What’s New

Highlighting of changes and what this means for your project.

Knowledge session

Blueriq

Topics

During this What's **New** we'll be discussing the following topics.

Introduction

- History
- Categories

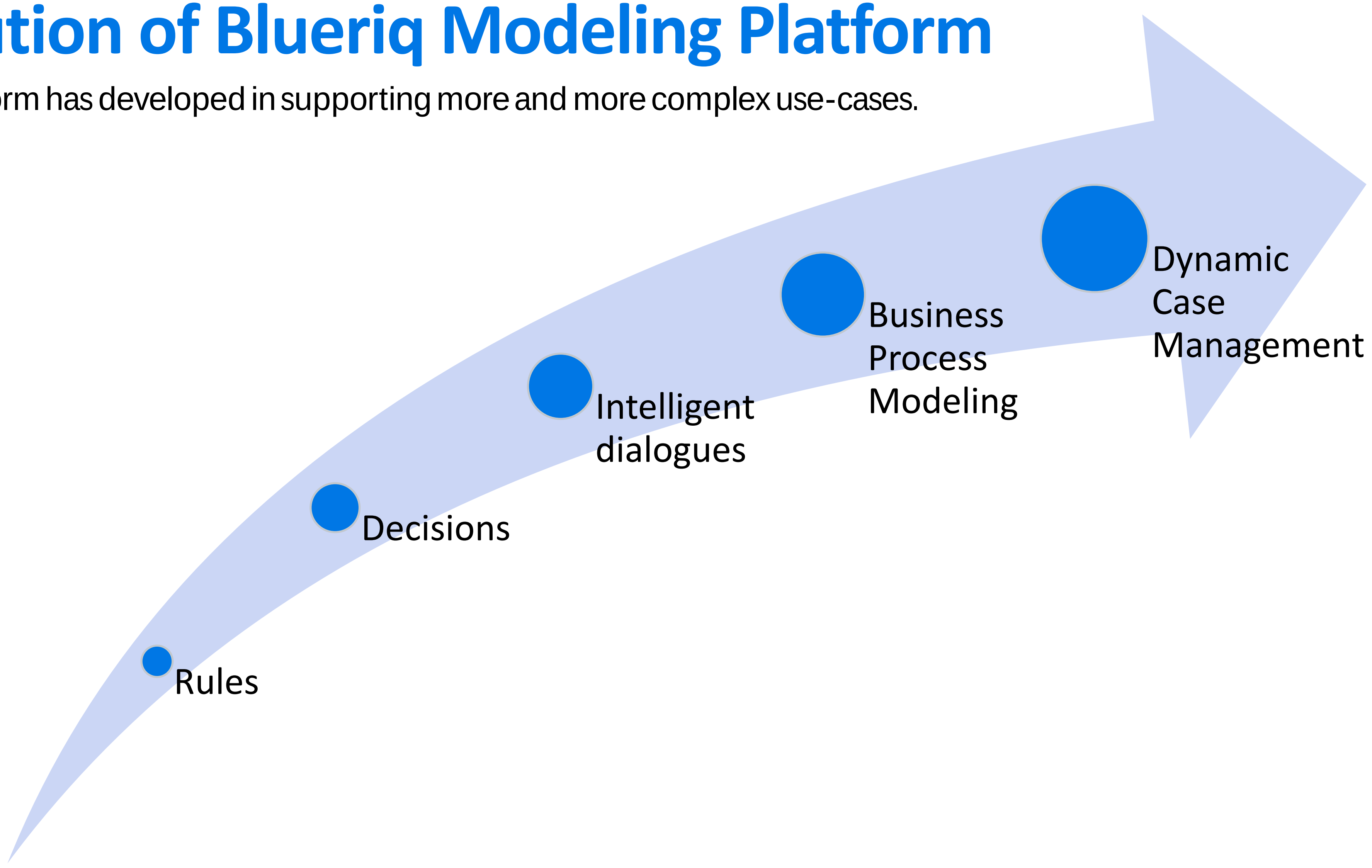
Releases

- Blueriq 10
- Blueriq 11
- Blueriq 12
- Blueriq 13

Improvement tips

Evolution of Blueriq Modeling Platform

Our platform has developed in supporting more and more complex use-cases.



Categories

Though not official sorted into categories by the product development team, we can distinguish certain traits of these changes.

Keeping up with technology

Ensuring we keep up to date with technological developments.

Time to market

Increasing the speed of development and delivery.

Quality

Having means to determine what the state of the model and resulting application is.

Efficiency

Needing less time to do the activities related to the Blueriq products.

Usability

Ensuring a user-friendly interaction with Blueriq products.

Business Engineering

Offering an ever-growing toolset to professionals in order to solve business problems by creating a knowledge model that can be executed.

Integrating Blueriq

Having means to integrate Blueriq into a landscape.

Separation of concerns

Ensuring that functionality, responsibility and implementation of components are separated.

Easy updates

Lowering the bar to adopt new versions and updates.

For each version we'll be discussing the following topics.

