Migration Guide DB Gene 4.3.0

September 20th, 2024

DB Gene 4.3.0 Migration Guide

- *For more details, please refer to the DB Gene Documentation available **here**.*
- *The order of the sections and subsections is based on the DB Gene Release Notes Changelog, available **here**.*
- *Content in each subsection is structured as follows:*
  Removed, Deprecated, Updated, and New.
- *Changes made to the DB Gene REST API are listed in Section **Dev Changes** > **REST API**.*
- *Changes made to the DB Gene dependencies are listed in Section **Dev Changes** > **3rd-Party Components**.*
- *Deprecations planned for removal in 4.4.0 are listed in Section **Deprecated Features and APIs scheduled for Removal in 4.4.0**.*

Application Changes

This part lists all changes related to:

- *general features on application element management, including events and hooks in Section* ***General Application Changes****.*
- *permissions and API keys, including the UI in Section* ***Access Control****.*
- *the application configuration file in Section* ***Application Configuration****.*
- *the lifecycle, non-UI features, or APIs of views and dashboards in Section* ***Views & Dashboards****.*
- *the workspaces and scenarios lifecycle, features, or APIs in Section* ***Workspaces & Scenarios****.*

General Application Changes

This section lists all general changes to application element management, including events and hooks.

New Beta Features

For the first time, DB Gene introduces beta features. They need to be generated during the build phase and can be enabled from the Application Preferences.

They are as follows:

- JupyterLab Integration
- Code Editor widget
- Rule Script Editor and Runtime

For more details on how to enable beta features, please refer to Section ***Build*** in ***Dev Changes*** and Section ***Application Preferences*** in ***UI Changes***.

For more details on the JupyterLab integration, please refer to Section ***Python*** in ***Dev Changes***.

For more details on the code and script editors, please refer to Section ***Code Editor*** in ***UI Changes***.

- -

Access Control

This section lists all changes related to permissions and API Keys, including the UI.

Updated Requirements for the Web Client Password

The password policy now requires a password with a minimal length of 8 with letters, numbers, and special chars. (The username is excluded from possible passwords.)

Updated Security for the Web Client Login

The user accounts are locked for 1 minute after 5 failed login attempts.

- -

Application Configuration

This section lists all changes related to the application configuration file, not the ***Application Preferences****.*

No changes

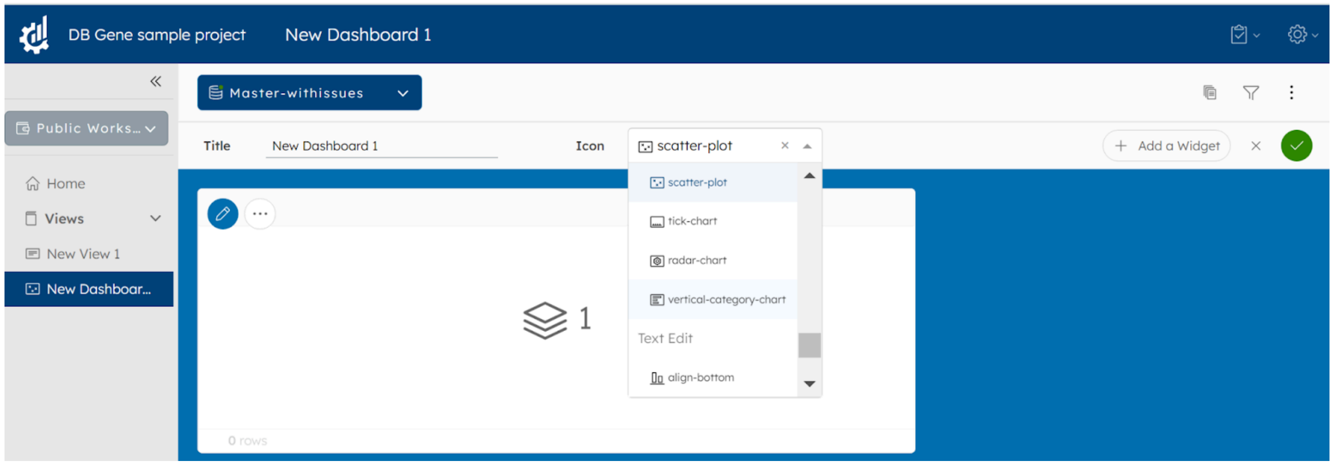
- -

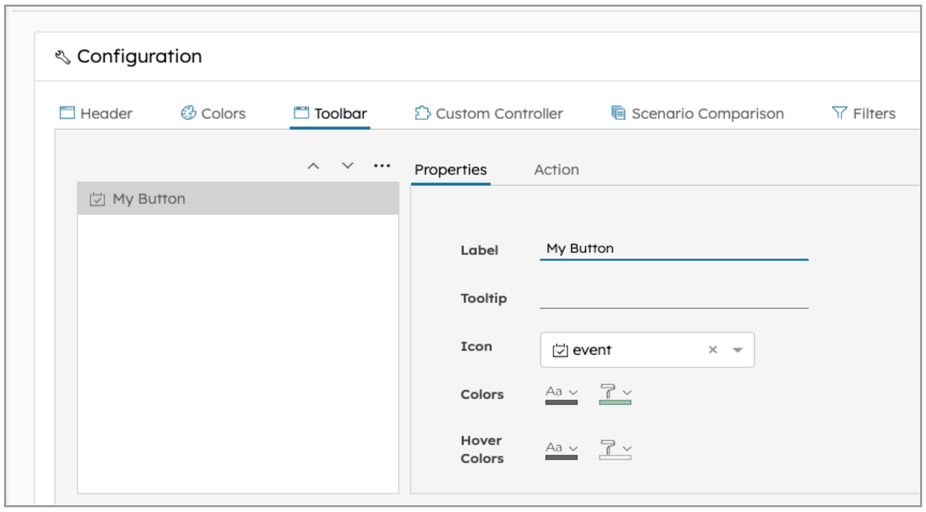
Views & Dashboards

This section lists all changes related to the lifecycle, the non-UI features, or the APIs of views and dashboards.

Improved Sidenav Icons

When editing the layout of a custom view or dashboard, users can now configure an *Icon* in addition to the *Title* that is displayed in the Sidenav.

