```
// EXAMPLE - connecting to a M.App Enterprise Vectorset
var dataset = {
         mode: "EnterpriseConnector",
         vectorset: "Atlanta",
         datasetNaåme: "AtlantaBoundaries"
 };

// EXAMPLE - connecting to the M.App Enterprise SQLQuery API
var dataset = "https://enterprise.okehouse.net/api/v1/sqlquery/AtlantaDynLoad?query=SELECT * FROM atlantaattr&format=csv";

// EXAMPLE - connecting to an online file. Can be any online file supported by Feature Analyzer:
// - GeoJSON
// - GeoJSON-T
// - TopoJSON
// - CSV
// - Shape
// - JSON dataå
// - A Zip file containing any of the above.
var dataset = "https://earthquake.usgs.gov/earthquakes/feed/v1.0/summary/all_month.geojson";

var stage = await Analyzer.createStage("boundaries", onlineFile);

// Retreives a list of field names
stage.dataPreview.FieldNames()

// Returns the first row of data.
stage.dataPreview.FirstRow;

// Create a default theme widget with default parameters. By default this widget is not added to the stage.
// You can change parameters on the object once it has been created or use the configruation GUI once the widget has been added to the system.
// The second parameter is the theme field name. Ensure that this exists in the source data.
// If you accidentally added it to feature analyzer with a bad field name - no worries. You can always change the field name in the additional
var theme = Analyzer.createThemeWidget(stage, "status");

// Creates a default feature layer using hardware acceleration. Different types are available:
// * feature.type = "primaryFeatureLayer"; // Creates a layer for use in a leaflet 2D window. (DEFAULT)
// * feature.type = "luciadFeatureLayer"; // Creates a layer for use in a luciad 3D window.
// * feature.type = "clusterLayer"; // Creates a cluster marker layer for use in a leaflet 2D window.

// NOTE: If you have multiple 2D/3D windows, change the "dock" parameter to match the name of the view you wish to place your new feature.
// Eg. feature.dock = "My New Chart"; // given that a 2d/3d window exists with this name.
var feature = Analyzer.createFeatureLayer(stage);

// The two lines add the widgets to Feature Analyzer.
Analyzer.addWidget(theme);
Analyzer.addWidget(feature);

// If you're using M.App Enterprise, you'll need to reload the view. This will bring back any additional attributes needed to display the new s
Analyzer.reloadView();

// This line will link two stages together (similar to how boundary data works today). Parameters are:
// - StageModel A (first stage model)
// - StageModel A Key (key field to use to link to the secondary stage - similar to the primary key field used today for boundary data)
// - StageModel B (second stage model)
// - StageModel B Key (key field to use to link to the primary stage - similar to the foreign key field used today for boundary data.)
Analyzer.linkStageModel(Analyzer.findStageModel(), "beatid", Analyzer.findStageModel("boundaries"), "zone");

// Again - if you're using M.App Enterprise, you'll need to reload the view. This will bring back any additional attributes needed to link the
Analyzer.reloadView();
```