and a 'Back' button.

# Hands-on

## Creating our DOM model using the DOM editor

- Send in a screenshot of the form in a LCA and earn **25 DevOps points!**

[aka.dataminer.services/  
dom-empower](https://aka.dataminer.services/dom-empower)

OR

support.automation-  
orchestration@skyline.be  
Subject: "Empower 2025"

A screenshot of a web form titled "Form 1" enclosed in a dashed border. The form contains four sections: "License Plate" with a text input field containing "AZN-070"; "Location" with a dropdown menu showing "Belgium"; "Size" with a dropdown menu showing "Medium"; and "Number Of Cameras" with a text input field containing "16".

License Plate

AZN-070

Location

Belgium

Size

Medium

Number Of Cameras

16

Form 1

# Best practices

## Things to keep in mind when building models

# Best Practices

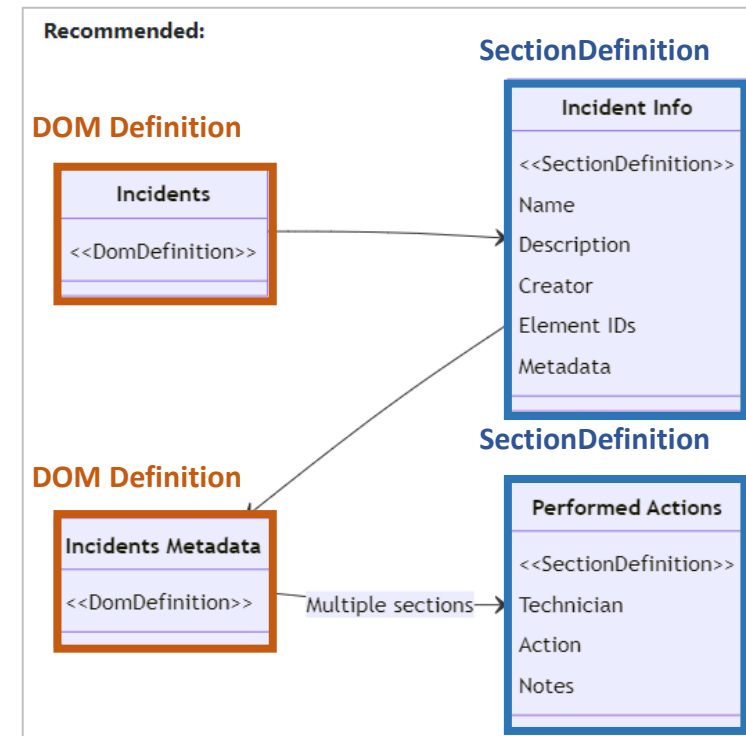
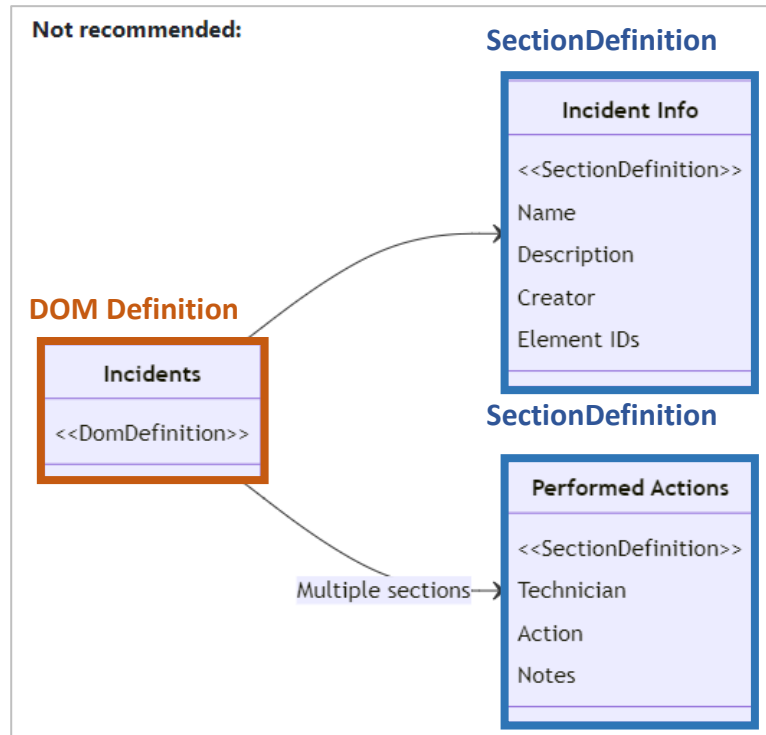
## Use compact DOM instances

- The **smaller** the DOM instance, the **faster** it can be processed
- Tips:
  1. **Only include data that is actually required:** It is easier to add a field later, than to remove one
  2. **Avoid data duplication:** Only store the same data once
  3. **Avoid storing JSON in a string field:** Use multiple sections or other DOM instances instead
  4. **Store rarely used (meta)data in a separate DOM instance:** Prevent retrieving data that you do not need