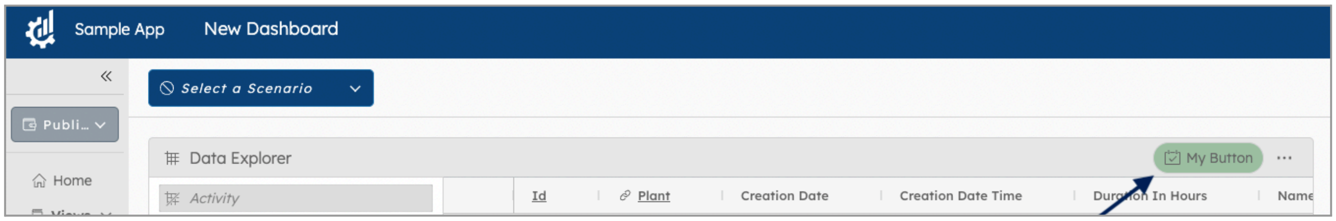




New Configurable Buttons for the Widget Toolbar

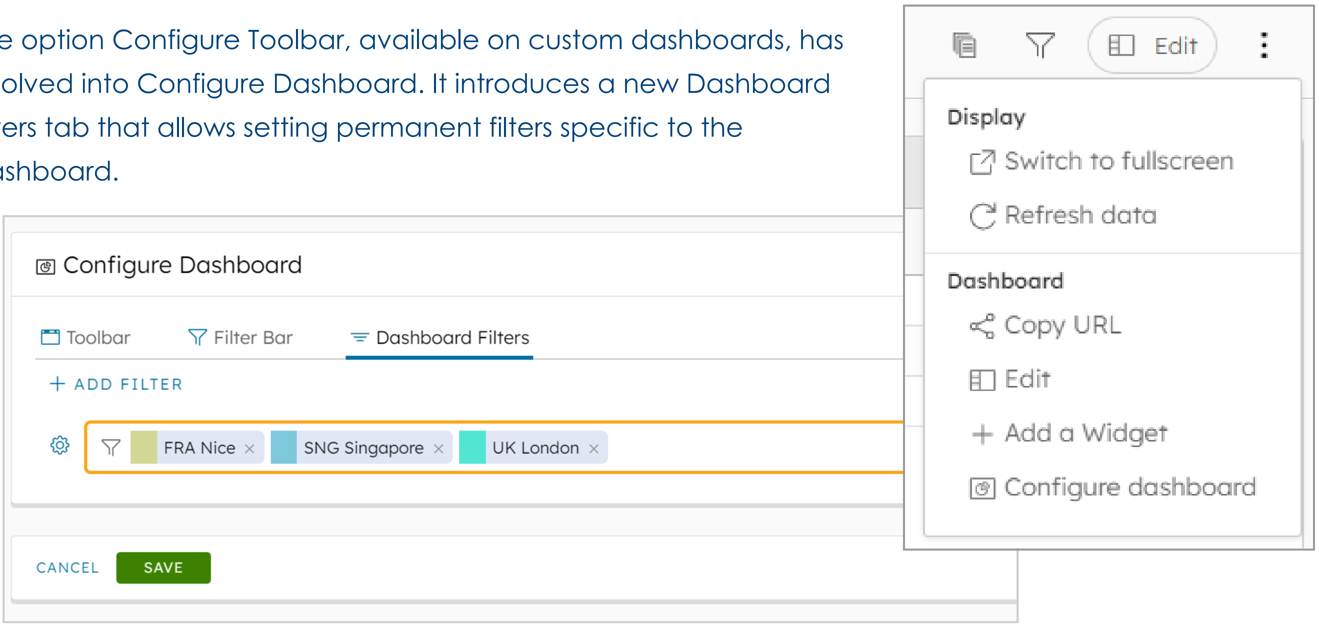
Every widget configurator now displays a new *Toolbar* tab.

It allows users to add action buttons, with labels, custom colors, and icons, to the toolbar of any widget.



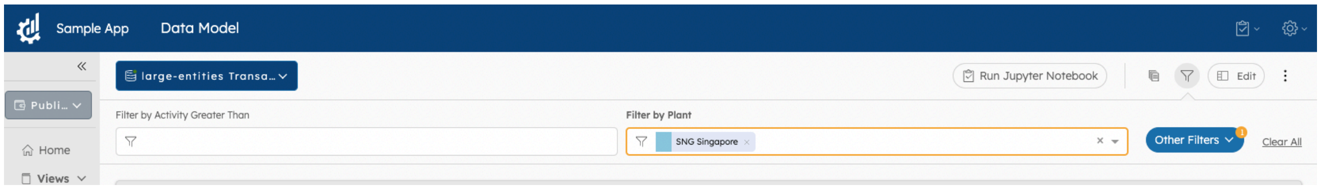
New Dashboard Filters

The option Configure Toolbar, available on custom dashboards, has evolved into Configure Dashboard. It introduces a new Dashboard Filters tab that allows setting permanent filters specific to the dashboard.



New Filter Bar

A new Filter Bar can now be displayed on all custom dashboards. It provides users with additional dynamic filtering on top of any permanent filters already set on the dashboard or on the widgets.



When using a Filter Bar, filters from other dashboards now show up in the Other Filters dropdown instead of in the ⏷ Context Selection dropdown which now switches to a toggle button.

New Filtering Scope

In the Application Preferences, the new FILTER_SCOPE parameter is set by default to GLOBAL. This means that the Filter widgets and new Filter Bar apply to all views and dashboards across the application.

It can be set to VIEW to limit the filtering to the dashboard or view of the element.

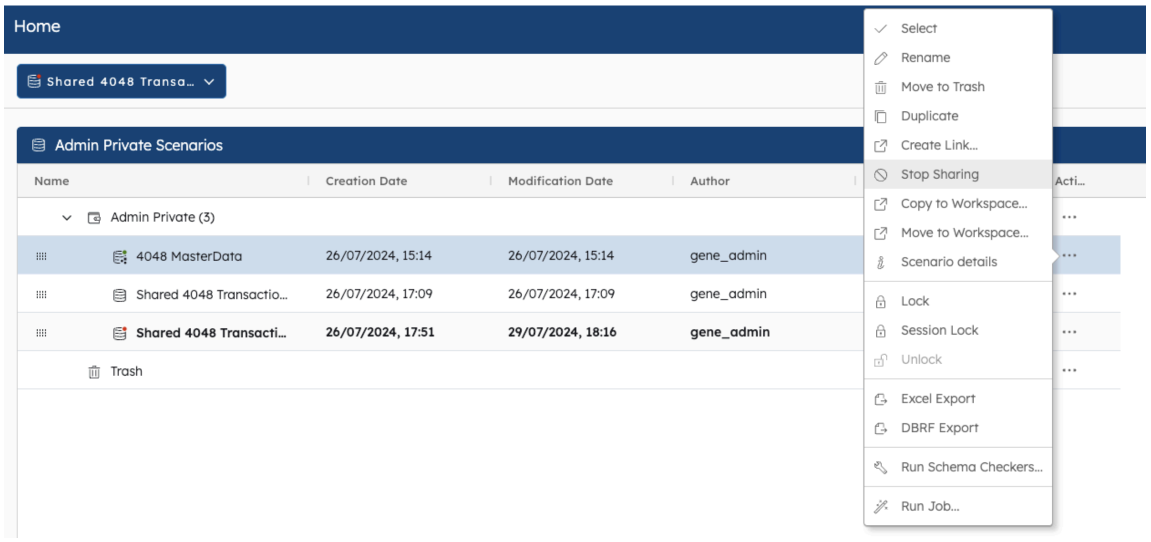
Workspaces & Scenarios

This section lists all changes related to the lifecycle, the features, or the APIs of workspaces and scenarios.

Updated Scenario Management

Several improvements in DB Gene 4.3.0 impact scenario management:

- Users can now select a scenario link as a reference when adding a CDM scenario.
- They can also duplicate a scenario and the referenced scenarios they have access to.
- The permissions evaluation has changed. Now, when a user creates a scenario link or a CDM scenario referencing another scenario, a permission rule giving access to the referenced scenario is created. This allows users to revoke access to a specific scenario referenced through scenario links or other CDM scenarios.
    - Note that users can only display the shared scenario when they have access to the scenario's workspace or when they have access to the created scenario link. For an application in a previous version of DB Gene with scenarios referenced by one or more scenario links, once the application is migrated to DB Gene 4.3.0, these scenarios are automatically migrated and associated with a permission rule giving access to the scenario. Also, The new permission rule will override any existing one.
- If the scenario is shared with everyone through a scenario link or a CDM scenario, the web client now displays an indicator on the scenario icon and the option Stop Sharing in the Action menu.



Name	Creation Date	Modification Date	Author	Acti...
Admin Private (3)				...
4048 MasterData	26/07/2024, 15:14	26/07/2024, 15:14	gene_admin	...
Shared 4048 Transactio...	26/07/2024, 17:09	26/07/2024, 17:09	gene_admin	...
Shared 4048 Transacti...	**26/07/2024, 17:51**	**29/07/2024, 18:16**	**gene_admin**	...
Trash				...

Data Changes

This part lists all changes related to:

- *concepts or the meta-model representation in Section **Model**.*
- *the language definition and parsing in Section **JDL**.*
- *generated code, either SQL, Java, Python, or else in Section **Generation**.*
- *the database schema, storage, or queries in Section **Database**.*
- *the GraphQL interface elements or extensions in Section **GraphQL**.*
- *the import or export mechanism provided by the product in Section **Built-in Import/Export**.*
- *the Data Integration core API in Section **Data Integration Framework**.*
- *the Data Service API in Section **Data Service**.*
- *the Scenario Service API in Section **Scenario Service**.*

Model

This section lists all changes related to concepts or the meta-model representation.

No changes

- -

JDL

This section lists all changes related to the language definition and parsing.

No changes

- -

Generation

This section lists all changes related to generated code, either SQL, Java, Python, or else.

No changes

- -

Database

This section lists all changes related to the database schema, storage, or queries.

No changes

GraphQL

This section lists all changes related to the GraphQL interface elements or extensions.

Deprecated Property in the Scenario Service GraphQL API

The Scenario Service GraphQL API was returning the `Path { scenarioReferenyceGraph }` property, which is incompatible with GraphQL structures. This property will be removed in the next version. It is recommended to use the corresponding REST APIs to access this property.