Release version

Studio

- New features
- Enhancements
- Usability
- Unit testing

Runtime

New products

Front-end

Removed

Blueriq 11

Studio

- New features
- Enhancements
- Usability
- Unit testing

Runtime

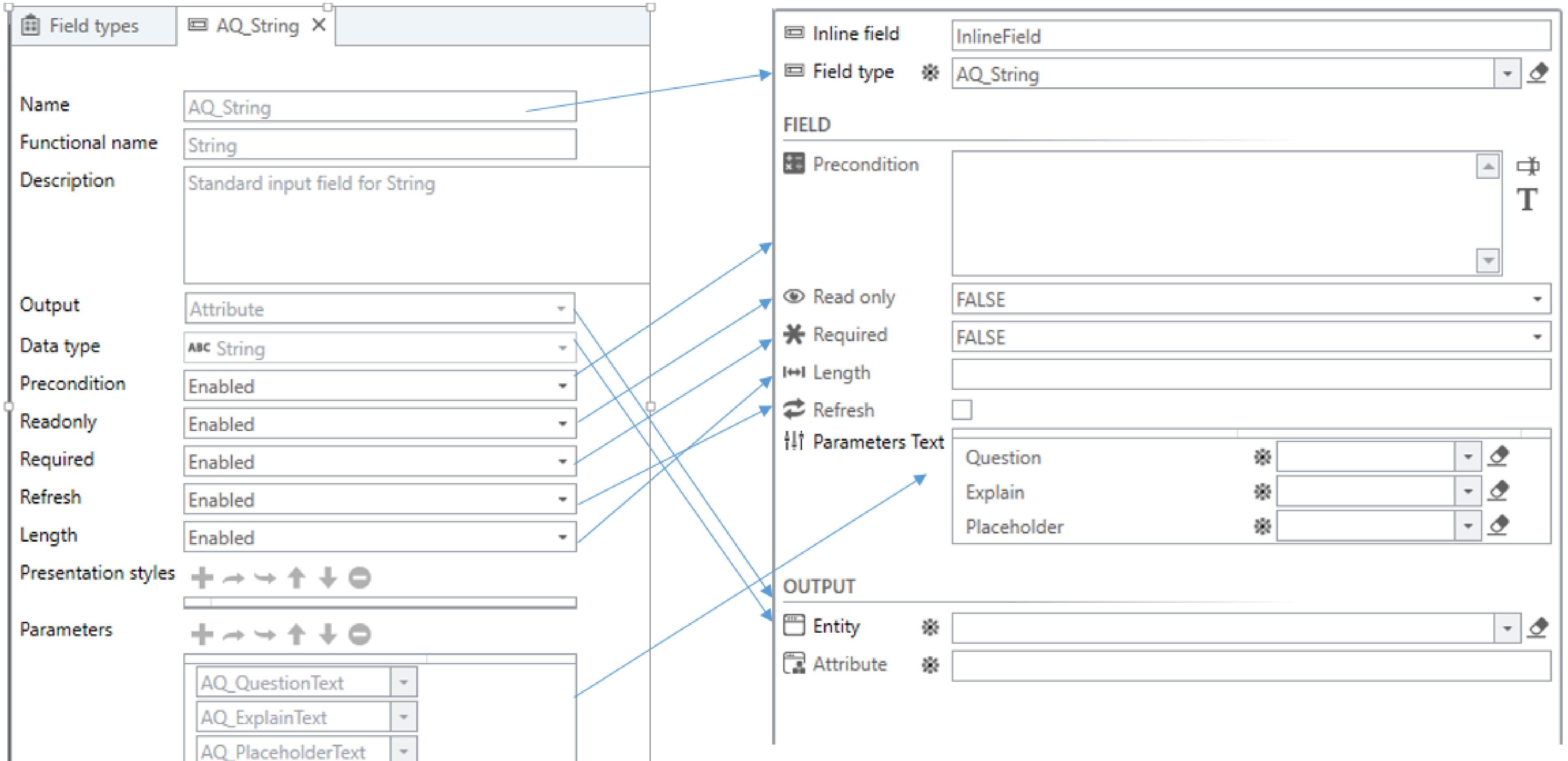
New products

Front-end

Removed

Fields

It is now possible to create fields that are used as a way of implementing attributes on a page. Fields act as a **wrapper** for attributes and separate the implementation on the page from the domain.



Field types | AQ_String

Name	AQ_String
Functional name	String
Description	Standard input field for String
Output	Attribute
Data type	ABC String
Precondition	Enabled
Readonly	Enabled
Required	Enabled
Refresh	Enabled
Length	Enabled
Presentation styles	
Parameters	AQ_QuestionText AQ_ExplainText AQ_PlaceholderText

Inline field	InlineField
Field type	AQ_String

FIELD

Precondition	
Read only	FALSE
Required	FALSE
Length	
Refresh	☐
Parameters Text	Question Explain Placeholder

OUTPUT

Entity	
Attribute	

11.3

Declarative relations

Relations can now be created in a declarative manner based on a default value rather then solely using service calls.

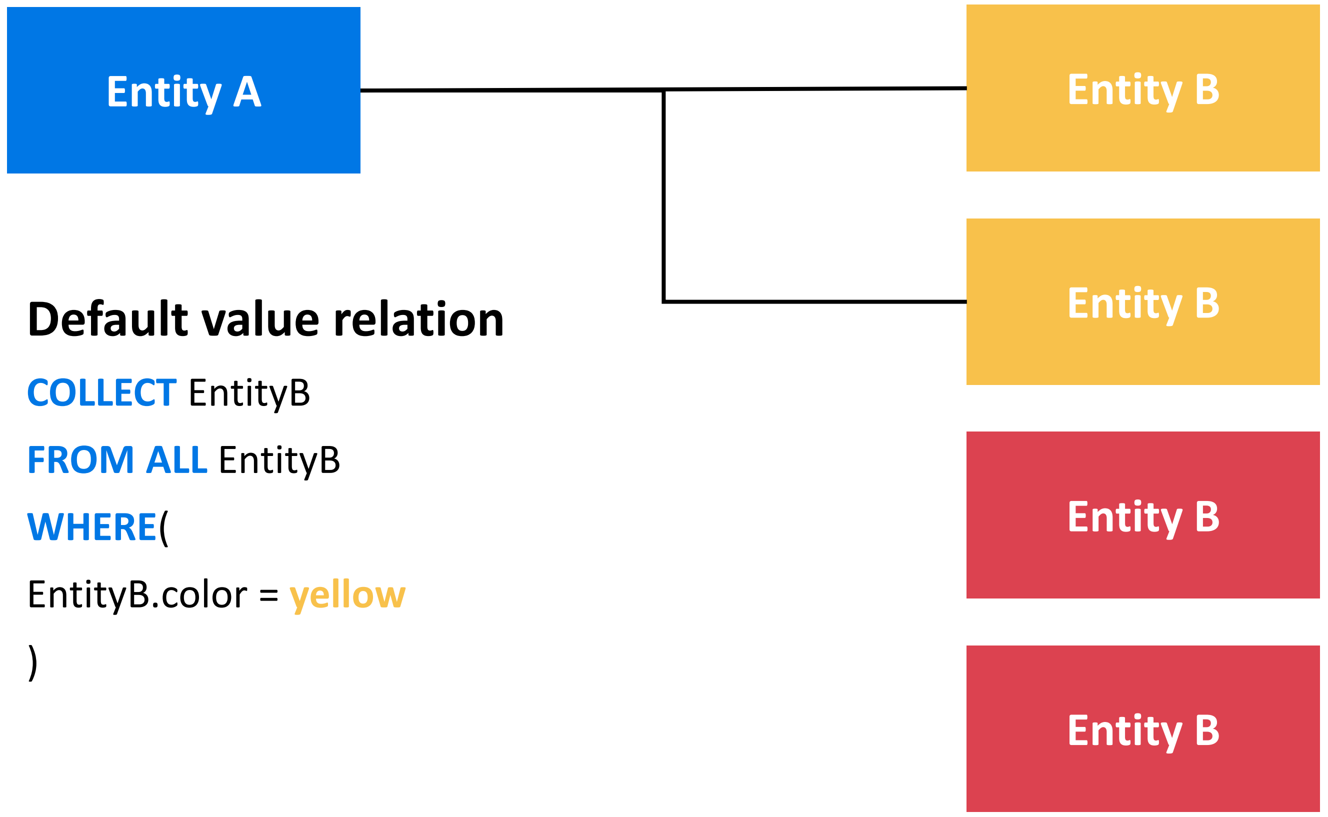
Inference engine

This engine will take care of keeping the statement up to date.

Instances will be **linked automatically** according to the expression.

Pro tip!

<u>**Never**</u> use the default value for the reverse relation. This will lead to performance issues.



