From Simcad to Digital Twin Studio

Migrating Simulation Models to a Live Digital Twin Environment

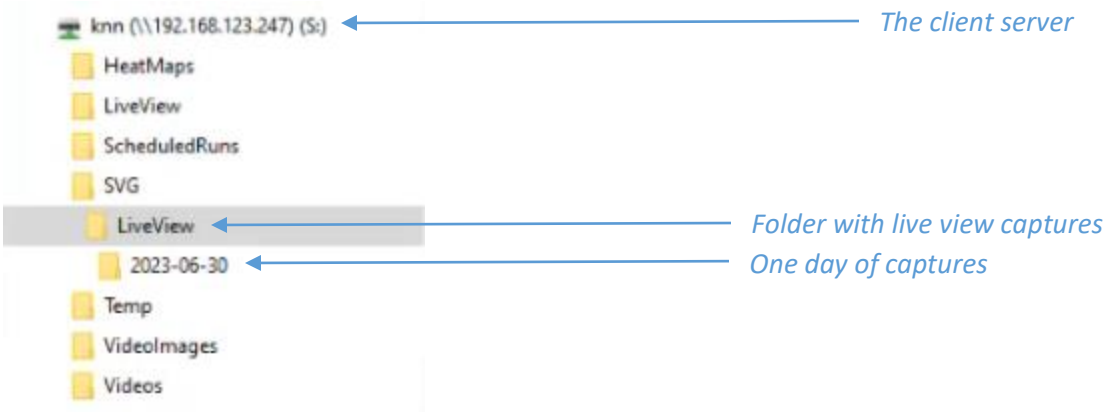
Introduction

With a few simple modifications, a Simcad model can be transformed into a live Digital Twin which outputs to a customer-accessible web interface. This guide will outline the components of the Digital Twin system and outline how to turn a Simcad model into a live Digital Twin.

The System

Simcad models first need connection to a client server. Data from the Simcad model will be exported to a drive on this server. The drive must have sufficient memory to support continual data export. Normally, this will not be the (C:) drive.

The model will create a folder on the drive for each day of tracking. Each folder will be titled in *yyyy-mm-dd* format. All model data for a particular day will be exported to the corresponding folder.



Clients log in to the DT Studio web interface to view the live Digital Twin. The web interface connects to and pulls data from the same drive which stores Simcad model data.

Now that we understand the structure of the Digital Twin system, we can detail exactly how to modify a Simcad model for live data output.

Capturing Model Data

There are two primary methods by which Simcad models can retrieve and export data:

- Via **Model Extensions**
- Via a **Department Window**

Both methods function by triggering an event which will capture data and other model information, and export the data to the drive on the client server.

Model Extensions

In Simcad, **Model Extensions** can be used to trigger data capture.

There are three different extensions which work together to capture model information and send it to the DT web interface. The three extensions will:

- Create a heat map and spaghetti diagram of the model
- Capture an image of the static model background
- Capture an image of the objects moving throughout the model

Extension One: Heat Map & Spaghetti Diagram

The heat map and spaghetti diagram extension creates an SVG file with the background layout, object positions, and anything that appears in the popup interface.

The following function is used to generate a heat map and spaghetti diagram:

```
DisplayHeatMapAndSpaghettiDiagram(ObjectList as String, DisplayHM,
DisplayPath, DisplayCongestion, AutoExpand)
```

This function is exactly the same as performing the analysis through the Simcad interface. The parameters of the function are as follows:

`DisplayHeatMapAndSpaghettiDiagram`

- Command issued to capture and display heat map and spaghetti diagram

`ObjectList as String`

- Defines the object parameter for the heat map and spaghetti diagram.

`DisplayHM`

- If set to `1`, heat map will be displayed. If set to `0`, not.

`DisplayPath`

- If set to `1`, spaghetti diagram will be displayed. If set to `0`, not.

`DisplayCongestion`

- If set to `1`, congestion map will be displayed. If set to `0`, not.

Below is an example of a completed call:

```
DisplayHeatMapAndSpaghettiDiagram('PickCart', 1, 1, 0, 1);
```

- On the web interface, this will display the heat map and spaghetti diagram for an object named `PickCart`. This is the same as model analysis in the Simcad interface.

Extension Two: Capture View for Live Tracking

Once the command for the heat map and spaghetti diagram is issued, a separate function will capture a view of the model, and export the model view to the DT web interface.