Improved GraphQL Introspection

As it should not be activated in production, GraphQL introspection feature is now disabled by default for the Data Service and Scenario Service.
To enable the introspection in development, use the following Spring property:

```
Unset
  spring:
    graphql:
      schema:
        introspection:
          enabled: true
```

Built-in Import/Export

This section lists all changes related to the import or export mechanism provided by the product.

No changes

Data Integration Framework

This section lists all changes related to the Data Integration core API.

Deprecated DBPF and CSV collectors

CSV collectors as well as the DBPF file format are now deprecated in favor of the XCSV format in scripted tasks and DOMCollector APIs.

New XCSV File Format

The new XCSV format to export or import scenario files now contains both the internal ID (as CSV collectors) and the business key (as in DBRF files).

An XCSV file has the `.xcsv` extension. It is a ZIP file that contains `.csv` files. In short, it is a CSV collector to which the business key fields have been added. In addition, the XCSV format now replaces the CSV format in scripted tasks.

The new endpoint `/data/xcsv-export` or the XCSVExportTask task can now be used to export a scenario to XCSV.

Data Service

This section lists all changes related to the Data Service API.

Deprecated Methods in the Data Service REST API

The following methods of the data-service-base library have been deprecated:

- In `BatchCollectorService`, method `saveItems(String scenarioId, List<T> entityDTOs)` has been deprecated in favor of `saveEntities(String scenarioId, List<T> entityDTOs)`.
- In `ScenarioUpdateService`, method `saveItems(String, List<DataServerEntityDTO> items, SchemaCheckersRunOptions)` has been deprecated in favor of `saveEntities(String, List<DataServerEntityDTO> entities, SchemaCheckersRunOptions)`.

Scenario Service

This section lists all changes related to the Scenario Service API.

Deprecated Excel Import/Export Endpoints in the Scenario Service REST API

The following endpoints have been deprecated:

- `/data/simple-excel-export`, replaced by `/data/excel-export`.
- `/data/simple-excel-import`, replaced by `/data/excel-import`.

The deprecated parameter `sortColumns` has also been removed from the Excel Export API.

Updated Scenario Service REST API

`PathEventDTO` model has been updated, its field `modificationDate` has been renamed to `lastModificationDate`.

DBOS Changes

This part lists all changes related to:

- *the DBOS worker registration, job scheduling & monitoring, and communication with Gene in Section* ***Master****.*
- *the DBOS Java task definition and worker library in Section* ***Worker (Java)****.*
- *the DBOS Python task definition, worker library, and data model mapping in Section* ***Worker (Python)****.*
- *the DBOS workers provided with the product in Section* ***Predefined Workers****.*
- *the DBOS Web Console in Section* ***Web Console****.*
- *the ability for customers to define other DBOS master clients than the Web Console in Section* ***Clients****.*

Master

This section lists all changes related to the DBOS worker registration, job scheduling & monitoring, and communication with Gene.

No changes

- -

Workers (Java)

This section lists all changes related to the DBOS Java task definition and worker library.

Updated Scenario Data Format for Java workers

The transferred scenario data is now in XCSV format instead of CSV.

Consequently, the default `DbDomCollector#loadSnapshot` and `DbDomCollector#saveSnapshot` now use the XCSV data format.

- -

Workers (Python)

This section lists all changes related to the DBOS Python task definition, worker library, and data model mapping.

Deprecated CSV Collector for Python Workers

The CSV collector, which is a `.zip` archive of `.csv` files, is now deprecated and has been replaced by the XCSV format.

New Business Key in XCSV Format for Python Workers

With the introduction of the XCSV format, the data frames loaded in the worker now contain the business key fields of each entity in addition to their internal ID. In a composite data model application, one must pay particular attention when saving data frames with consistent business keys across multiple scenario types.

In addition, when saving a Python collector to XCSV, the business keys are saved, too.

New Debug Configuration for Python Workers

Developers can use IntelliJ to debug a running Python worker. To do so in IntelliJ, one must:

- Add the *Run configuration* of type *Gradle*.
- Add the *Run configuration* of type *Python Debug Server*.
- Start the *Run configuration* of type *Python Engine Worker* in *debug* mode.
- Start the *Run configuration* of type *Python Debug Server*.

Developers can then add debug breakpoints to the Python code.

- -

Predefined Workers

This section lists all changes related to the DBOS workers provided with the product.

No changes

- -

Web Console

This section lists all changes related to the DBOS Web Console.

No changes

- -

Clients

This section lists all changes related to the ability, for customers, to define other DBOS master clients than the Web Console.

No changes

Dev Changes

This part lists all changes related to:

- *the DB Gene REST API in Section* ***REST API****,*

  except for the ***Data Integration Framework****,* ***Data Service****, and* ***Scenario Service*** *API.*
- *the DB Gene external dependencies and libraries, such as Spring, Angular, or Keycloak*

  in Section ***3rd-Party Components****.*
- *the DB Gene Scaffolding and Gradle scripts in Section* ***Build****. any DB Gene security issues*

  in Section ***Security****.*
- *the DB Gene integration to Docker or Helm in Section* ***Deployment****.*
- *the Python integration to DOC, except for* ***Workers (Python)*** *and* ***Routines*** *in Section* ***Python****.*
- *the DB Gene integration to Tableau in Section* ***Tableau****.*
- *the DB Gene integration to Jupyter in Section* ***Jupyter****.*
- *the DB Gene integration to CPLEX in Section* ***CPLEX****.*
- *the DB Gene and DBOS documentation in Section* ***Documentation****.*

REST API

This section lists all changes related to the DB Gene REST API, except for the Data Integration Core, Data Service, and Scenario Service API.

No changes

- -

3rd-Party Components

This section lists all changes related to the DB Gene external dependencies and libraries, such as Spring, Angular, or Keycloak.

Updated Angular Dependencies

Angular has been updated to version 18.2.2. Changes are described in this Medium article.

Unset
```
// ------ Updated Angular Dependencies
@angular-devkit/build-angular -> 18.2.1
@angular/animations -> 18.2.2
@angular/cdk -> 18.2.2
@angular/cli -> 18.2.1
@angular/common -> 18.2.2
@angular/compiler -> 18.2.2
@angular/compiler-cli -> 18.2.2
@angular/core -> 18.2.2
@angular/forms -> 18.2.2
@angular/language-service -> 18.2.2
@angular/localize -> 18.2.2
@angular/platform-browser -> 18.2.2
@angular/platform-browser-dynamic -> 18.2.2
@angular/router -> 18.2.2
@cds/angular -> 6.10.0
@cds/core -> 6.10.0
@clr/angular -> 17.0.1
@clr/ui -> 16.3.7 (not latest, see DBPF-5940)
@danielmoncada/angular-datetime-picker -> 18.1.0
@fullcalendar/angular -> 6.1.15
@fullcalendar/core -> 6.1.15
@ng-select/ng-select -> 13.7.0
ag-grid -> 31.3.2
angular-gridster2 -> 18.0.1
keycloak-angular -> 16.0.1
keycloak-js -> 24.0.2
ngx-quill -> 26.0.8
quill -> 2.0.2
typescript -> 5.5.4
zone.js -> 0.14.10
daypilot-pro-angular -> 2024.3.6155
```

```
// ------ Removed Angular Dependencies

@asymmetrik/ngx-leaflet
@asymmetrik/ngx-leaflet-markercluster

// ------ New Angular Dependencies

@bluehalo/ngx-leaflet
@bluehalo/ngx-leaflet-markercluster

// ------ New Angular Dependencies (Beta Feature)
monaco-editor 0.49.0
ngx-monaco-editor-v2 18.0.1
```

Typescript has been updated from 5.2 to 5.5, implying notable changes, such as:

- Regular Expression Syntax Checking
- Support for New ECMAScript Set Methods .

The list of changes can be found at:

- https://devblogs.microsoft.com/typescript/announcing-typescript-5-3/
- https://devblogs.microsoft.com/typescript/announcing-typescript-5-4/
- https://devblogs.microsoft.com/typescript/announcing-typescript-5-5/

Updated Java Dependencies

Unset
```
// ------ Updated Java Dependencies

0A.pache commons compress -> 1.26.1
Apache commons text -> 1.12.0
Apache POI -> 5.2.5
Gradle -> 7.6.4
Keycloak -> 24.0.2
mongock -> 5.4.4
Node -> 20.17
RabbitMQ -> 3.13.0
Spring Cloud -> 2023.0.1
Spring -> 3.2.4
```

```
// ------ New Java Dependencies (Beta Feature)

drools-mvel in 8.44.0.Final
drools-engine
```

New Python Dependency

The module `pydevd-pycharm` has been added to the Python dependencies required during development.