Default value relation

```
COLLECT EntityB
FROM ALL EntityB
WHERE(
EntityB.color = yellow
)
```

11.5

Error context clearing

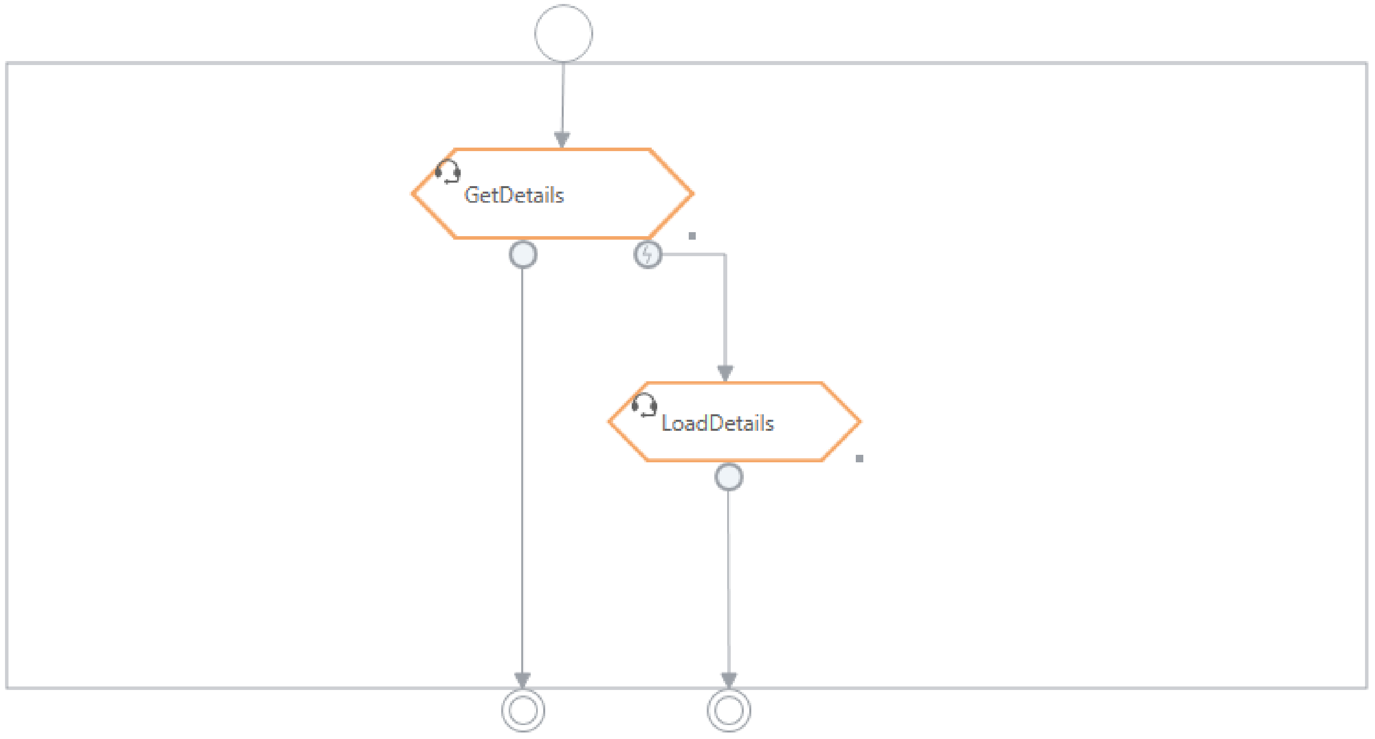
Before

LoadDetails would fail no matter what. This is because of the **error context** not being cleared in the Runtime.

Now

When the exception node is modeled **then** errors on the context are cleared.

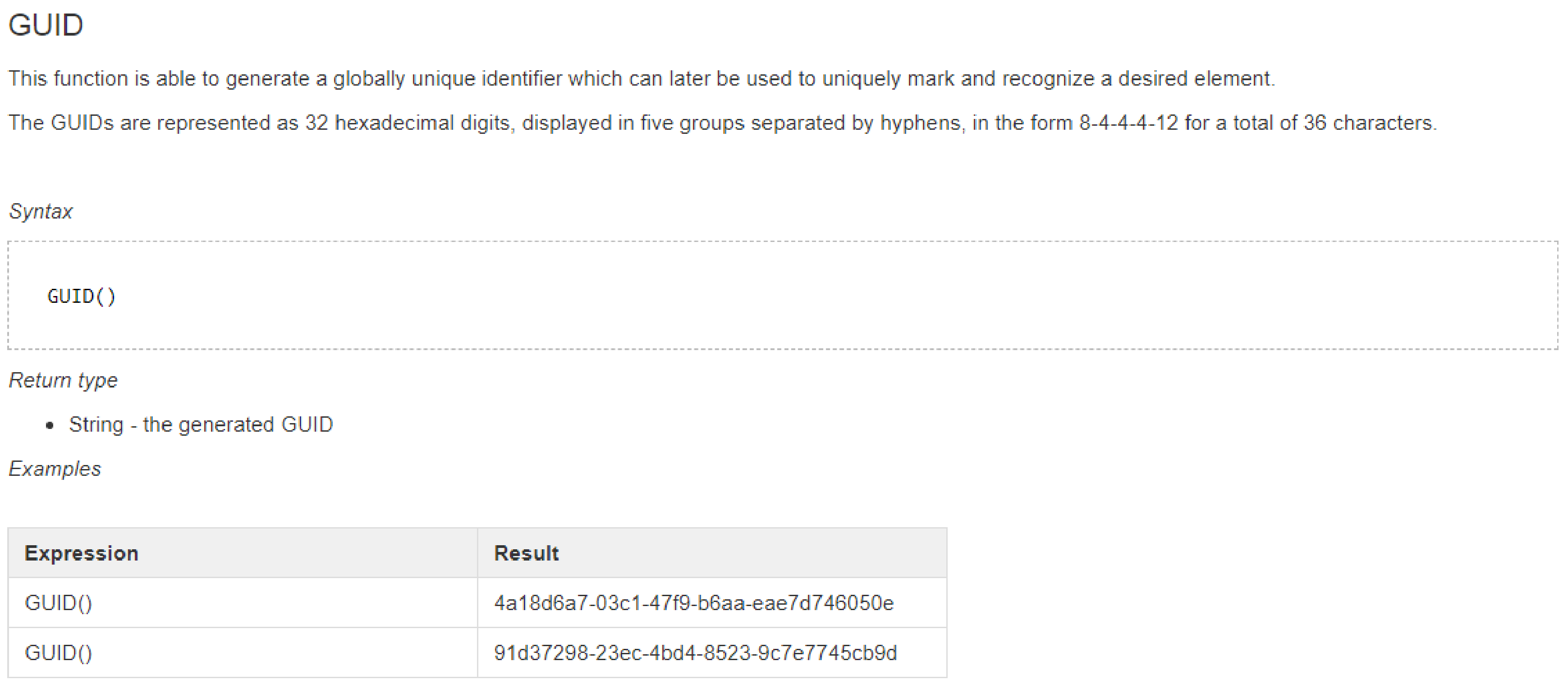
This prevents Blueriq from **always taking the exception node of the next servicecall** even if there is nothing wrong with that call itself.



11.0

GUID function

It is now possible to use GUID() to create one using **32 hexadecimal digits**. There is no longer a need for custom services to enable this functionality.



GUID

This function is able to generate a globally unique identifier which can later be used to uniquely mark and recognize a desired element.

The GUIDs are represented as 32 hexadecimal digits, displayed in five groups separated by hyphens, in the form 8-4-4-4-12 for a total of 36 characters.