The following extension is used to capture a static view of the model for the DT web interface:

```
CaptureViewForTracking(WindowName, OverrideFolderName,
UniqueStringIdentifier)
```

The parameters of the function are as follows:

`WindowName`

- Identifies the department name if capture is performed in a department window. Leave empty if capturing the entire layout.

`OverrideFolderName`

Identifies the folder to which DT Studio will save the capture. Live view files and data will be stored on a virtual folder named *LiveView* on the server.

A sub-folder will be created under this virtual folder for each day the live system is active. Replays, images, resources, and other files will be stored here. A unique SVG file will be created for each capture.

There are two ways to access the virtual folder:

- Through the File Explorer navigation bar
- Via the website as a reference to the virtual folder

`UniqueStringIdentifier`

- Will set a unique value at the beginning of the file name to avoid overwriting. In the case that multiple captures are performed simultaneously, the value will be inserted at the beginning of each file name.

Extension Three: Capture Interactive View for Tracking

Once the command for the model capture is issued, a third function will capture a view of the objects in the model, and export the view of the objects in the model to the DT web interface.

The following extension is used to capture a view of the objects in the Digital Twin model:

```
CaptureInteractiveViewForTracking(WindowName, OverrideFolderName,
UniqueStringIdentifier, LayoutProcessName, PickerValue, ImageName
```

The parameters of the extension are as follows:

`CaptureInteractiveViewForTracking`

- Command issued to capture the interactive view in the DT Studio web interface.

`WindowName, OverrideFolderName, UniqueStringIdentifier`

- These components serve the same purposes as in the `CaptureViewForTracking` function.

`LayoutProcessName`

- The name of the Image Holder Process in Simcad which displays the layout. Note that the capture will only include objects within the borders of the defined layout.

- Note that multiple calls may be performed with multiple layouts, if multiple layouts exist in the model. This will create a combined view of the multiple layouts.
- Note that layouts must be under a certain size (8000x8000) for capture to work properly. If the layout does not display on the web interface, check the **Grid Properties** of the layout in Simcad.

Several corresponding functions enable the capture of the objects in the model, and are used in conjunction with the `CaptureInteractiveViewForTracking` command. They are:

`ExportModelImages(OverrideFolderName)`

- Exports everything that is an image file (objects, carriers, resources, etc.) to the defined folder. If on the DT Studio web interface, images are missing, it is because this command is not being used.

`CaptureModelViewForInteractiveTracking(WindowName, OverrideFolderName, UniqueStringIdentifier, LayoutProcessName, ScaleFactor)`

- Similar to the `CaptureInteractiveViewForTracking` command, this command when used with interactive tracking will capture an image of the model for interactive tracking on the DT web interface. Each component of the command serves the same function as in the `CaptureInteractiveViewForTracking` command.
- The only additional variable is `ScaleFactor`, which is a value between 0 and 1 which determines the resolution of the image. In cases where file size is a consideration, this variable may need to be set to a value less than 1.

The final component of the command will export all the data to a data table with an image file and a time stamp.

```
DB: DTStudio –AdvSQL: Insert into active DTS_ImageListForVideo
(VideoIdentifier, ImageFile)
```

All together, the model extension will look something like:

```
--BranchTo:exit;

--M_TempStr = CaptureViewForTracking(‘’, ‘C:\DTStudioImages\KNN\VideoImages’,
IntToStr(GetActiveScenarioIndex()) );

--DB: DTStudio –AdvSQL: Insert into active DTS_ImageListForVideo
(VideoIdentifier, ImageFile)

--Values (‘<%EXPS IntToStr(GetActiveScenarioIndex())%>’, ‘<%M_TempStr%>’ );

--New Capture

DisplayHeatMapAndSpaghettiDiagram(‘PickCart’, 1, 1, 0, 1);

M_TempStr = CaptureInteractiveViewForTracking(‘’, ‘s:\svg\LiveView\2023-06-
12’, ‘1’, ‘Layout’, ‘Reports/Resource-Analyzer?WorkerID=<%0_PickerID%>’,
‘/DTStudioCapture/svg

IF ((M_CaptureInterval MOD 100 ) -- 0)
```

```
        --DisplayMessage('Capturing');

        ExportModelImages('s:\svg\LiveView\2023-06-12');

        M_ImageFile = CaptureModelViewForInteractiveTracking('',
        's:\svg\LiveView\2023-06-12', '1', 'Layout', 0.25);

        M_ImageFile = 'LiveView/2023-06-12/' & m_ImageFile;

        DB: DTStudio –AdvSQL: Insert into DTS_TEST_SVGImageListForVideo
        (BGImageFile, SVGImageFile)

        Values ('<%M_ImageFile%>' '<%EXPS 'LiveView/2023-06-12/' &
        m_TempStr%>');

ELSE

        DB: DTStudio –AdvSQL: Insert into DTS_TEST_SVGImageListForVideo
        (BGImageFile, SVGImageFile)

        Values ( '<%M_ImageFile%>', '<%EXPS 'LiveView/2023-06-12/' &
        m_TempStr%>' );

END IF

M_CaptureInterval = M_CaptureInterval + 1;
```

