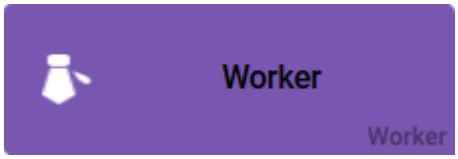
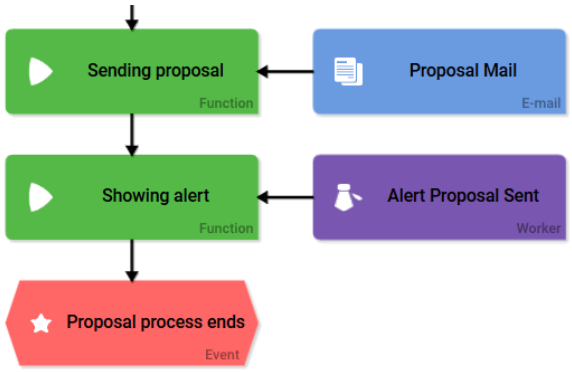
Worker

The Element Worker

Element	
Function	The EPK element Worker offers advanced users the possibility to program functionalities in JavaScript in order to extend the app with their own scripting. Worker can be easily exchanged, added, moved or removed. A Worker can be executed from client side or from server side. Workers are mainly used to execute server-side functionalities. A worker can be used for example for the automated generation of IDs. Other use cases are desired changes to the user interface such as the display and hiding of fields or manipulation of data in the container.
Connection	Predecessor: None Successor: **Function**
Example	During the ACME Bidding Proces users shall be informed of the successful dispatch of their Proposal Mail. The Worker Alert Proposal Sent contains therefore a JavaScript coding triggering a pop-up window with corresponding confirmation information to appear post dispatch. 

On this Page:

- The Element Worker
  - Configuration Options
    - Editor Worker Settings

Related Pages:

- Service
  - Bidding Process
  - Idea Management Process

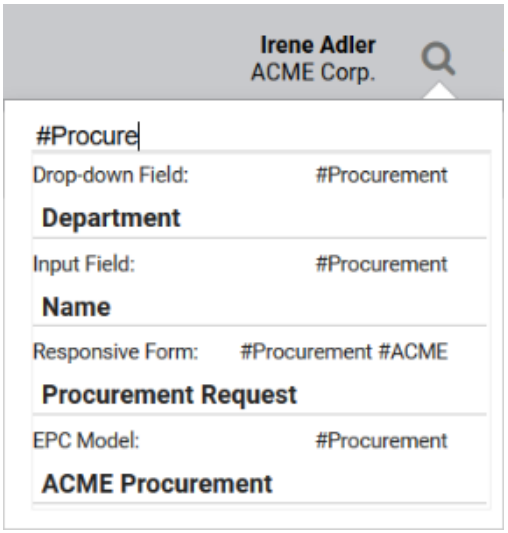
Related Documentation:

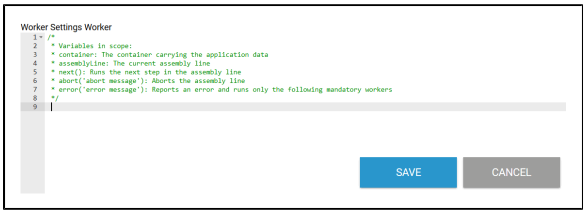
- BPaaS Entwicklerhandbuch (German)
  - Worker einsetzen

Configuration Options

Edit the element via the **Edit Sidebar**.

Name Worker	**Name** Name of the element within modeling.
ID rp02727f33-b7cf-40ef-886e-b4da441295ff	**ID ( )** The read-only field contains the model ID of the element. Designers ca example in the search to link the element in other models.

Description	**Description** A commentary field for the modeler. Its content is invisible to users and internal documentation.
Hashtags	**Hashtags** Possibility to insert your own key terms . The leading hashtag sign # **(h** inserted automatically. Use the spacebar to insert multiple hashtags: One hashtag may be issued for multiple elements: 
☐ Mandatory	**Mandatory** Marks the corresponding element as required. Activating this checkbox renders the execution of the worker mandator
WorkerGroupId	**WorkerGroupId** Please enter the name of the processing element executing the coding Generally this would be: • **frontend**: The worker is executed in browser/on your PC. • **backend**: The worker is executed in the backend/by the task engi If this field remains empty then the previous worker's GroupID is used

Worker Settings	**Worker Settings** A click on the gear wheel opens the editor where the worker's JavaScr entered. The gear wheel is already colored dark grey since the worker contains various (commented out) information regarding available envi  Double click on the element to open the **editor**. Do not start worker coding with a declaration.
Editor Worker Settings	
SAVE	Use the **Save** button in the pop-up window to save changes.
CANCEL	You can leave the pop-up window via the **Cancel** button. **Caution:** In that case, changes will not be saved!

Many functionalities as for example default settings for form fields can be customized via the form functions **Before Form Execution, After Show Form** or **After Form Execution** and do not require the use of a worker. Workers should mainly be used for more complex applications.