Syntax

```
GUID()
```

Return type

- String - the generated GUID

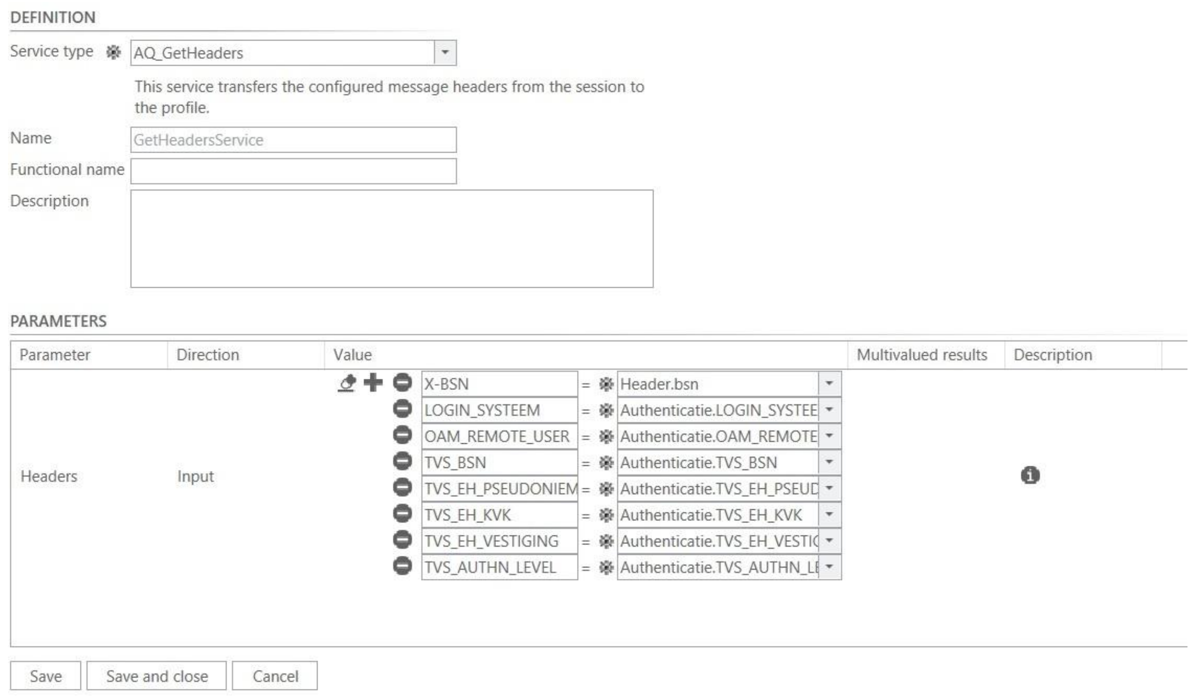
Examples

Expression	Result
GUID()	4a18d6a7-03c1-47f9-b6aa-eae7d746050e
GUID()	91d37298-23ec-4bd4-8523-9c7e7745cb9d

11.1

Getting and setting HTTP headers

It was always a hassle, trying to handle HTTP headers in a model. **AQ_GetHeaders** enables the handling of headers in the model.



DEFINITION

Service type	AQ_GetHeaders
	This service transfers the configured message headers from the session to the profile.
Name	GetHeadersService
Functional name	
Description	

PARAMETERS

Parameter	Direction	Value	Multivalued results	Description

Headers	Input	X-BSN = Header.bsn
		LOGIN_SYSTEEM = Authenticatie.LOGIN_SYSTEE
		OAM_REMOTE_USER = Authenticatie.OAM_REMOTE
		TVS_BSN = Authenticatie.TVS_BSN
		TVS_EH_PSEUDONIEM = Authenticatie.TVS_EH_PSEUD
		TVS_EH_KVK = Authenticatie.TVS_EH_KVK
		TVS_EH_VESTIGING = Authenticatie.TVS_EH_VESTIC
		TVS_AUTHN_LEVEL = Authenticatie.TVS_AUTHN_LE		

Auto assign

Using custom HTTP header setup, Blueriq can also **auto assign** received headers to outgoing webservice calls.

Pro tip!

When setup, Blueriq will **log** whether it can **find** the right headers in the incoming message.

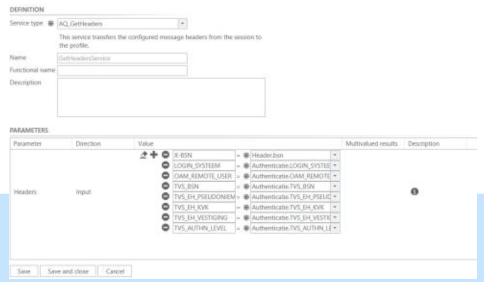
Use this to your advantage!

11.6

Getting and setting HTTP headers

It was always a hassle, trying to handle HTTP headers in a model. **AQ_GetHeaders** enables the handling of headers in the model.

```
blueriq.user.headers=
header1,header2,header3
```



Service type: AQ_GetHeaders

This service transfers the configured message headers from the session to the profile.

Name: GetHeadersService

Parameter	Direction	Value		Multivalued results	Description
Headers	Input	X-BSN	Header.bsn		
		LOGIN_SYSTEEM	Authenticatie.LOGIN_SYSTEE		
		OAM_REMOTE_USER	Authenticatie.OAM_REMOTE		
		TVS_BSN	Authenticatie.TVS_BSN		
		TVS_EH_PSEUDONIEM	Authenticatie.TVS_EH_PSEUD		
		TVS_EH_KVK	Authenticatie.TVS_EH_KVK		
		TVS_EH_VESTIGING	Authenticatie.TVS_EH_VESTIG		
		TVS_AUTHN_LEVEL	Authenticatie.TVS_AUTHN_LI		

```
blueriq.connection.headers=
header1,header2,header3
```