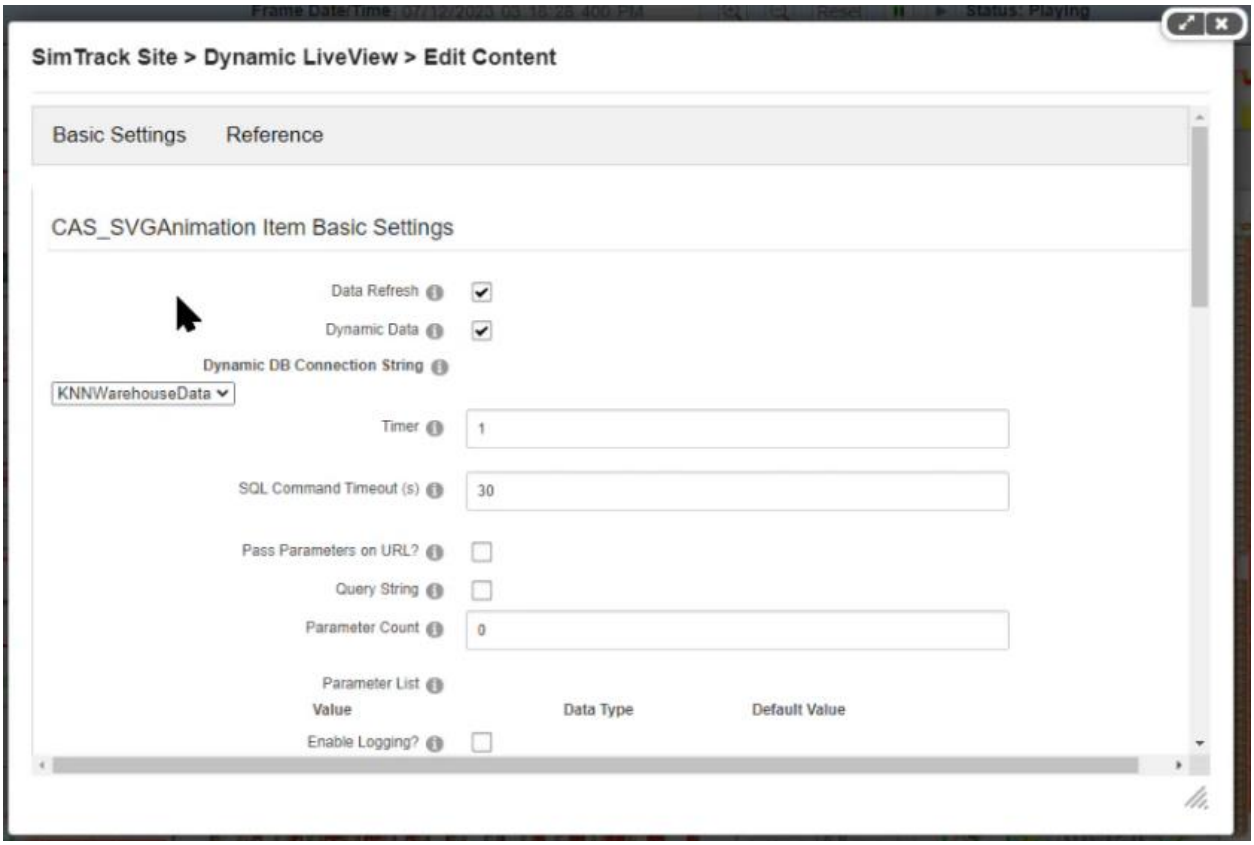
Editing Web Modules

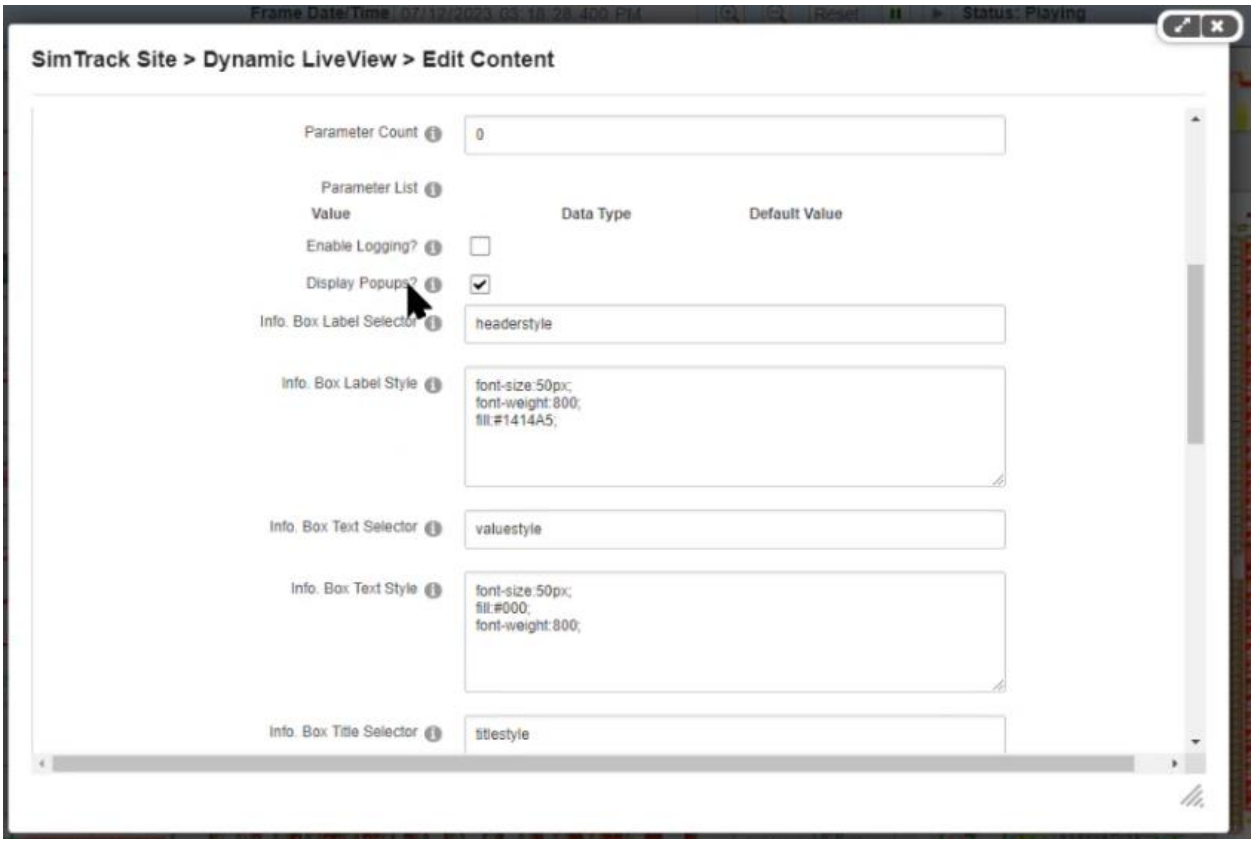
Simulation controls are outlined in red above. Play and pause controls may be used on the live Digital Twin. Re-wind and fast-forward controls may be used in replay mode.