# Best Practices

## Use compact DOM instances

**Store rarely used (meta)data in a separate DOM instance:** Prevent retrieving data that you do not need.



# Best Practices

## Optimize CRUD calls

- Every call to DataMiner and the database has a certain overhead. By reducing the number of calls, you can limit this overhead.
- Tips:
  - Use **bulk** create, update, or delete whenever possible
  - Avoid doing multiple updates to the same DOM instance in the same script/code
  - Read all relevant DOM data in **one call**

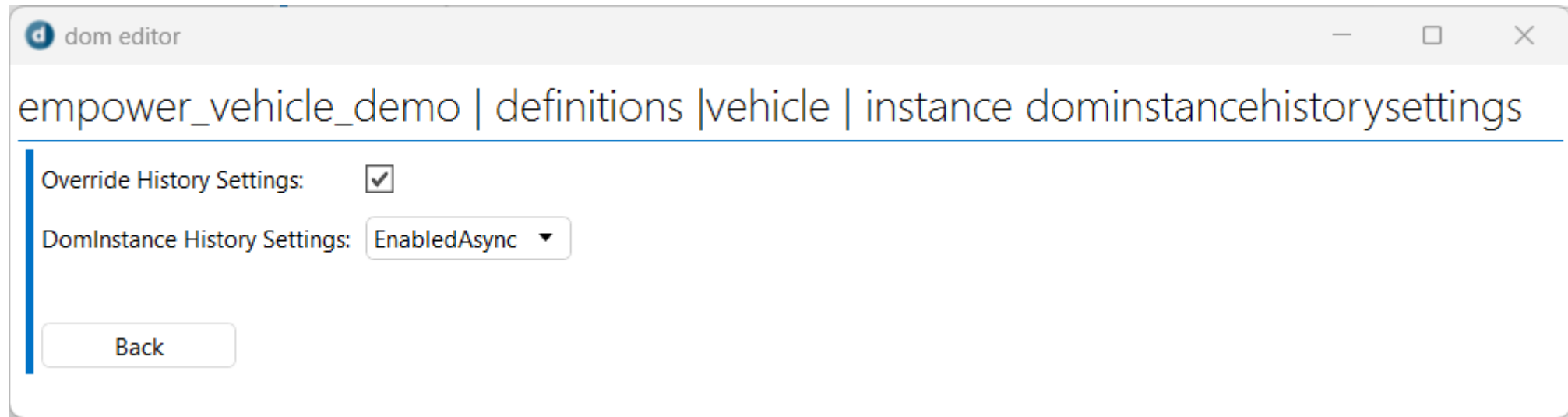
```
private List<DomInstance> GetDomInstances(DomHelper helper, List<DomInstanceId> ids)
{
    return Tools.RetrieveBigOrFilter(
        ids,
        id => DomInstanceExposers.Id.Equal(id),
        filter => helper.DomInstances.Read(filter));
}
```



# Best Practices

## Disable instance history

- By default, **any change** made to a DOM instance **is tracked** in a history record.
- Disabling the history **reduces the load on the system and database**.
- Especially important for DOM manager that see **frequent DOM instance updates**.
- Can be disabled on **module or DOM definition level**.



dom editor

empower\_vehicle\_demo | definitions | vehicle | instance dominstancehistorysettings

Override History Settings: ☒

DomInstance History Settings: EnabledAsync ▼

Back

# DOM in the DataMiner Docs

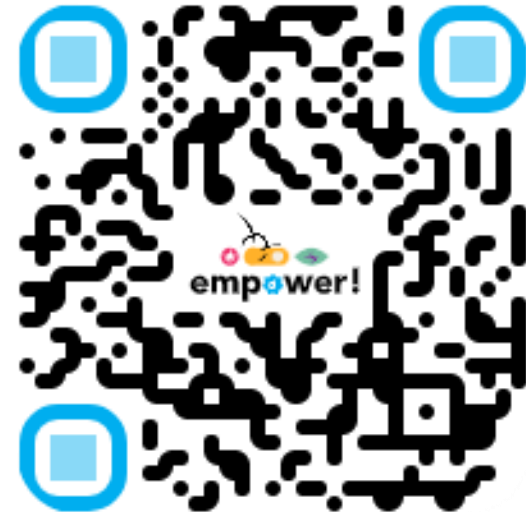
**Learn more about DOM!**

# DataMiner Docs

Expand your DOM knowledge with the DataMiner Docs

- The DataMiner Docs contains nearly everything there is to learn on DOM.
  - Getting started with DOM
  - Code examples
  - Tutorials
  - Additional features (e.g. **status system**)

<https://aka.dataminer.services/DOM>



# Additional exercise

## Earn extra DevOps points

- Let your imagination run wild! Earn **50 DevOps points** by emailing a screenshot of the LCA form of a new DOM model that would enrich the vehicle use-case.
- Examples:
  - Insurance Tracking
  - Maintenance History
  - Vehicle Inventory Manager
- Deadline: Tuesday 19h CET

[aka.dataminer.services/  
dom-empower](https://aka.dataminer.services/dom-empower)

OR

support.automation-  
orchestration@skyline.be  
Subject: "Empower 2025"

[ ASK AWAY ]



# Questions?