Determine HTTP header to store

Use AQ_GetHeaders to map to the profile

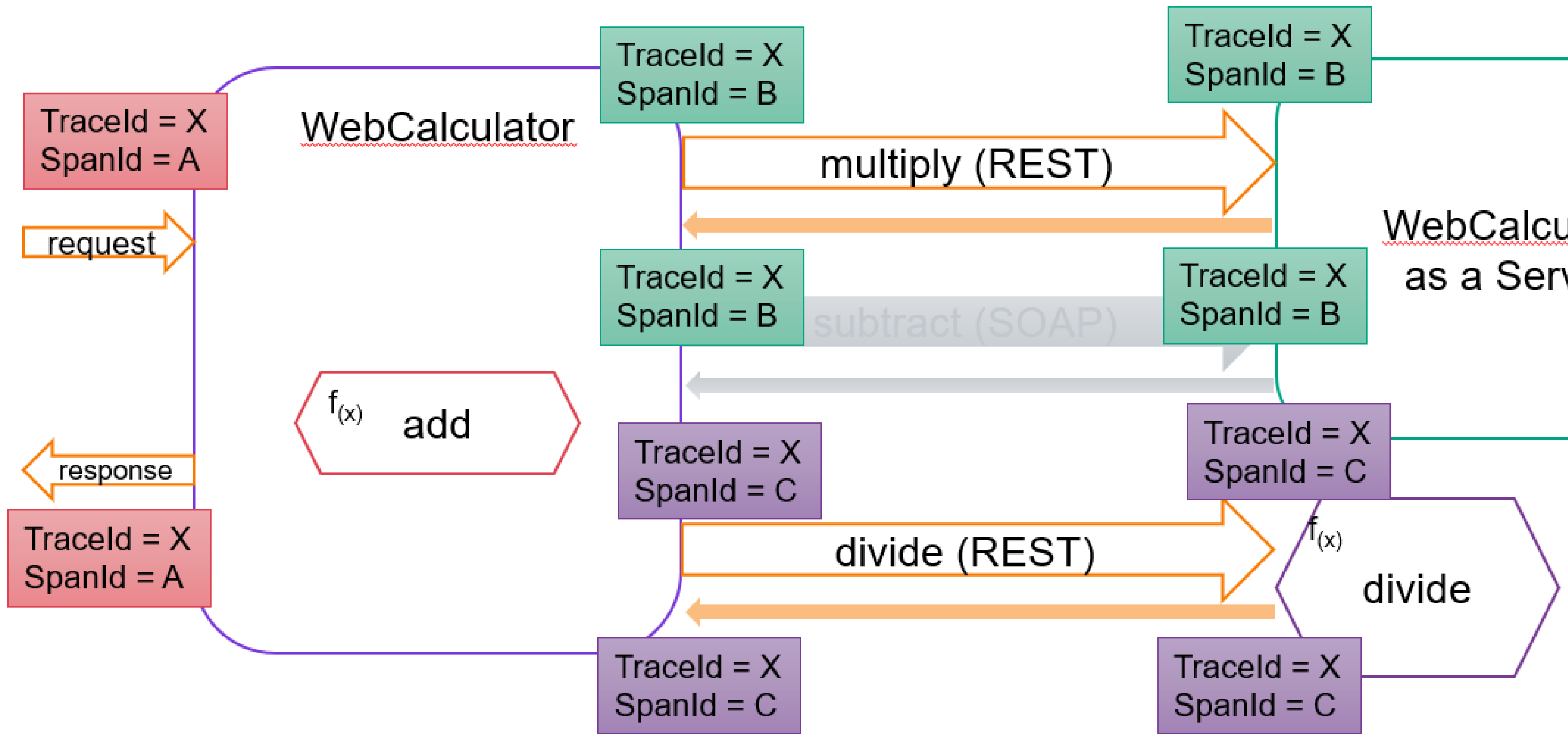
(optional)

Configure Blueriq to use HTTP header in outgoing traffic

11.6

Log correlation

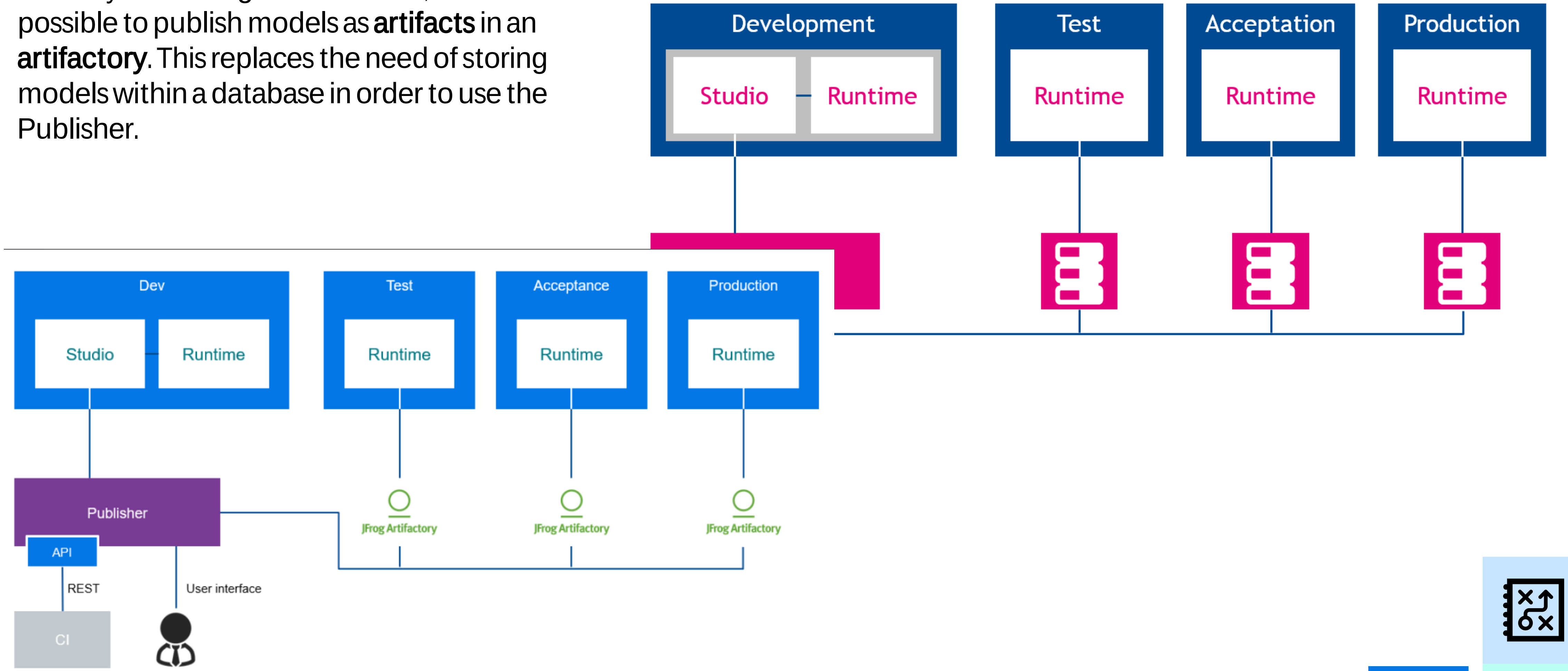
Blueriq now uses the **Brave Header** syntax for passing **trace headers** to the next service it calls or when it receives such a header itself. This is can be used by 3rd party tooling (Splunk of ELK) to correlate these ids throughout the landscape.



11.8

Publisher fine-tuning

In case you're using the **Publisher**, it is now possible to publish models as **artifacts** in an **artifactory**. This replaces the need of storing models within a database in order to use the Publisher.