It is used to enable IntelliJ Python worker debugging.

New Spring Property Migrator Tool

In the different extensions (`backend-extension`, etc...), project properties can be customized with specific code or Spring behavior.

With each Spring migration, these properties might be subject to changes induced by Spring libraries.

To help detect and deal with these cases, Spring provides users with a tool called Property Migrator.

The Property Migrator tool must be added at runtime in the Spring application and be launched only once to detect which properties should be migrated.

To add the Property Migrator to the classpath, uncomment the following lines in the file `gradle/templates/java-spring-application.gradle`:

```
Unset
dependencies {

    // For Spring migration ONLY
    runtimeOnly("org.springframework.boot:spring-boot-properties-migrator")

}
```

When the micro-service is launched with the Property Migrator, the logs indicate which properties to migrate as follows:

```
Unset
2024-05-15T19:59:41.452+02:00  WARN 24872 --- [backend-service] [
main] o.s.b.c.p.m.PropertiesMigrationListener  :
The use of configuration keys that have been renamed was found in the
environment:


Property source 'Config resource 'class path resource [application.yml]' via
location 'optional:classpath:/'':
      Key: spring.datasource.separator
            Line: 43
            Replacement: spring.sql.init.separator




Each configuration key has been temporarily mapped to its replacement for
your convenience. To silence this warning, please update your configuration
to use the new keys.
```

Once each deprecated property has been fixed for all the micro-services, the Property Migrator dependency has to be commented back because it should not be packaged in production applications.

---

Build

This section lists all changes related to the DB Gene Scaffolding and Gradle scripts.

Deprecated Old DSL of Code Replicate Plugin

The content of the block `codeUpdates { }` should be migrated to `codeReplicas { }`.

Updated Code Replicate Plugin

The plugin `com.decisionbrain.code-replicate-plugin` has been updated to version 0.9.8. Consequently:

- It has been renamed `com.decisionbrain.gradle.code-replicate-plugin`.
- It uses a new DSL (Domain Specific Language) but supports backward compatibility.

The old DSL is still available with the `block codeUpdates {}` and the new with the `codeReplicas {}`.

The scaffolded `build.gradle` files have been migrated but it is necessary to also migrate the block `codeReplicas {}` if it has been customized, Here is a comparison of the old and new DSL:

Unset
```
//Old DSL

codeUpdates {
  globalVersion(globalValue)

  update("file.txt") {
    regex("foo(.*?)bar")
  }

  update("setup.py") {

pythonVersion('version="%s"')
  }
}
```

Unset
```
//New DSL

codeReplicas {
  withValue(globalValue)

  inFile("file.txt") {
    atLocation(regex("foo(.*?)bar"))
  }

  inFile("setup.py") {
    withValuePythonVersion()

atLocation(regexPythonVersion('version="%s"')
  }
}
```

For more details, please refer to Section "Learn more" in the official plugin documentation.

Updated Gradle Files for Python

In the `python-engine-worker` module, the file `build.gradle` has been updated. Some of its content has been moved to the new file `python-worker.gradle`.

In addition, the files `python-library.gradle` and `python-root.gradle` have also been changed.

Updated Gradle Files for CPLEX

The file `local-cplex.gradle`, which eases the CPLEX integration from a local CPLEX installation into a Java library, has been updated. In addition, the file `local-cplex-worker.gradle` was added to integrate CPLEX into the module `engine-worker`. For more details, please refer to the Platform documentation.

Updated Packaging Dependencies

In the file `package.json`, the web client dependencies are now sorted alphabetically. This facilitates migration from previous versions as it makes comparison between `package.json` files easier to read.

New Option for Beta Features in the Application Generator

The `generator.sh` script now accepts a new option to enable beta features in the scaffolded project.

To do so, the `generator.sh` script can be called:

- with `--experimentalFeatures`,
- with `--experimentalFeatures=yes`. or
- without any form of the `--experimentalFeatures` flag, but with any form of the `--dbInternal` flag.

If one calls the script with no flag or with `--experimentalFeatures=no`, the scaffolded project will not contain beta features.

Updated Helm Chart

The scaffolded Helm chart now allows users to pass arguments to the `mongodb` command line.

In addition, the Helm chart parameter `InitialRAMPercentage` is now removed from the scaffolded file `/deployment/helm/src/values.yaml`.

- -

Security

This section lists all changes related to any DB Gene security issues.

No changes

- -

Deployment

This section lists all changes related to DB Gene integration to Docker and Helm.

New Deployment Files for Jupyter Notebook

Deployment files for Docker and Helm have been added for Jupyter.

- -

Python

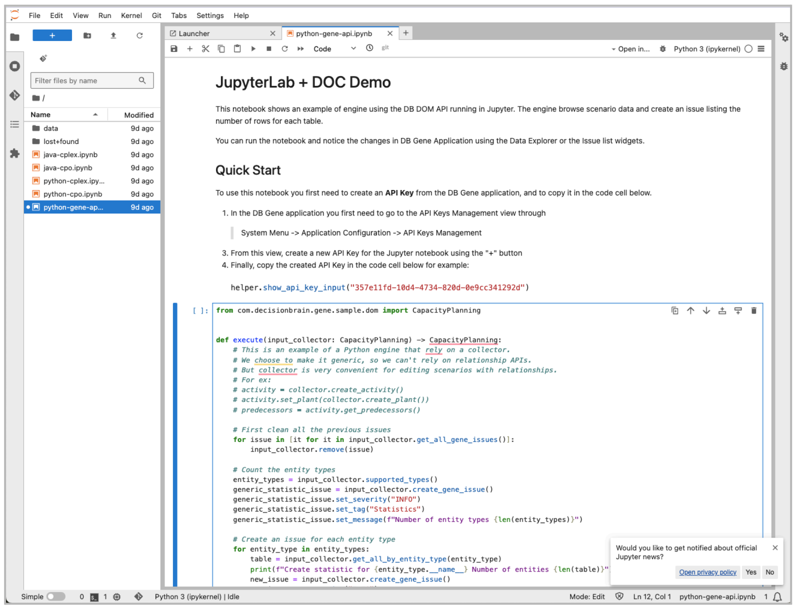
This section lists all changes related to Python integration, except for ***Workers (Python)*** *and* ***Routines***.

New Python Module for Jupyter

A Python module called `processing/jupyter-notebook` has been added.

It invokes a JupyterLab interface (formerly Jupyter Notebook) which allows processing scenario data and updating scenarios accordingly.

Developers can locally run the JupyterLab interface from the Docker Compose stack.



JupyterLab + DOC Demo

This notebook shows an example of engine using the DB DOM API running in Jupyter. The engine browse scenario data and create an issue listing the number of rows for each table.

You can run the notebook and notice the changes in DB Gene Application using the Data Explorer or the Issue list widgets.

Quick Start

To use this notebook you first need to create an **API Key** from the DB Gene application, and to copy it in the code cell below.

1. In the DB Gene application you first need to go to the API Keys Management view through

   > System Menu -> Application Configuration -> API Keys Management

3. From this view, create a new API Key for the Jupyter notebook using the "+" button
4. Finally, copy the created API Key in the code cell below for example:

```
helper.show_api_key_input("357e11fd-10d4-4734-820d-0e9cc341292d")
```

```
[ ]: from com.decisionbrain.gene.sample.dom import CapacityPlanning


def execute(input_collector: CapacityPlanning) -> CapacityPlanning:
    # This is an example of a Python engine that rely on a collector.
    # We choose to make it generic, so we can't rely on relationship APIs.
    # But collector is very convenient for editing scenarios with relationships.
    # For ex:
    # activity = collector.create_activity()
    # activity.set_plant(collector.create_plant())
    # predecessors = activity.get_predecessors()

    # First clean all the previous issues
    for issue in [it for it in input_collector.get_all_gene_issues()]:
        input_collector.remove(issue)

    # Count the entity types
    entity_types = input_collector.supported_types()
    generic_statistic_issue = input_collector.create_gene_issue()
    generic_statistic_issue.set_severity("INFO")
    generic_statistic_issue.set_tag("Statistics")
    generic_statistic_issue.set_message(f"Number of entity types {len(entity_types)}")

    # Create an issue for each entity type
    for entity_type in entity_types:
        table = input_collector.get_all_by_entity_type(entity_type)
        print(f"Create statistic for {entity_type.__name__} Number of entities {len(table)}"
        new_issue = input_collector.create_gene_issue()
```

Would you like to get notified about official Jupyter news?

