FLOWX.AI

BUILDING BLOCKS / Node / Message events / message-catch-boundary-event

Contents

BUILDING BLOCKS / Node / Message events / Message Catch Boundary Events

QUICK INTRO

What is it? A Message Catch Boundary Event is a special

The fallback content to display on prerendering

that can be attached to a user task in a

The fallback content to display on prerendering

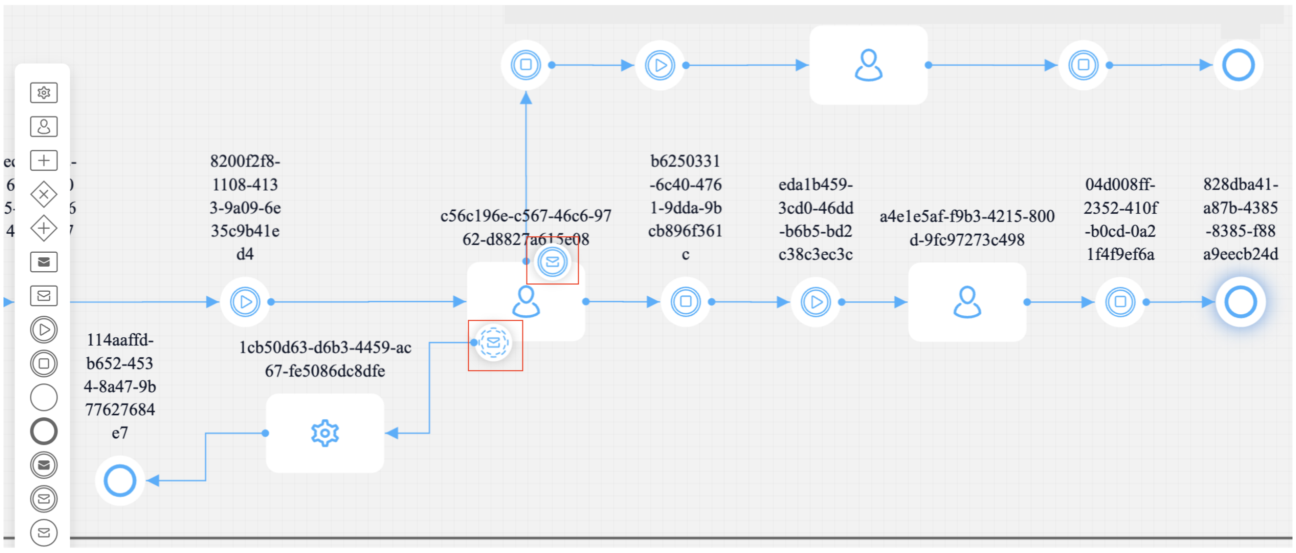
.

Why it is important? It allows the process to listen for and capture specific messages during the execution of the associated user task.

When used as a boundary event on a **user task**, message catch boundary event nodes behave similar to an **exclusive gateway**, but they are activated upon receiving an event. This means you can proceed in the process without receiving an event and continue through the sequence initiated from the user task.

If an event is received, it advances through the sequence from the intermediate

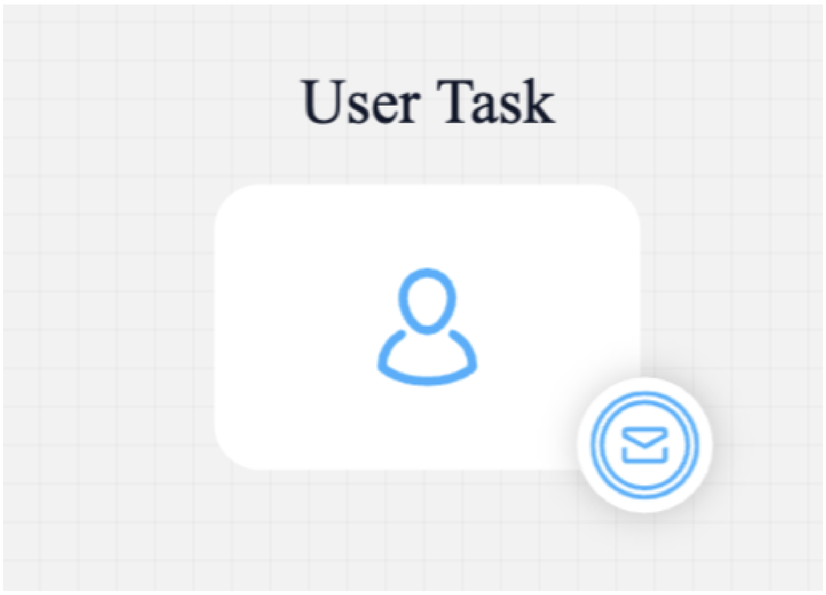
The fallback content to display on prerendering
. You can have multiple intermediate boundary events on the same user task, but only one can be activated at a time.