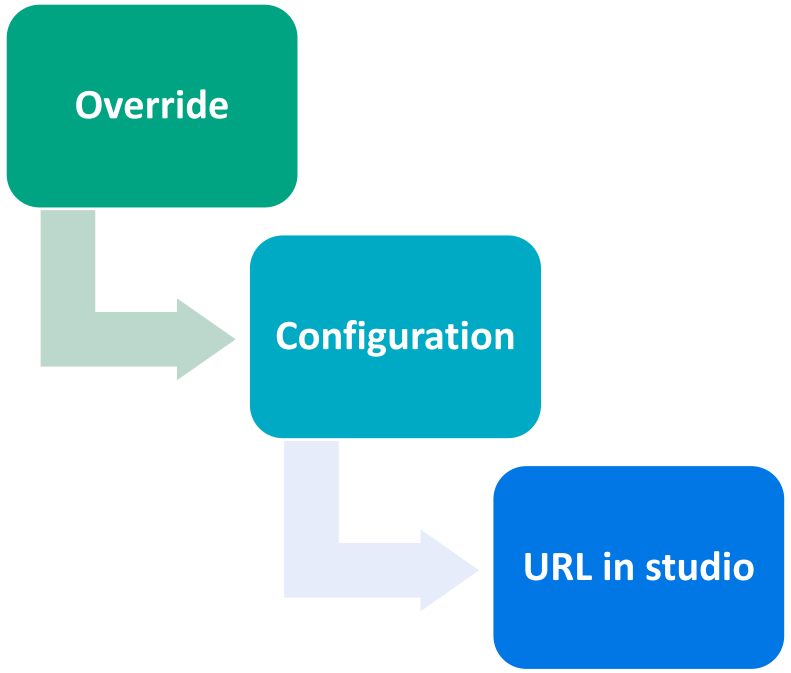


11.0

URL and connection overrides

Cleaning up your connections is possible with the connection override feature. A specific connection can be used by referencing it in the model.

Order in determining connection



Pro tip!

If you use multiple endpoints of one connection, you should use the override function.

Cleaning up the property file

Servicecall add.url = connection/add
Servicecall subtract.url = connection/subtract
Servicecall multiply.url= connection/multiply

Connection.url = connection

Place /add, /subtract. /multiply in the service definition in the model

11.0

Blueriq 11

Studio

- New features
- Enhancements
- Usability
- Unit testing

Runtime

New products

Front-end

Removed

Blueriq Publisher

Although the Blueriq Publisher exists for a longer time, it can't hurt to shine a light on its functionality once again.

Publisher

Propagating models

The publisher can distribute versions of a model to a **database** or **artifactory**.

Runtime extension

The Blueriq Runtime with the Publisher profile can **detect new versions** of a model and execute them right away.

CI/CD integration

The Publisher is designed to be **integrated** into **build pipelines**. Automate the **publish** and **decrease** your **time to market** even more.

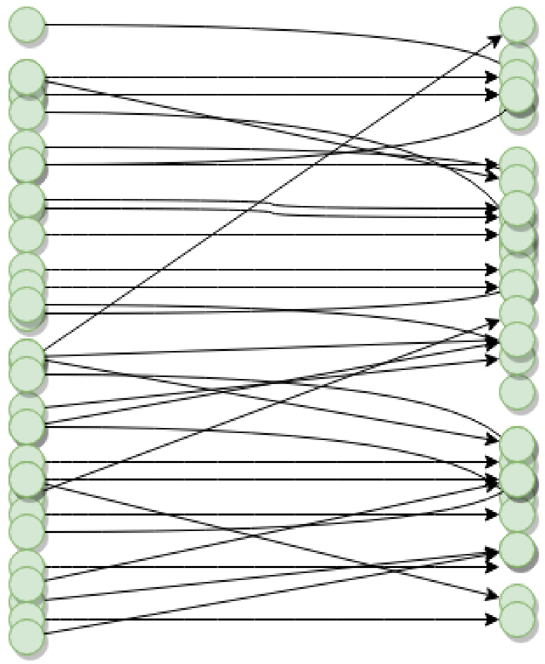
11.0

Blueriq Model Mapper

It is now possible to test the mappings within the model by using the model mapper. This ensure quick feedback on the functioning of the mapping without needing to execute the model.

Input profile

You define a **profile** that will **undergo** the mapping.



Output profile

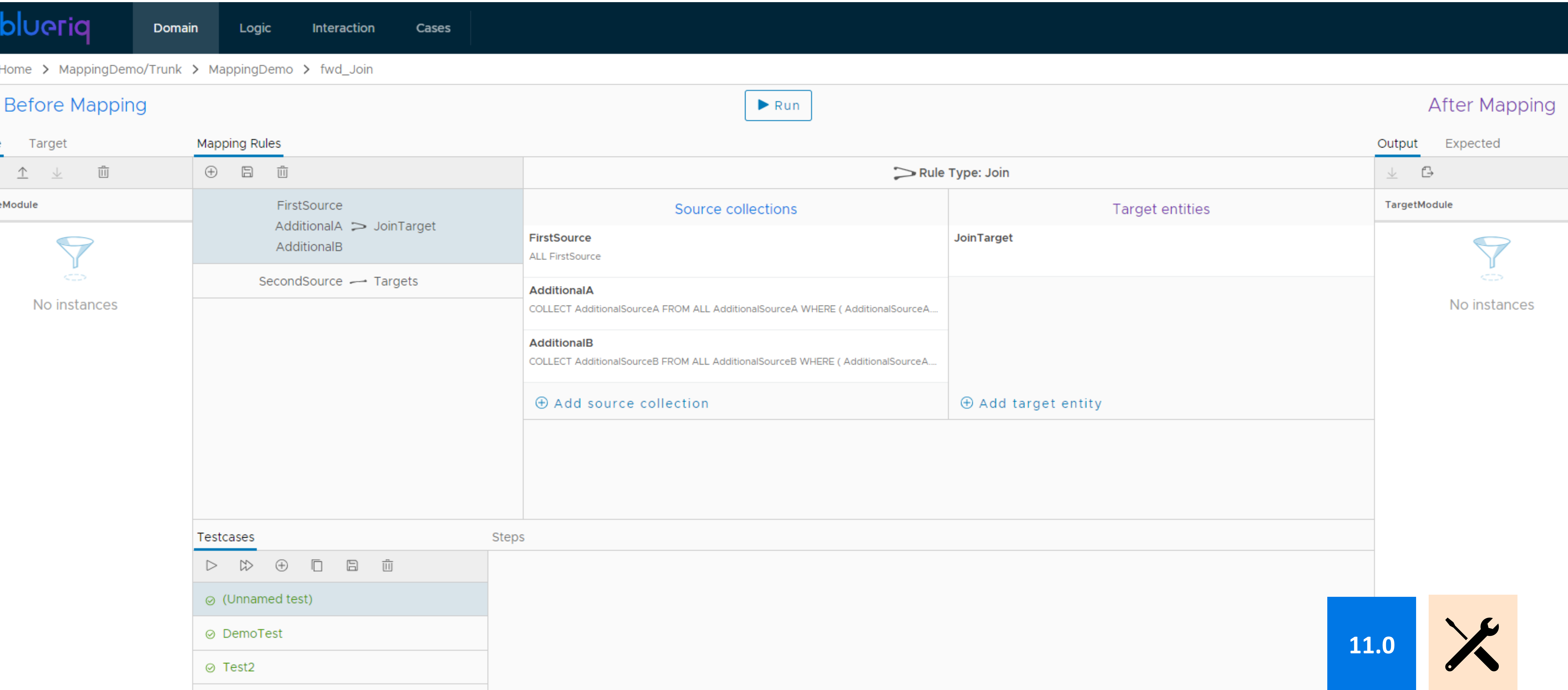
You define an **expected profile** that will be the result of the mapping process.

And **assess** whether the actual **result matches** with what you **expected**.