Open privacy policy Yes No

- -

Jupyter

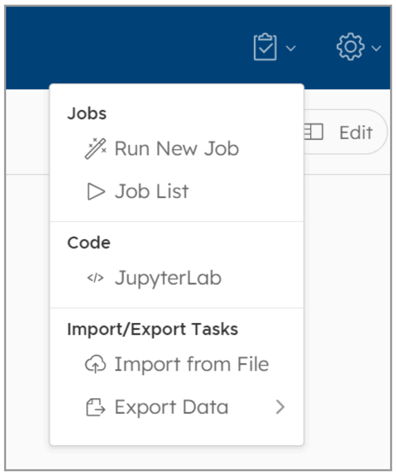
This section lists all changes related to Jupyter.

New JupyterLab Integration

The new integration of JupyterLab into the Platform as a beta feature eases online development with Python, Java, and CPLEX.

Users are provided with an option in the Topbar Tasks menu and the following sample notebooks in a base Docker image:

- CPLEX for Python
- CPLEX for Java
- CPO for Python
- CPO for Java



In addition, the Helm chart has been improved to allow deploying the JupyterLab service**.**

- -

Tableau

This section lists all changes related to Tableau.

No changes

- -

CPLEX

This section lists all changes related to CPLEX.

Updated CPLEX Integration

The CPLEX integration has been improved. For more details, please refer to the Platform documentation.

- -

Documentation

This section lists all changes related to the DB Gene and DBOS documentation.

Updated Documentation

Documentation on all widget configurators and Prometheus endpoints are now available.

In addition, the documentation now explains how to prevent scenarios from being moved to the Lost and Found workspace.

Scripted Tasks Changes

This part lists all changes related to:

- *the task definition and script language in Section **Definition**.*
- *job triggering, execution, and monitoring in Section **Jobs**.*
- *routines in Java & Python in Section **Routines**.*

Definition

This section lists all changes related to the task definition and script language.

Updated File Formats in Scenario Processing Method

In the method `ScenarioDataExpression`, the CSV and DBRF file formats have been marked as deprecated. XCSV is now the new default file format.

Updated Parameter in Scenario Creation Method

In the method ScenarioCreationExpression.of, the first parameter has been changed to an expression that evaluates to `Types.SCENARIO_CREATION_PARAMETERS`.

There are two ways to obtain such value within a scripted task, either by building it with `ScenarioCreationParametersExpression`, or from a job input as follows:

```
Unset
VariableAccessExpression scenarioCreationParameters =
VariableAccessExpression.of(SCENARIO_CREATION_PARAMETERS)
task.getScript()

.addStatement(AskInputStatement.of(scenarioCreationParameters.getVariableNam
e(), true, JobInputType.SCENARIO_CREATION_PARAMETERS, "Parameters to create
the Scenario"));
// Then use the expression scenarioCreationParameters whenever you need
```

Jobs

This section lists all changes related to job triggering, execution, and monitoring.

Updated Tasks

Scenario creation is now available in all scripted tasks and can be automated.

In addition, scenario creation tasks now also allow setting scenario characteristics.

Finally, the sample task *Create an empty scenario* has been improved for CDM applications.

- -

Routines

This section lists all changes related to routines in Python or Java.

Deprecated File Formats in Routines

In routines, the CSV and DBRF file formats have been marked as deprecated

New File Format in Routines

In routines, XCSV is now the new default file format.

UI Changes

This part lists all changes related to:

- *all widgets, regardless of their type, or non-widget elements of the UI, such as the sidebar, menus, scenario picker, or views and dashboards, except for* ***Access Control*** *elements, such as permissions and API keys in Section* ***General UI Changes****.*
- *the Application Preferences, not its* ***Configuration*** *in Section* ***Application Preferences****.*
- *the UI layout, styling, theming, and interaction principles in Section* ***Look & Feel****.*
- *the Scenario Comparison Mode in Section* ***Scenario Comparison****.*
- *the custom renderers and controllers in Section* ***Extensibility****.*
- *all table-based widgets, regardless of their type, such as the Data Explorer, Data Grid, Job List, Scenario List, Workspace List, Issue List, and Issue Details widgets in Section* ***Tables****,*
- *specific UI widgets in Section* ***Data Grid/Explorer****,* ***Scenario****,* ***Issue****,* ***Job****,* ***Navigation Button****,* ***Chart****,* ***Gantt****,* ***KPI****,* ***Filter****,* ***Composite****,* ***Map****,* ***Calendar****,* ***Rich Text****, or* ***Pivot Table*** *Widgets — if a change impacts more than one widget, it is listed in the most relevant section and referred to in the others.*

General UI Changes

This section lists all changes related to all widgets, regardless of their type, or non-widget elements of the UI, such as the sidebar, menus, scenario selector, or views and dashboards, except for the ***Access Control*** *elements, such as permissions and API keys.*

No changes

- -

Application Preferences

This section lists all changes related to the Application Preferences, not the ***Application Configuration****.*

New Parameter for Beta Features in Application Preferences

A new *EXPERIMENTAL_FEATURE* parameter is set to *true* by default in the Application Preferences.

New Option for Application Controllers in Application Preferences

The application controller mechanism has been improved. Application controllers are no longer set by code but by configuration through the Application Preferences.

Updated Scenario Import Parameter in Application Preferences

The parameter *DEFAULT_EXCEL_IMPORT_TASK_ID* has been removed in favor of *DEFAULT_SCENARIO_IMPORT_TASK_ID*.

Requesting to import an Excel file from the web client now launches a job from the task defined using the *DEFAULT_SCENARIO_IMPORT_TASK_ID* parameter.

Look & Feel

This section lists all changes related to the UI layout, styling, theming, and interaction principles.

No changes

Scenario Comparison

This section lists all changes related to the Scenario Comparison Mode.

No changes

Extensibility

This section lists all changes related to the custom renderers and controllers.

Deprecated Method in the Application Controller API

The method `setCustomApplicationController(..)` from the `GeneApplicationService` has been deprecated and replaced by `registerApplicationController(..)`.

Updated Application Controller API

The application controller mechanism has been improved. Application controllers are no longer set by code but by configuration through the Application Preferences.

Also, developers can now register multiple application controllers through the API.

Unset
```
    // Register a custom GeneApplicationController.
    // @param controllerName the name of the application controller
    // as it will be displayed in the UI
    // @param controllerFactoryFn the factory function
    // that will create the controller instance
    registerApplicationController(controllerName: string,
                                  controllerFactoryFn: (injector:Injector)
=> GeneApplicationController);
```

The application controller API has also been improved to let developers provide the sidebar and the header bar with custom components to replace the default ones.

In the folder `web/modules/sample-controllers/application`, some code samples illustrate how to use this API.

Unset
```
export interface GeneApplicationConfiguration {
   // ...
  header?: {
        // When provided, the component will replace
        // the default Gene component for the header bar
        component?: Type<any>;
       // ...
     },
    sidenav?: {
            /**

        // When provided, the component will replace
        // the default Gene component for the sidebar
        component?: Type<any>;
    }
}
```

- -

Tables

This section lists the changes related to all table-based widgets, regardless of their type, such as the Data Explorer, Data Grid, Job List, Scenario List, Workspace List, Issue List, and Issue Details widgets.

Updated AG Grid Import

After the AG Grid upgrade to 31.3.2, some imports may have to be updated, for example, in the custom controller code.

