**Display**

|  |  |  |  |
| --- | --- | --- | --- |
| **Action** | **API** | **Activity** | **Registration Code** |
| Apply Conditions | fireACOnElem | No Server Interaction | NA |
| Close | * closeContainer | SubmitModalFlowAction | ClipboardPage actionAPI = ((PegaAPI)tools).getUIEngine().getUIActionPage("closeContainer");  ((PegaAPI)tools).getUIEngine().getUIAction(actionAPI).register(); |
| Expand Collapse | pega.u.d.toggleExpandCollapse |  |  |
| Home | pega.desktop.activateDocument(0); | ShowHarness | ClipboardPage actionAPI = ((PegaAPI)tools).getUIEngine().getUIActionPage("showHarness");  actionAPI.getPage("pyActionAPI").putString("pyActivityClass","@baseclass");  actionAPI.getPage("pyActionAPI").putString("pyHarnessName","[portal harness name]");  actionAPI.getPage("pyActionAPI").putString("pyTarget","newDocument");  actionAPI.getPage("pyActionAPI").putString("pyReadOnly","No");  ((PegaAPI)tools).getUIEngine().getUIAction(actionAPI).register();  actionAPI.removeFromClipboard(); |
| Menu |  | pzGetMenu | ClipboardPage actionAPI = ((PegaAPI)tools).getUIEngine().getUIActionPage("showMenu");  actionAPI.getPage("pyActionAPI").putString("pyDataSource",menuName);  ((PegaAPI)tools).getUIEngine().getUIAction(actionAPI).register(); |
| Mobile Search |  |  |  |
| Print |  | PrintWork | ClipboardPage actionAPI = ((PegaAPI)tools).getUIEngine().getUIActionPage("openUrlInWindow");        actionAPI.getPage("pyActionAPI").putString("pyActivity","PrintWork");        ((PegaAPI)tools).getUIEngine().getUIAction(actionAPI).register();  actionAPI.removeFromClipboard() |
| Refresh Section | reloadSection | ReloadSection | NA - Registered by default |
| Refresh Row |  |  |  |
| Show Section |  | pzShowSection | ClipboardPage actionAPI = ((PegaAPI) tools).getUIEngine().getUIActionPage("section");  actionAPI.getPage("pyActionAPI").putString("pySectionSource", "[property/name]");  actionAPI.getPage("pyActionAPI").putString("pySectionName", "[section name]");  actionAPI.getPage("pyActionAPI").putString("pyPageContext", "[none/datapage/clipboard/property]");  actionAPI.getPage("pyActionAPI").putString("pyUsingPage", "[page name]");  ((PegaAPI) tools).getUIEngine().getUIAction(actionAPI).register();  actionAPI.removeFromClipboard(); |
| Set Focus |  | No Server Interaction |  |
| Set Style |  | No Server Interaction |  |
| Set Value |  | No Server Interaction |  |
| Post Value |  | ReloadCell | NA - Registered by default |
| Show SmartInfo | No public api | ReloadSection |  |
| Spell Checker |  | Data-SpellChecker.SpellCheck | Not needed, depricated since 7.x |
|  |  |  |  |

**Process Work**

|  |  |  |  |
| --- | --- | --- | --- |
| **Action** | **API** | **Activity** | **Registration Code** |
| Add New Work | doFormSubmit("pyActivity=Add") | Add | ClipboardPage actionAPI = ((PegaAPI)tools).getUIEngine().getUIActionPage("addWork");  ((PegaAPI)tools).getUIEngine().getUIAction(actionAPI).register(); |
| Cancel | doClose | doClose | ClipboardPage actionAPI = ((PegaAPI)tools).getUIEngine().getUIActionPage("cancel");  ((PegaAPI)tools).getUIEngine().getUIAction(actionAPI).register();  actionAPI.removeFromClipboard(); |
| Contents | doFormSubmit("pyActivity=OpenCover&HarnessPurpose=Review&AllowInput=false") | OpenCover |  |
| Enable Action Section | Deprecated |  |  |
| Explore | openUrlInWindow | pzProcessURLInWindow | ClipboardPage actionAPI = ((PegaAPI)tools).getUIEngine().getUIActionPage("openUrlInWindow");        actionAPI.getPage("pyActionAPI").putString("pyActivity","[ activity name]");  actionAPI.getPage("pyActionAPI").putString("pyUseAlternateUrlBase", "false");  actionAPI.getPage("pyActionAPI").putString("pyPreDataTransform", "[data transform name]");        ((PegaAPI)tools).getUIEngine().getUIAction(actionAPI).register();  actionAPI.removeFromClipboard(); |
| Finish Assignment | doFormSubmit("pyActivity=FinishAssignment") | FinishAssignment | ClipboardPage actionAPI = ((PegaAPI) tools).getUIEngine().getUIActionPage("finishAssignment");  ((PegaAPI) tools).getUIEngine().getUIAction(actionAPI).register();  actionAPI.removeFromClipboard(); |
| Perform Action | takeAction |  |  |
| Review | doFormSubmit("pyActivity=Open&Action=Review&AllowInput=false") | Open |  |
| Save | doSave | SaveAndContinue  doSave  ReloadHarness | ClipboardPage UIActionPage = ((PegaAPI) tools).getUIEngine().getUIActionPage("save");  ClipboardPage actionAPI = UIActionPage.getPage("pyActionAPI");  actionAPI.putString("pyCloseAfterSave", "true/false");  ((PegaAPI) tools).getUIEngine().getUIAction(UIActionPage).register();  UIActionPage.removeFromClipboard(); |
| Show Flow Location | openUrlInWindow | pzProcessURLInWindow |  |
| Show Re-Open Screen | doFormSubmit(""pyActivity=Open&Action=Reopen&AllowInput=true&HarnessPurpose=Reopen") | Open |  |
| Update | handleUpdate | DoUpdate |  |
| View Attachments | openUrlInWindow | pzProcessURLInWindow |  |
| View History | openUrlInWindow | pzProcessURLInWindow |  |