Model, simulate and test your data mapping

11.0

Blueriq Model Mapper



Domain
Logic
Interaction
Cases

Home > MappingDemo/Trunk > MappingDemo > fwd_Join

Before Mapping

Run

After Mapping

Target

Mapping Rules

Output

Expected

eModule

No instances

FirstSource
AdditionalA ⊃ JoinTarget
AdditionalB

SecondSource — Targets

Rule Type: Join

Source collections	Target entities
FirstSource ALL FirstSource	**JoinTarget**
AdditionalA COLLECT AdditionalSourceA FROM ALL AdditionalSourceA WHERE (AdditionalSourceA....	
AdditionalB COLLECT AdditionalSourceB FROM ALL AdditionalSourceB WHERE (AdditionalSourceA....	
⊕ Add source collection	⊕ Add target entity

TargetModule

No instances

Testcases

Steps

- (Unnamed test)
- DemoTest
- Test2

11.0

Blueriq 11

Studio

- New features
- Enhancements
- Usability
- Unit testing

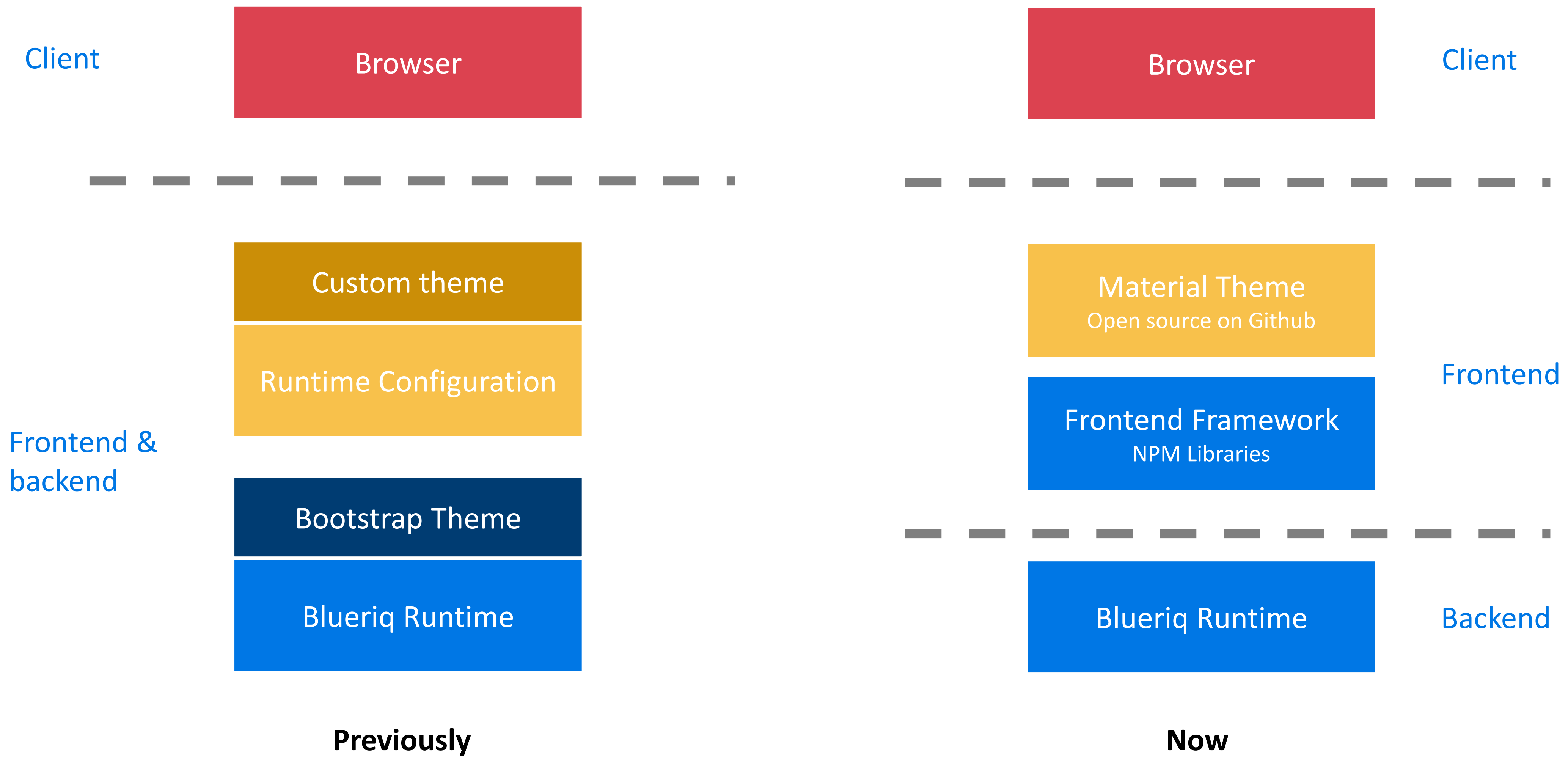
Runtime

New products

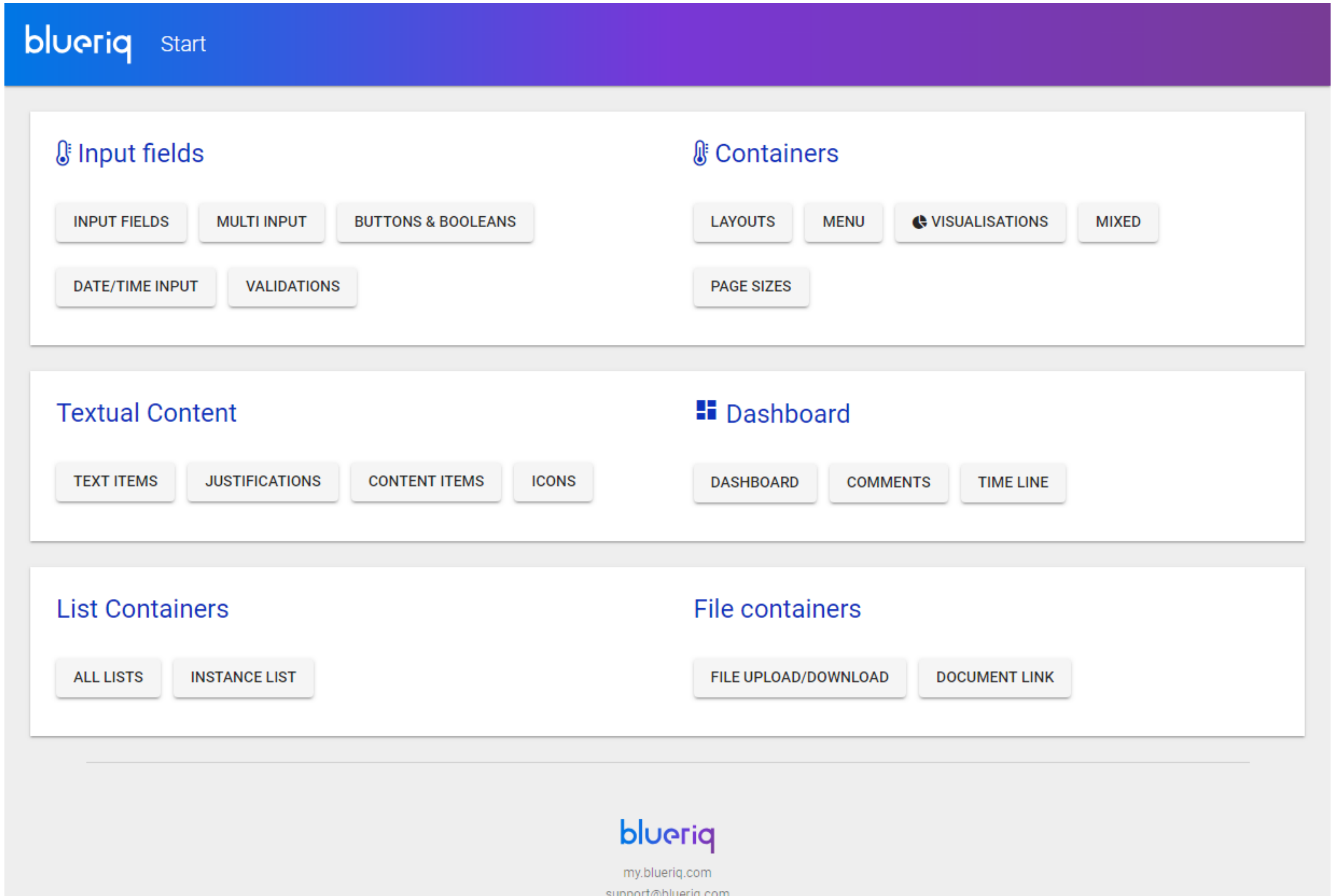
Front-end

Removed

Angular frontend framework



Angular theme



Input fields

INPUT FIELDS · MULTI INPUT · BUTTONS & BOOLEANS · DATE/TIME INPUT · VALIDATIONS

Containers

LAYOUTS · MENU · VISUALISATIONS · MIXED · PAGE SIZES

Textual Content

TEXT ITEMS · JUSTIFICATIONS · CONTENT ITEMS · ICONS

Dashboard

DASHBOARD · COMMENTS · TIME LINE

List Containers

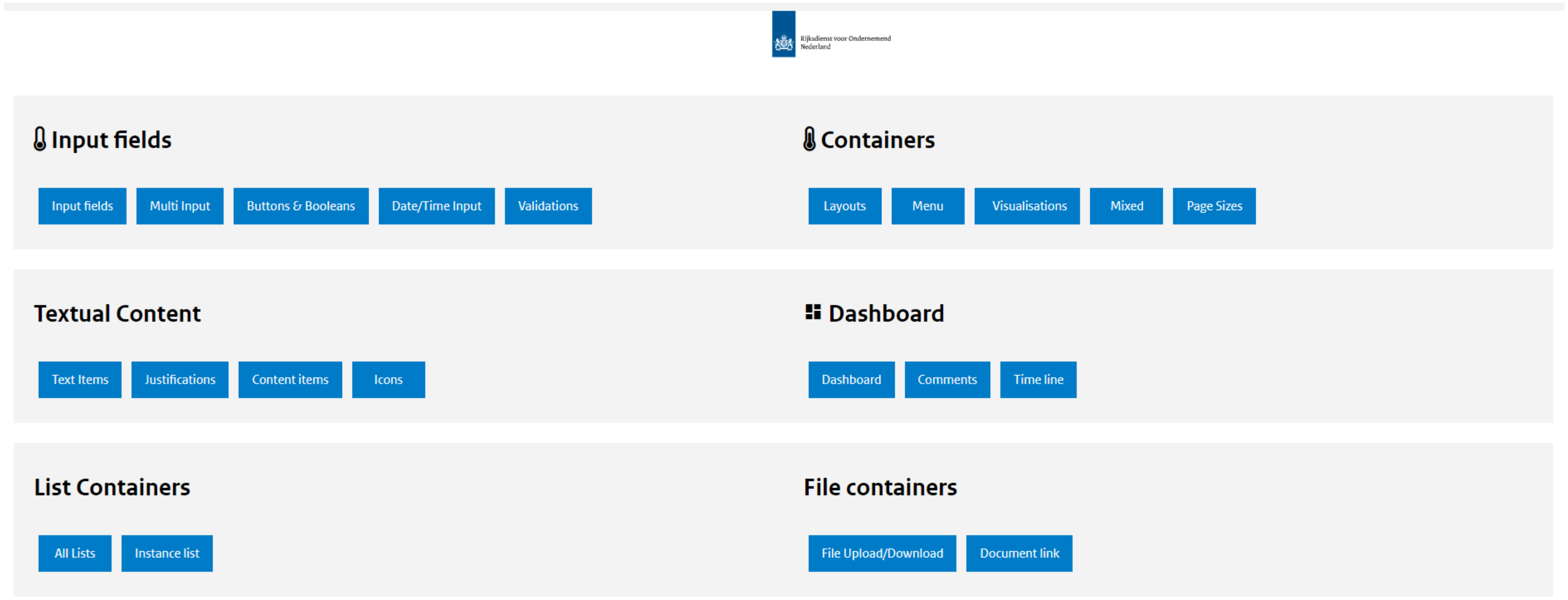
ALL LISTS · INSTANCE LIST

File containers

FILE UPLOAD/DOWNLOAD · DOCUMENT LINK

blueriq

Angular Rijkshuisstijl



Rijksdienst voor Ondernemend Nederland

Input fields

Input fields | Multi Input | Buttons & Booleans | Date/Time Input | Validations

Containers