This applies to most imports where a path was needed, such as:

```
Unset
import { ValueGetterParams } from
'ag-grid-community/dist/lib/entities/colDef';
```

As with most other `ag-grid` imports, this can now be simplified as follows:

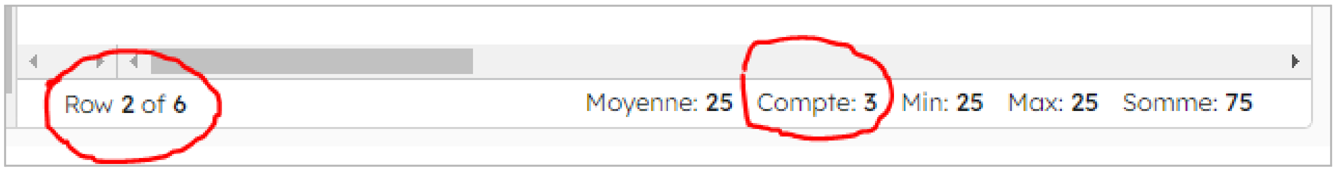
```
Unset
import { ValueGetterParams } from 'ag-grid-community';
```

Data Grid/Explorer Widgets

This section lists all changes related to the Data Grid and Data Explorer widget.

New Index Bar Localization

It is now possible to localize the message "row X of Y" at the bottom of the Data Grid widget, which gives the index of the row selected.


Row 2 of 6 Moyenne: 25 Compte: 3 Min: 25 Max: 25 Somme: 75

New Advanced Filters (EXPERIMENTAL)

In the tab *Advanced* of the Data Explorer and Data Grid widgets configurator, an option now allows to *Enable advanced Filter*.

When this option is enabled, the Data Grid has a filter field where users can type complex expressions, for example:

[Id] contains "22" AND ([Duration In Hours] < 5 OR [Duration In Hours] > 10)

The user can edit the filter expression by typing the expression in the filter field or using the graphical filter builder.

Several limitations of the advanced filters are to be considered in 4.3.0:

- For *Duration* and *LocalTime* entities, advanced filters are not available.
- For *LocalDateTime* and *Instant* entities, advanced filters are not available and, in a simple filter, the value will be the selected date with a time of *00:00*.

- -

Workspace and Scenario Widgets

This section lists all changes related to the Scenario List and Scenario Timeline widget.

No changes

- -

Issue Widgets

This section lists all changes related to the Issue List and Issue Details widgets.

No changes

- -

Job Widgets

This section lists all changes related to the New Job Button, Job List, and Job Details widgets as well as the Jobs menu.

No changes

- -

Button Widget

This section lists all changes related to the Navigation Button widget.

No changes

Chart Widgets

This section lists all changes related to the Chart widget.

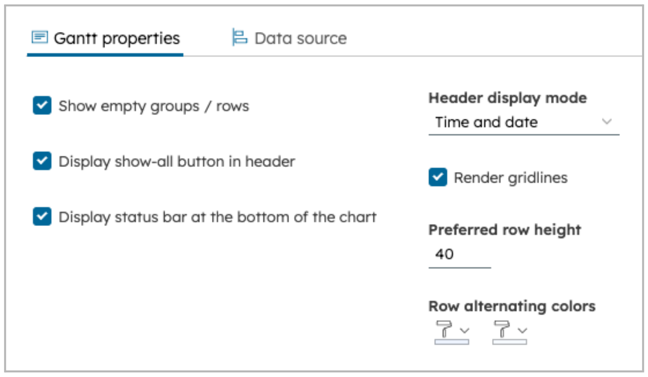
No changes

Gantt Chart Widget

This section lists all changes related to the Gantt Chart widget.

New Selection Feature

Just like for the Data Grid/Explorer widgets, it is now possible to select events and resources from the widget, on the *None* / *Select* / *Highlight* basis.



Improved Display Options

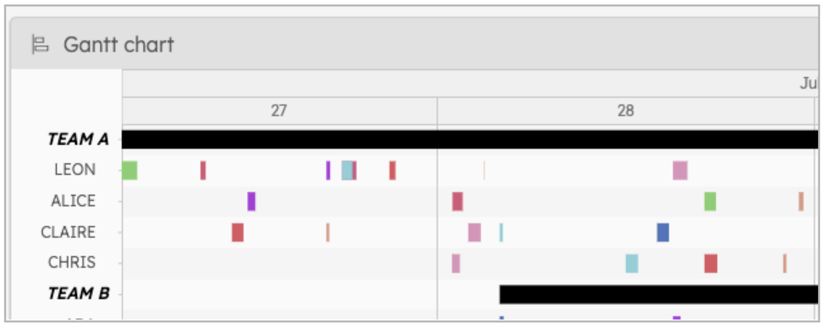
New options for row height and alternating colors are now available.

Also, the Gantt Chart widget now uses the same Time axis options as the Chart widget.

Finally, the zoom level and scroll position are now saved after leaving the view or dashboard.

Improved Controller for Event Labels

Using a custom controller, it is now possible to set a renderer for the Event label.



KPI Widget

This section lists all changes related to the KPI widget.

No changes

Filter Widget

This section lists all changes related to the Filter widget.

No changes

Composite Widget

This section lists all changes related to the Composite widget.

No changes

Map Widget

This section lists all changes related to the Map widget.

Improved Map Widget Controller

In the Map widget, the following custom controller method now allows overriding the marker series tooltip.

```
Unset
     getMarkerPopupHtmlContent = (markerInfo: MarkerInfo): string => {
          return `

              This <i>${markerInfo.tag}</i> is located at

              <ul>
                  <li><b>Latitude</b>: ${markerInfo.position.latitude}</li>
                  <li><b>Longitude</b>: ${markerInfo.position.longitude}</li>
              </ul>
          `
      }
```

Calendar Widget

This section lists all changes related to the Calendar widget.

No changes

- -

Rich Text Widget

This section lists all changes related to the Rich Text widget.

No changes

- -

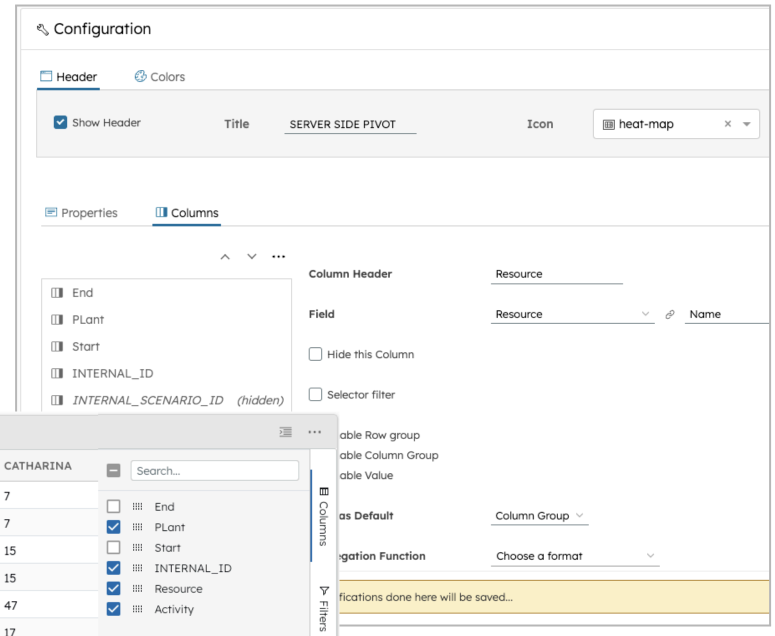
Pivot Table Widget

This section lists all changes related to the Pivot Table widget.

Improved Pivot Table Widget

DB Gene 4.3.0 introduces a new version of the Pivot Table widget. The former version has been renamed Pivot Table (Legacy) and might be removed in a future release.

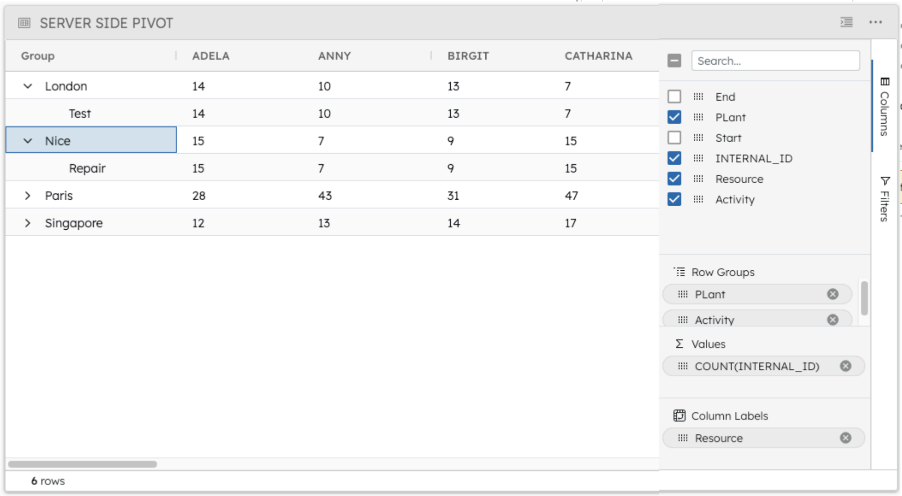
This new version allows using a server-side data source.