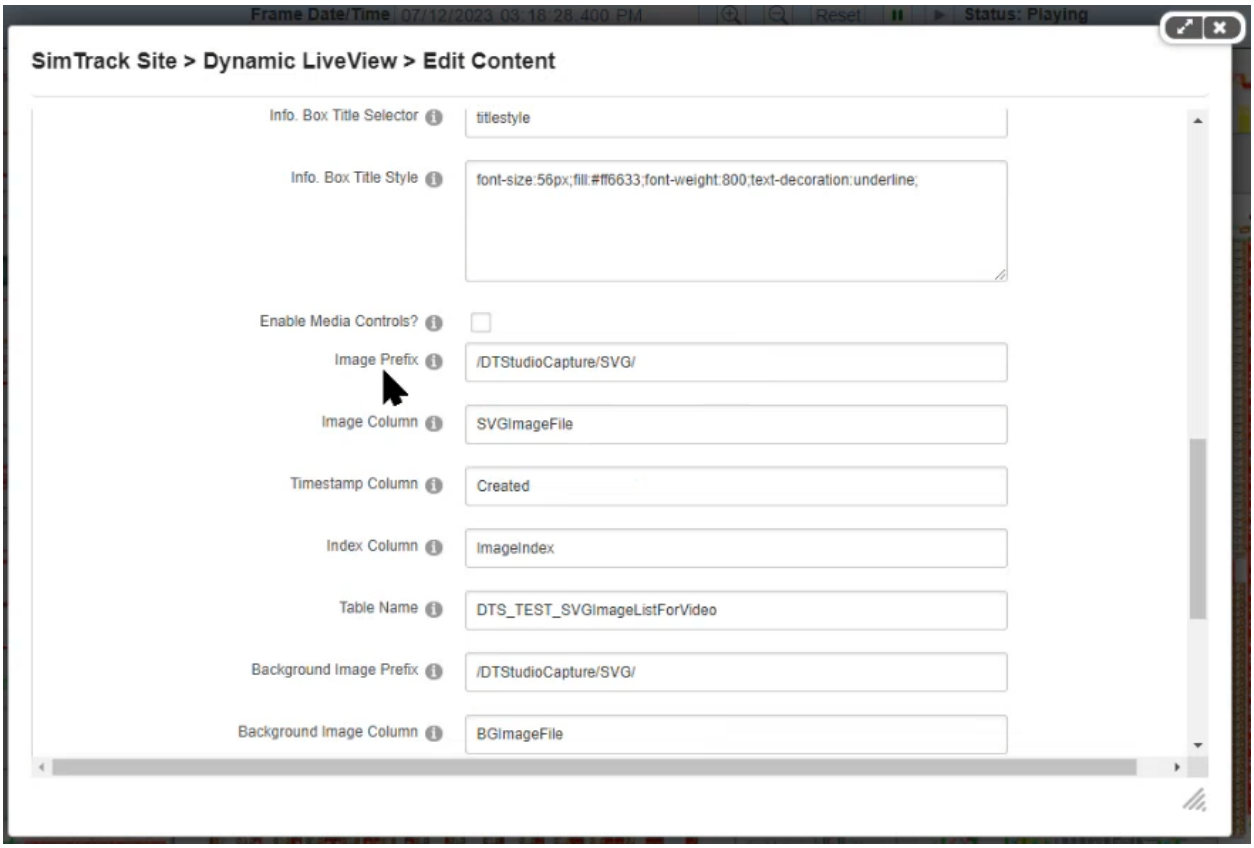
Frame Skip increases or decreases the fast-forward and re-wind speed in replay mode.

