Creatio Academy

Collections

Process collections

Version 8.0

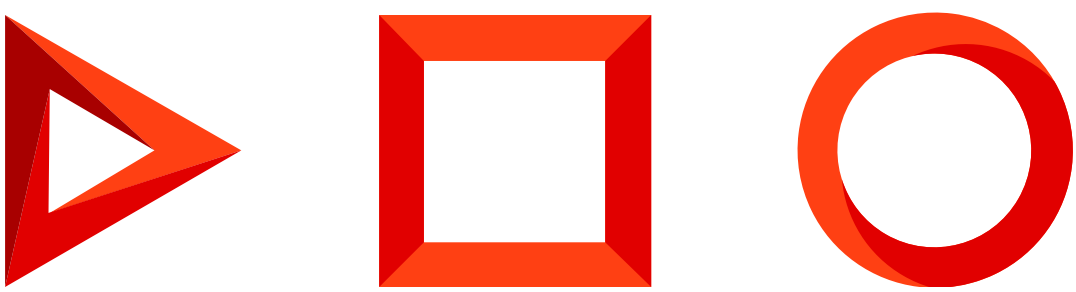


Table of Contents

Process collections

PRODUCTS: ALL CREATIO PRODUCTS

Several types of process elements, namely the [*Sub-process*], [*Read data*], and [*Call web-service*], can have outgoing parameters of the "Collection" type. To be able to work with the collection "instances" (separate records within a collection), the collection needs to be processed.

There are several ways of processing a collection:

- Pass a collection of records to another parameter of a "collection" type. For example, you can pass a collection of records to a [*Call web service*] element that has a collection in its request parameters.
- Use the [*Script task*] element to process the parameters of the "collection" type. For example, you can use a C# script to parse the collections of records into separate records that can be passed to other process elements.
- Display a collection of records using the [*Pre-configured page*] element. For example, display a list of invoices an employee has to review. To set up a collection in the pre-configured page, find the page view model in the [*Configuration*] section by title, add the "Serializable list of composite values" parameter with the needed sub-parameters to the model, and code the business logic.
- Use a collection in the [*User task*] element. For example, bulk update contacts. To set up a collection in the user task, click the button to open the task in the User Task Designer, add the "Serializable list of composite values" parameter with the needed sub-parameters, and code the business logic.
- Use the [*Sub-process*] element to handle each instance of a collection in a separate instance of the sub-process. If any of the incoming parameters of the [*Sub-process*] element are mapped to a data collection, the element will automatically run a separate instance of the sub-process per each instance of the collection.

Use sub-processes to handle "collection" parameters

Sub-process is the preferable method of working with collections, as it is not limited by the capabilities of a third-party web service, like the [*Call web service*] element, and does not involve coding, like the [*Script task*], [*Pre-configured page*], and [*User task*] elements.

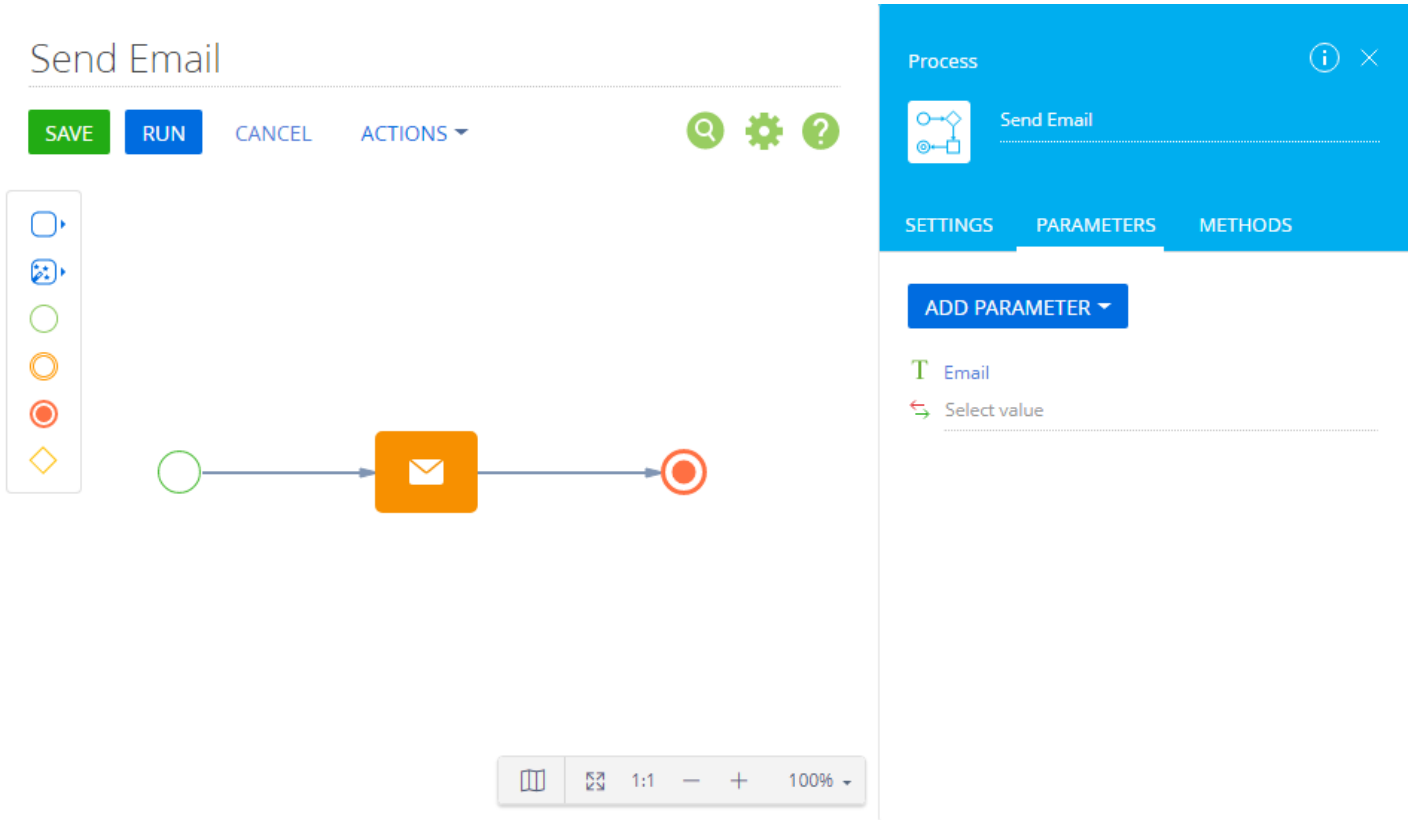
The general procedure for processing a collection via the [*Sub-process*] element is as follows:

1. Prepare a process that will handle collection instances (this process will be used as a sub-process).
2. Add the [*Sub-process*] element in the parent process and select the prepared sub-process.
3. Map the incoming and bi-directional parameters of the [*Sub-process*] element to the parameters of the collection type.

Prepare the process that will handle collection instances

Create or modify a process that will work with a collection instance as if it were a single record. For example, to send an email to a collection of contacts, prepare a process to send email to a single contact (Fig. 1).

Fig. 1 A basic email sub-process



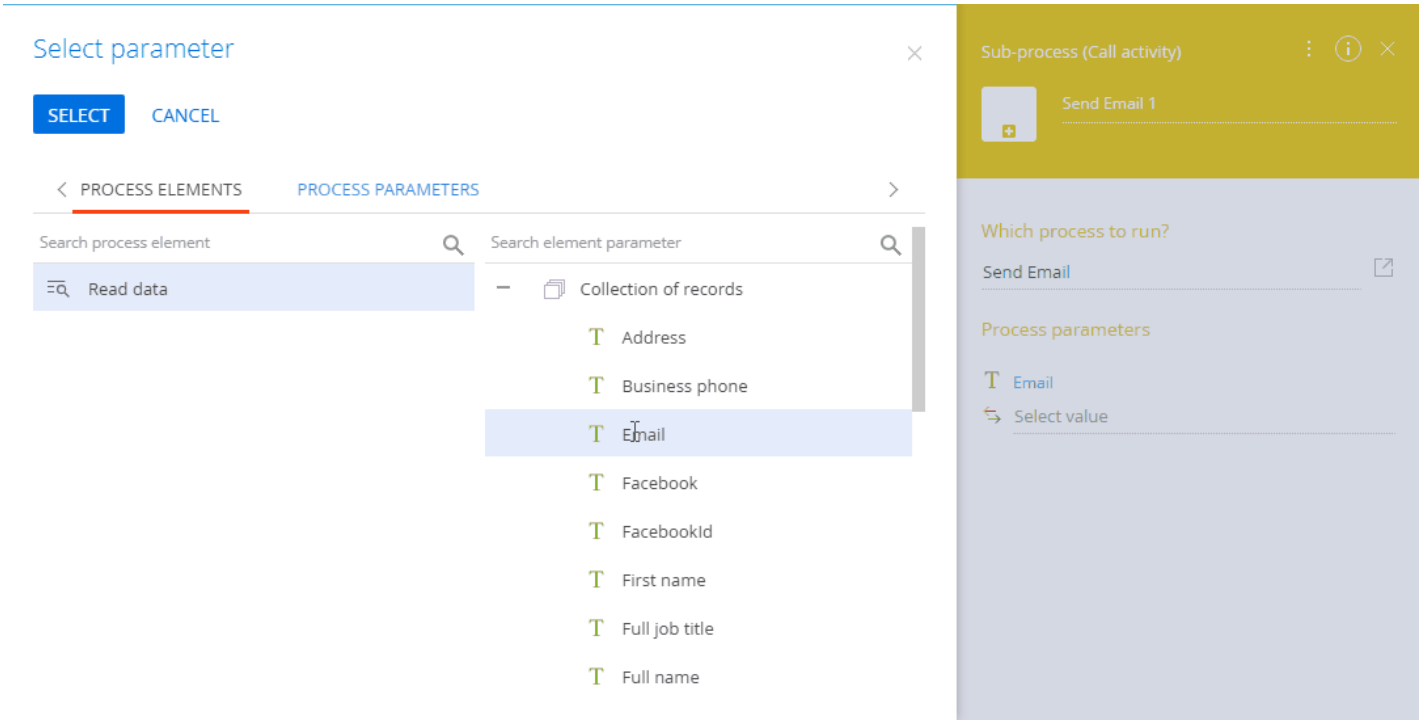
The incoming parameters of the sub-process must match the nested parameters of the collection. For example, sending an email requires the contact's email address. This means that the collection must include the [*Email*] column, while the process for sending email must have corresponding incoming [*Email*] parameter, whose value is then passed to the [*Send email*] element.