☑ Show Header
Title: SERVER SIDE PIVOT
Icon: heat-map

End
PLant
Start
INTERNAL_ID
INTERNAL_SCENARIO_ID (hidden)

Column Header: Resource
Field: Resource — Name
☐ Hide this Column
☐ Selector filter
able Row group
able Column Group
able Value
as Default: Column Group
gation Function: Choose a format
fications done here will be saved...


SERVER SIDE PIVOT

Group	ADELA	ANNY	BIRGIT	CATHARINA
London	14	10	13	7
Test	14	10	13	7
Nice	15	7	9	15
Repair	15	7	9	15
Paris	28	43	31	47
Singapore	12	13	14	17

6 rows

☐ End
☑ PLant
☐ Start
☑ INTERNAL_ID
☑ Resource
☑ Activity

Row Groups: PLant, Activity
Values: COUNT(INTERNAL_ID)
Column Labels: Resource

It leverages the AG Grid enterprise Pivot feature and can handle large scenarios.

- -

Code Editor Widget

This section lists all changes related to the Code Editor widget.

New Code Editor Widget

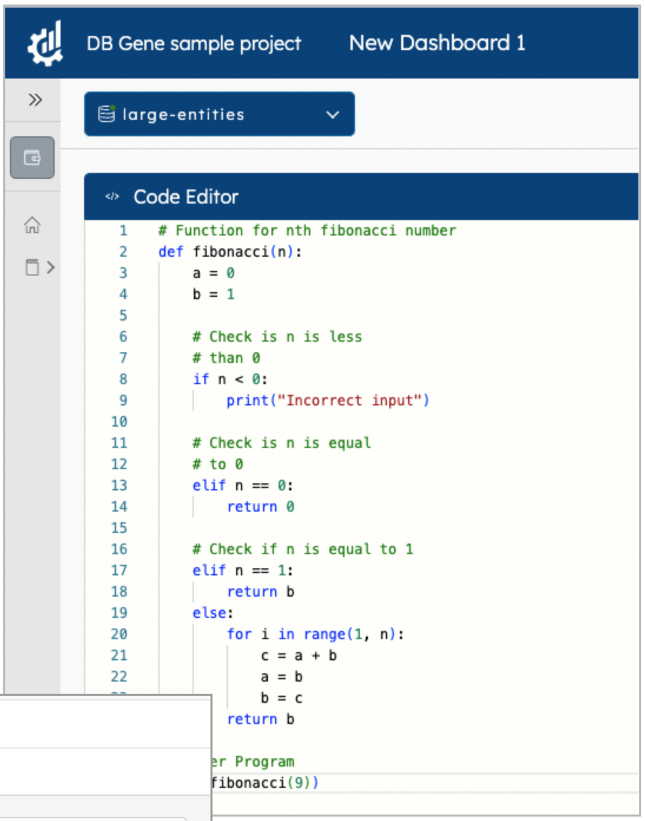
The new Code Editor widget is now available as a beta feature.

It allows a user to edit the content of a `GeneParameter` in the selected scenario.

The name of the parameter is defined in the widget configuration.

The row is selected on its `name` column, and its `value` column is used.

The widget can also be associated with a scripted task.



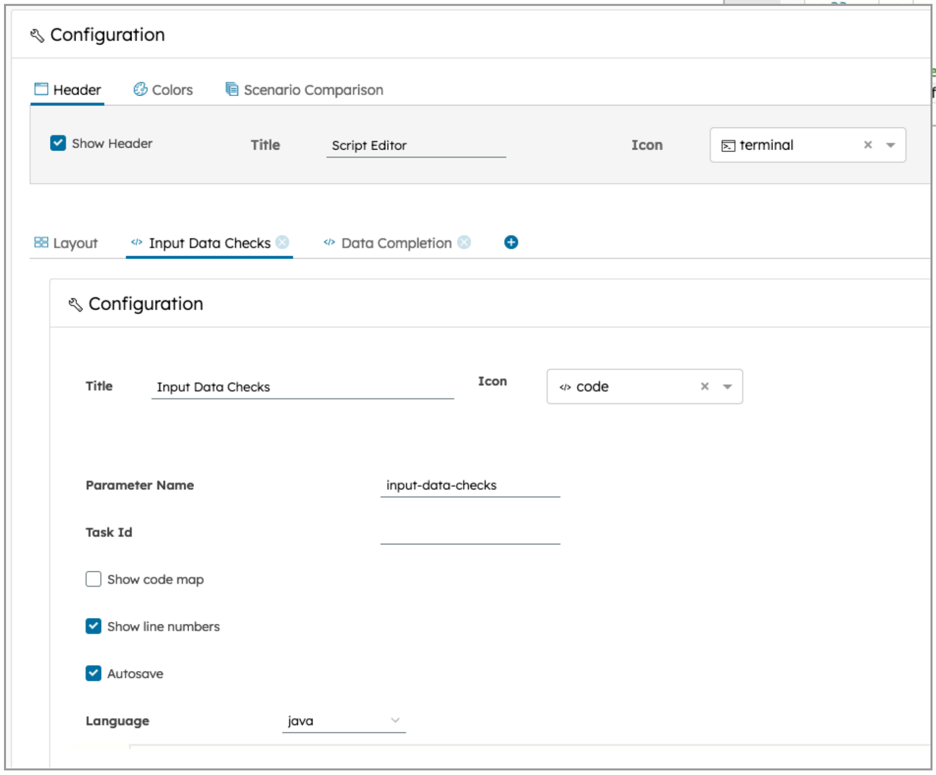
```
# Function for nth fibonacci number
def fibonacci(n):
    a = 0
    b = 1

    # Check is n is less
    # than 0
    if n < 0:
        print("Incorrect input")

    # Check is n is equal
    # to 0
    elif n == 0:
        return 0

    # Check if n is equal to 1
    elif n == 1:
        return b
    else:
        for i in range(1, n):
            c = a + b
            a = b
            b = c
        return b

er Program
fibonacci(9))
```



✓ Show Header	Title: Script Editor	Icon: terminal	

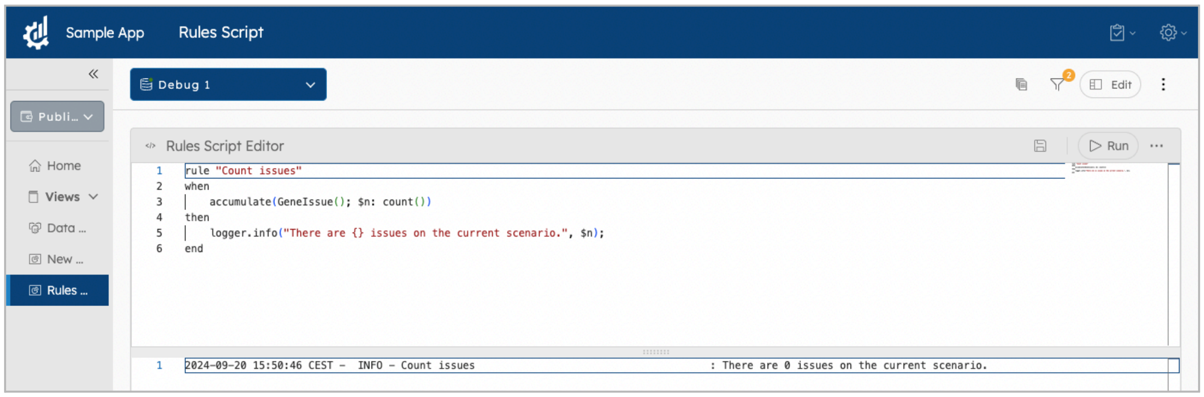
Title	Input Data Checks
Icon	code
Parameter Name	input-data-checks
Task Id	
☐ Show code map	
✓ Show line numbers	
✓ Autosave	
Language	java

If so, the widget displays a *Run* button which, when clicked, invokes the task with two input parameters:

- one named *scenario*, which receives the ID of the current scenario, and
- one named *scriptName*, which receives the name of the `GeneParameter`.