**Launch**

|  |  |  |  |
| --- | --- | --- | --- |
| **Action** | **API** | **Activity** | **Registration Code** |
| Flow In Modal Dialog | pega.u.d.launchFlowInModal |  |  |
| Get Directions | getDirections |  |  |
| Show Harness | showHarness  showHarnessWrapper  pega.api.ui.actions.launchHarness | doUIAction  @baseclass.pzTransformAndRun | ClipboardPage UIActionPage = ((PegaAPI) tools).getUIEngine().getUIActionPage("showHarness");  ClipboardPage actionAPI = UIActionPage.getPage("pyActionAPI");  actionAPI.putString("pyActivityClass", "[harness class]");  actionAPI.putString("pyHarnessName", "[harness name]");  actionAPI.putString("pyReadOnly", "Yes/No");  actionAPI.putString("pyKey", "key value");  actionAPI.putString("pyUsingPage", "[uasing page name]");  actionAPI.putString("pyTarget", "[newDocument/current/popup]");  ClipboardPage dtpage = actionAPI.getPage("pyPreDataTransform");  dtpage.putString("pyName", "[pyPreDataTransform name]");  ClipboardPage dtParam = dtpage.getProperty("pyDataTransformParams").getPageValue(ClipboardProperty.LIST\_APPEND);  dtParam.putString("pyName","[name of DT param]");  dtParam.putString("pyValue","[value of DT param]");  actionAPI.putString("pyActivity", "[pre activity name]");  ClipboardPage preActParam = actionAPI.getProperty("pyActivityParams").getPageValue(ClipboardProperty.LIST\_APPEND);  preActParam.putString("pyName","[name of pre activity param]");  preActParam.putString("pyValue","[value of pre activity param]");  ((PegaAPI) tools).getUIEngine().getUIAction(UIActionPage).register();  UIActionPage.removeFromClipboard(); |
| Mobile View |  |  |  |
| Landing Page | openLandingPage | doUIActionpega.u.d.processAction("pxModalDialog", '', "Rule-HTML-Section", '', '', true, event, 'pyModalTemplate', oSafeURL, element, '', '', callbacks, options, false); | ClipboardPage UIActionPage = ((PegaAPI) tools).getUIEngine().getUIActionPage("openlanding");  ClipboardPage actionAPI = UIActionPage.getPage("pyActionAPI");  actionAPI.putString("pyAction", "[Display/CreateNewWork]");  actionAPI.putString("pyClassName", "[class value]");  actionAPI.putString("pyName", "[name value]");  actionAPI.putString("pyLevelA", "[pyLevelA value]");  actionAPI.putString("pyLevelB", "[pyLevelb value]");  actionAPI.putString("pyLevelC", "[pyLevelc value]");  ClipboardPage pyDisplay = actionAPI.getPage("pyDisplay");  pyDisplay.putString("pyPage", "[primary page name]");  pyDisplay.putString("pyReadOnly", "[true/false]");  pyDisplay.putString("pyHarnessName", "[harness name]");  ClipboardPage pyCreateNewWork = actionAPI.getPage("pyCreateNewWork");  pyCreateNewWork.putString("pyFlowName", "[flow name  name]");  ClipboardPage params = actionAPI.getProperty("pyParameters").getPageValue(ClipboardProperty.LIST\_APPEND);  params.putString("pyName", "[param name]");  params.putString("pyValue", "[parameter value]");  ((PegaAPI) tools).getUIEngine().getUIAction(UIActionPage).register();  UIActionPage.removeFromClipboard(); |
| Local Action | processAction | ProcessAction | // String actionSection => Template section  // String flowActionName => Flow action rule name  HashMap<String, String> config = new HashMap<String, String>();  config.put("ActionSection", actionSection);  config.put("NewTaskStatus", flowActionName);  ((PegaAPI)tools).getUIEngine().getUIAction("localAction", config).register(); |
| Open URL In Window | openUrlInWindow | pzProcessURLInWindow | ClipboardPage UIActionPage = ((PegaAPI)tools).getUIEngine().getUIActionPage("openURLInWindow");  ClipboardPage actionAPI =  UIActionPage.getPage("pyActionAPI");  actionAPI.putString("pyActivity",activityName);  ClipbardProperty queryString = actionAPI.getProperty("pyQueryString");  ClipboardPage queryStringItem = tools.createPage("Embed-NameValuePair","");  queryStringItem.putString("pyName",Key);  queryStringItem.putString("pyValue",Value);  queryString.add(queryStringItem);  ((PegaAPI)tools).getUIEngine().getUIAction(UIActionPage).register(); |
| Open Mobile App |  |  |  |
| Report Definition | reportDefinition | doUIAction |  |
| Scan Barcode / QR Code | scanCode |  |  |
| Open Wizard | openWizard | StartWizard | ClipboardPage UIActionPage = ((PegaAPI) tools).getUIEngine().getUIActionPage("openWizard");  ClipboardPage actionAPI = UIActionPage.getPage("pyActionAPI");  actionAPI.putString("pyClass", "[class value]");  actionAPI.putString("pywizardLabel", "[pywizardLabel value]");  ClipboardPage params = actionAPI.getProperty("pyParameters").getPageValue(ClipboardProperty.LIST\_APPEND);  params.putString("pyName", "[param name]");  params.putString("pyValue", "[parameter value]");  ((PegaAPI) tools).getUIEngine().getUIAction(UIActionPage).register();  UIActionPage.removeFromClipboard(); |