Layouts | Menu | Visualisations | Mixed | Page Sizes

Textual Content

Text Items | Justifications | Content items | Icons

Dashboard

Dashboard | Comments | Time line

List Containers

All Lists | Instance list

File containers

File Upload/Download | Document link

Blueriq 11

Studio

- New features
- Enhancements
- Usability
- Unit testing

Runtime

New products

Front-end

Removed

Removed

- **XML representation of the RuleGraph**
  - RuleGraphView was removed
  - ISessionMonitor and IApplicationMonitor no longer provide methods for getting a RuleGraphView
- **Studio SQL Connectivity Wizard**
- **Process Engine**
  - Incoming and Outgoing Messages
    - Process event was removed
    - Event type was removed
    - Outgoing message nodes removed
- **AQ_MailServiceClient**
  - Generating an attachment from within the AQ_MailServiceClient is removed. Use a connection instead.
- **Expressions**
  - Version 6 expressions are no longer supported. Use the expression syntax that has been introduced in version 7.
  - The R4ANY and R4OCCURS grammer rules have been removed from the version 7 syntax.
- **Dossier SQL Store Plugin**
  - AQ_DossierManager
  - AQ_DossierList
- **XSLT Plugin**
- **AquimaLibrary**
  - AQ_ClearAttribute
  - AQ_SetAttribute
  - AQ_InstanceSelectorPlus
  - AQ_MenuBar
  - AQ_StartCaseProcess
  - AQ_SqlConnectivtyService
- **Request Parameter Service Plugin**
  - BB_SetRequestParameters
- **FileUpload Plugin**
  - BB_SingleFileUpload
- **WebFileUpload Plugin**
  - AQ_Web_FileUpload
- **Clusters**
  - Existing Clusters are automatically migrated to Labels