New Rule Script Editor Widget

A generic task and a generic routine now allow calling a rule engine on a scenario. The added task is identified by the id `ExperimentalExecuteRulesetOnScenarioTask` and is compatible with the new Rule Script Editor widget.



```
rule "Count issues"
when
    accumulate(GeneIssue(); $n: count())
then
    logger.info("There are {} issues on the current scenario.", $n);
end
```

```
2024-09-20 15:50:46 CEST -  INFO - Count issues                               : There are 0 issues on the current scenario.
```

If the user configures the widget to execute this task by clicking on the *Run* button, it is launched with the parameter name containing the script written in the Editor and the current scenario.

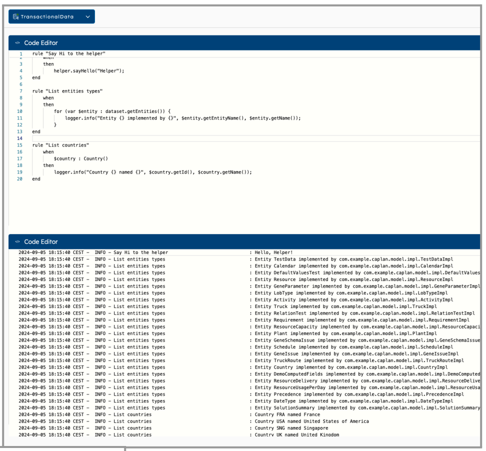
When launched, the task:

- calls the routine,
- retrieves the ruleset script from the parameter name as the input parameter,
- compiles the ruleset, and
- runs it on the current scenario.

The logs of the execution or failed compilation are stored in another parameter. For example, if the script is stored in a parameter named `checker-editor-script`, the last run logs will be stored in a parameter named `checker-editor-script-logs`.

The logs are visible in the task output.

If an error occurs during the routine execution, the task is set in an alerting state with a clear message.

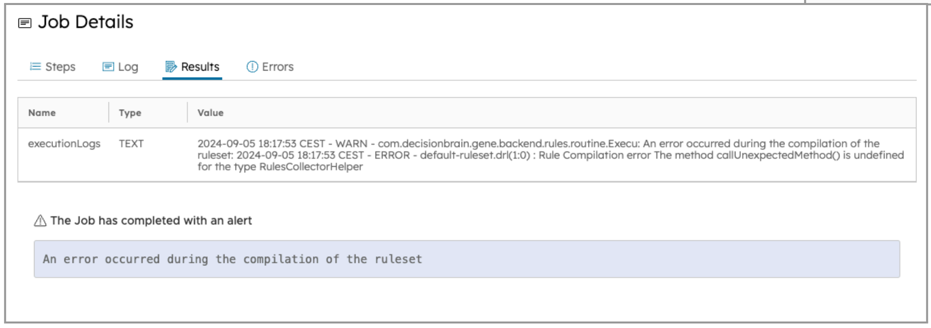


```
rule "Say Hi to the helper"
    when
    then
        helper.sayHello("Helper");
end

rule "List entities types"
    when
    then
        for (var $entity : dataset.getEntities()) {
            logger.info("Entity {} implemented by {}", $entity.getEntityName(), $entity.getName());
        }
end

rule "List countries"
    when
        $country : Country()
    then
        logger.info("Country {} named {}", $country.getId(), $country.getName());
end
```

```
2024-09-05 18:15:40 CEST -  INFO - Say Hi to the helper                : Hello, Helper!
2024-09-05 18:15:40 CEST -  INFO - List entities types                 : Entity TestData implemented by com.example.caplan.model.impl.TestDataImpl
2024-09-05 18:15:40 CEST -  INFO - List entities types                 : Entity Calendar implemented by com.example.caplan.model.impl.CalendarImpl
2024-09-05 18:15:40 CEST -  INFO - List entities types                 : Entity DefaultValuesTest implemented by com.example.caplan.model.impl.DefaultValues
2024-09-05 18:15:40 CEST -  INFO - List entities types                 : Entity Resource implemented by com.example.caplan.model.impl.ResourceImpl
2024-09-05 18:15:40 CEST -  INFO - List entities types                 : Entity GeneParameter implemented by com.example.caplan.model.impl.GeneParameterImpl
2024-09-05 18:15:40 CEST -  INFO - List entities types                 : Entity LobType implemented by com.example.caplan.model.impl.LobTypeImpl
2024-09-05 18:15:40 CEST -  INFO - List entities types                 : Entity Activity implemented by com.example.caplan.model.impl.ActivityImpl
2024-09-05 18:15:40 CEST -  INFO - List entities types                 : Entity Truck implemented by com.example.caplan.model.impl.TruckImpl
2024-09-05 18:15:40 CEST -  INFO - List entities types                 : Entity RelationTest implemented by com.example.caplan.model.impl.RelationTestImpl
2024-09-05 18:15:40 CEST -  INFO - List entities types                 : Entity Requirement implemented by com.example.caplan.model.impl.RequirementImpl
2024-09-05 18:15:40 CEST -  INFO - List entities types                 : Entity ResourceCapacity implemented by com.example.caplan.model.impl.ResourceCapaci
2024-09-05 18:15:40 CEST -  INFO - List entities types                 : Entity Plant implemented by com.example.caplan.model.impl.PlantImpl
2024-09-05 18:15:40 CEST -  INFO - List entities types                 : Entity GeneSchemaIssue implemented by com.example.caplan.model.impl.GeneSchemaIssue
2024-09-05 18:15:40 CEST -  INFO - List entities types                 : Entity Schedule implemented by com.example.caplan.model.impl.ScheduleImpl
2024-09-05 18:15:40 CEST -  INFO - List entities types                 : Entity GeneIssue implemented by com.example.caplan.model.impl.GeneIssueImpl
2024-09-05 18:15:40 CEST -  INFO - List entities types                 : Entity TruckRoute implemented by com.example.caplan.model.impl.TruckRouteImpl
2024-09-05 18:15:40 CEST -  INFO - List entities types                 : Entity Country implemented by com.example.caplan.model.impl.CountryImpl
2024-09-05 18:15:40 CEST -  INFO - List entities types                 : Entity DemoComputedFields implemented by com.example.caplan.model.impl.DemoComputed
2024-09-05 18:15:40 CEST -  INFO - List entities types                 : Entity ResourceDelivery implemented by com.example.caplan.model.impl.ResourceDelive
2024-09-05 18:15:40 CEST -  INFO - List entities types                 : Entity ResourceUsagePerDay implemented by com.example.caplan.model.impl.ResourceUsa
2024-09-05 18:15:40 CEST -  INFO - List entities types                 : Entity Precedence implemented by com.example.caplan.model.impl.PrecedenceImpl
2024-09-05 18:15:40 CEST -  INFO - List entities types                 : Entity DateType implemented by com.example.caplan.model.impl.DateTypeImpl
2024-09-05 18:15:40 CEST -  INFO - List entities types                 : Entity SolutionSummary implemented by com.example.caplan.model.impl.SolutionSummary
2024-09-05 18:15:40 CEST -  INFO - List countries                      : Country FRA named France
2024-09-05 18:15:40 CEST -  INFO - List countries                      : Country USA named United States of America
2024-09-05 18:15:40 CEST -  INFO - List countries                      : Country SNG named Singapore
2024-09-05 18:15:40 CEST -  INFO - List countries                      : Country UK named United Kingdom
```



Name	Type	Value
executionLogs	TEXT	2024-09-05 18:17:53 CEST - WARN - com.decisionbrain.gene.backend.rules.routine.Execu: An error occurred during the compilation of the ruleset: 2024-09-05 18:17:53 CEST - ERROR - default-ruleset.drl(1:0) : Rule Compilation error The method callUnexpectedMethod() is undefined for the type RulesCollectorHelper

The Job has completed with an alert

```
An error occurred during the compilation of the ruleset
```

Deprecated Features and APIs Scheduled for Removal in 4.4.0

This section lists all removals that are planned to come in DB Gene 4.4.0.

No changes