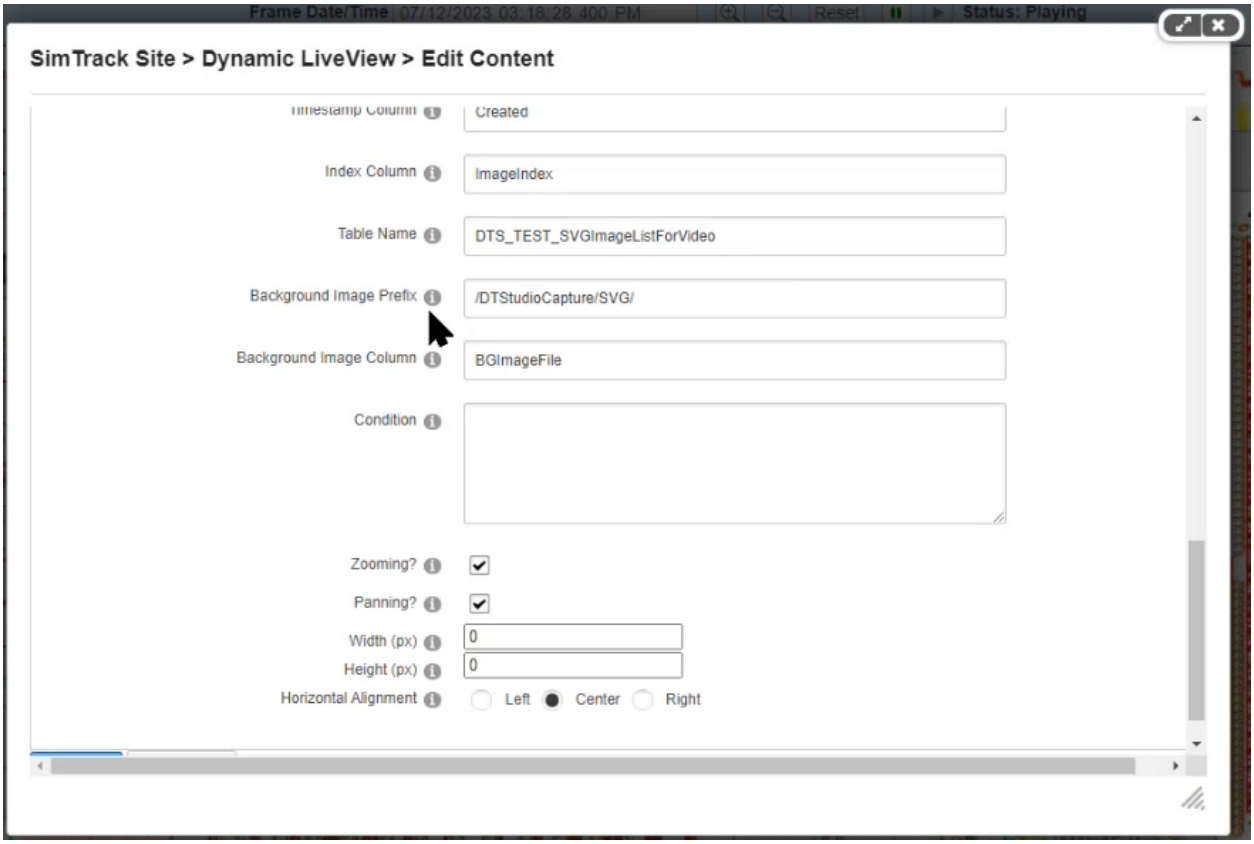
Frame Seek allows users to skip to a certain time stamp in the run.

Reset will reset the view of the model to the default state.




Frame Date/Time 07/12/2023 03:18:28.400 PM
Reset
Status: Playing
SimTrack Site > Dynamic LiveView > Edit Content
Parameter Count
0
Parameter List
Value
Data Type
Default Value
Enable Logging?
Display Popups?
Info. Box Label Selector
headerstyle
Info. Box Label Style
font-size:50px;
font-weight:800;
fill:#1414A5;
Info. Box Text Selector
valuestyle
Info. Box Text Style
font-size:50px;
fill:#000;
font-weight:800;
Info. Box Title Selector
titlestyle


Frame Date/Time 07/12/2023 03:18:28.400 PM
Reset
Status: Playing
SimTrack Site > Dynamic LiveView > Edit Content
Info. Box Title Selector
titlestyle
Info. Box Title Style
font-size:56px;fill:#ff6633;font-weight:800;text-decoration:underline;
Enable Media Controls?
Image Prefix
/DTStudioCapture/SVG/
Image Column
SVGImageFile
Timestamp Column
Created
Index Column
ImageIndex
Table Name
DTS_TEST_SVGImageListForVideo
Background Image Prefix
/DTStudioCapture/SVG/
Background Image Column
BGImageFile



Data Refresh must be selected to enable the live view.

Data Refresh must be de-selected in Replay mode. To perform replay, **Media Controls** must also be selected.

Timer defines how often, in seconds, the live view will refresh. Every time the module refreshes, the module will query the table from the SQL server, and display the last value.

Display Popups? will enable pop-up display for objects and resources in the web interface.

Image Prefix defines the file name in the DT Studio server. The file name will be appended with the date and time of the capture.

Image Column represents the column name in the SQL server. Columns may be added at a later date for future conditions that are needed.

Timestamp Column represents the column where timestamps are stored.

Zooming? and **Panning?** enable navigation by zoom and pan.

Width(px) and **Height(px)** allow finer control over the content size.

Once the parameters of the module have been set, the module will automatically set up a background query to capture these parameters and display them on the web interface.