Note. You can access the sub-process diagram "on the go", while working with the parent process, by clicking + to create a new sub-process or to edit the current sub-process in the [*Which process to run*] field of the [*Sub-process*] element. Learn more: [*Sub-process*] element parameters.

Set up the [Sub-process] element

Add the [*Sub-process*] element to the diagram of the process where a collection parameter is obtained. Select a process that will handle collection instances in the [*Which process to run?*] field and map the incoming parameters of the selected process (Fig. 2):

Fig. 2 Mapping a data collection

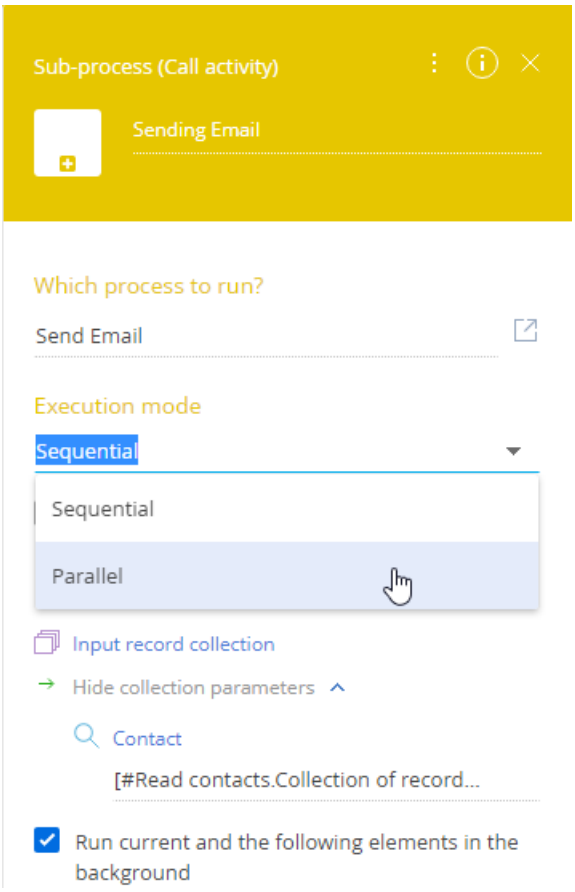


Attention. Parameters of a [*Sub-process*] element can only be mapped to one collection. There are no restrictions on using values from non-collection parameters.

As a result, the [*Sub-process*] element will become a multi-instance sub-process. Multi-instance sub-processes initiate a separate process instance for each item in the data collection, using data from that item as parameter values.

Multi-instance sub-processes have two execution modes (Fig. 3):

Fig. 3 Select the execution method



- **Sequential.** In this mode, sub-process instances run successively; each new sub-process instance runs upon completion of the previous instance. This is the default method.
- **Parallel.** In this mode, the [*Sub-process*] element initiates all its instances at once, without waiting for the completion of the already running instances before running a new instance. Sub-process instances are unlikely to complete in the same order they started.

Upon completion of the last instance, the [*Sub-process*] element will update its outgoing and bidirectional parameter values from the corresponding parameters of the completed sub-process instances and activate its outgoing flows.

This means that after processing a collection, the [*Sub-process*] element can return a new collection, based on the incoming one. For example, in the case of sending emails to a collection of contacts, the sub-process can be configured to return a collection of sending results and/or errors.

When using the parallel execution methods, outgoing parameters are added to the resulting data collection as soon as the corresponding sub-process instance completes. As a result, the order of items in the resulting data collection is unpredictable. For example, a sub-process instance mapped to the first element in the incoming data collection may be the last one to complete.