8200f2f8-1108-4133-9a09-6e35c9b41ed4
c56c196e-c567-46c6-9762-d8827a615e08
b6250331-6c40-4761-9dda-9bcb896f361c
eda1b459-3cd0-46dd-b6b5-bd2c38c3ec3c
a4e1e5af-f9b3-4215-800d-9fc97273c498
04d008ff-2352-410f-b0cd-0a21f4f9ef6a
828dba41-a87b-4385-8385-f88a9eecb24d
114aaffd-b652-4534-8a47-9b77627684e7
1cb50d63-d6b3-4459-ac67-fe5086dc8dfe

There are two types of Message Catch Boundary Events:

- Interrupting
- Non-Interrupting

Message Catch Interrupting Event



When an Interrupting Message Catch Boundary Event is triggered by receiving a message, it interrupts the associated task that is being performed. The task is immediately finished, and the

The fallback content to display on prerendering
continues to advance based on the received message.

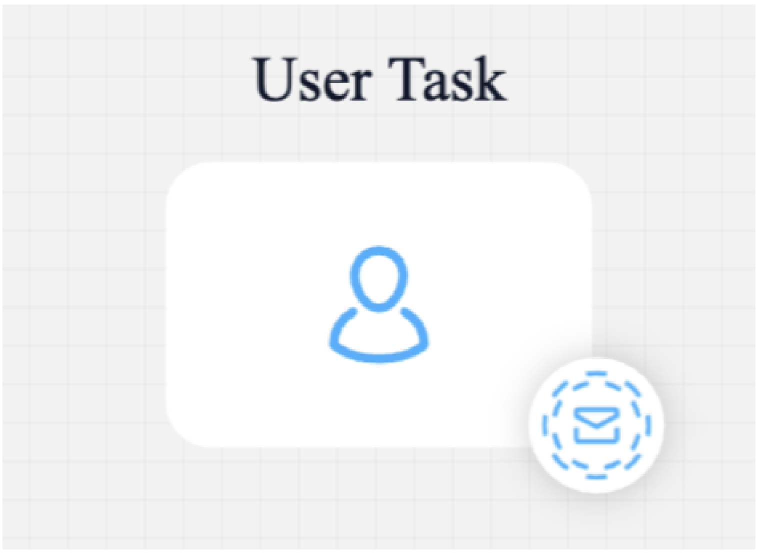
It can also be used as a standalone node, see more information on the following section:

» Message Catch Intermediate Event

Message Catch Non-Interrupting Event

It is used only as a boundary event and is placed only on a user task. If your process is in that user task and receives

The fallback content to display on prerendering
, the event is activated, and a new token is created that advances independently. Sections with non-interrupting events should not contain user tasks. You can have multiple non-interrupting events on the same user task, and all of them can be activated simultaneously.



A Non-Interrupting Message Catch Boundary Event also listens for messages while the associated task is being performed. However, in this case, the task is not immediately finished when messages are received. The event captures the messages, allowing the task to continue its execution. Multiple non-interrupting events can be received while the task is still active, and the task will continue until its completion.

Configuring a Message Catch Interrupting/Non-Interrupting Event

General config

- **Correlate with throwing events -** the dropdown contains all throw events from the process definitions accessible to the user

> **INFO**
>
> It is used to establish the correlation between the catch event and the corresponding throw event. By selecting the appropriate throw event, the catch event will be triggered when a message is thrown from that event.

- **Correlation key** - process key used to establish a correlation between the received message and a specific process instance

> **INFO**
>
> Correlation key serves as a means to correlate the incoming message with the specific process instance it belongs to. When a message is received with a matching correlation key, the catch event will be triggered.

- **Receive data (process key)** - the catch event can receive data associated with the message and store it in a process variable with the specified process key

> **INFO**

This data can then be used within the process instance for further processing or decision-making.

Was this page helpful?