**Get Work**

|  |  |  |  |
| --- | --- | --- | --- |
| **Action** | **API** | **Activity** | **Registration Code** |
| Create Work | createNewWork | doUIAction  NewFromFlow | ClipboardPage UIActionPage = ((PegaAPI) tools).getUIEngine().getUIActionPage("createWork");  ClipboardPage actionAPI = UIActionPage.getPage("pyActionAPI");  actionAPI.putString("pyClassName", "[class name]");  actionAPI.putString("pyFlowName", "[Flow name]");  ClipboardPage flowParams = actionAPI.getProperty("pzRuleParameters").getPageValue(ClipboardProperty.LIST\_APPEND);  flowParams.putString("pxObjClass", "Embed-NameValuePair", );  flowParams.putString("pyName", "[flow parameter name]");  flowParams.putString("pyValue", "[flow parameter value]");  ((PegaAPI) tools).getUIEngine().getUIAction(UIActionPage).register();  UIActionPage.removeFromClipboard(); |
| Get Next Work | getNextWorkItem | doUIAction  Work-.GetNextWork | ClipboardPage UIActionPage = ((PegaAPI) tools).getUIEngine().getUIActionPage("nextWork");  ((PegaAPI) tools).getUIEngine().getUIAction(UIActionPage).register();  UIActionPage.removeFromClipboard(); |
| Open Assignment | openAssignment | doUIAction  Assign-Worklist.ProcessAssignment | ClipboardPage UIActionPage = ((PegaAPI) tools).getUIEngine().getUIActionPage("openAssignment");  ClipboardPage actionAPI = UIActionPage.getPage("pyActionAPI");  actionAPI.putString("pyInsKey", "[assignment key]");  ((PegaAPI) tools).getUIEngine().getUIAction(UIActionPage).register();  UIActionPage.removeFromClipboard(); |
| Open Work By Handle | openWorkByHandle | doUIAction  Open | ClipboardPage UIActionPage = ((PegaAPI) tools).getUIEngine().getUIActionPage("openWorkByHandle");  ClipboardPage actionAPI = UIActionPage.getPage("pyActionAPI");  actionAPI.putString("pyKey", "[work handle]");  ((PegaAPI) tools).getUIEngine().getUIAction(UIActionPage).register();  UIActionPage.removeFromClipboard(); |
| Open Work Item | openWorkItem | doUIAction  GetWorkByID | ClipboardPage UIActionPage = ((PegaAPI) tools).getUIEngine().getUIActionPage("openWorkByHandle");  ClipboardPage actionAPI = UIActionPage.getPage("pyActionAPI");  actionAPI.putString("pyItemId", "[work item id]");  ((PegaAPI) tools).getUIEngine().getUIAction(UIActionPage).register();  UIActionPage.removeFromClipboard(); |
| Reopen Work Item | doFormSubmit("pyActivity=Reopen") | doUIAction |  |

**List Actions**

|  |  |  |  |
| --- | --- | --- | --- |
| **Action** | **API** | **Activity** | **Registration Code** |
| Add Child | No public API | ReloadSection | Same as "Reload section" action |
| Add Item | **Non template table**  No public API  **Repeating dynamic layout**  addToRepeatSource(event, datasource, position, localAction, templateName, rowClass, packagedPageName, bSkipLocalAction)  **Template table**  No public API | **Non template table**  ReloadSection  **Repeating dynamic layout**  @baseclass.pzAddToRepeatSource  **Template table**  @baseclass.pzDoListPreAddItem | **Non template table**  Same as "Reload section" action  **Repeating dynamic layout**  ClipboardPage uiActionPage = ((PegaAPI) tools).getUIEngine().getUIActionPage("addRow");  ClipboardPage actionAPI = uiActionPage.getPage("pyActionAPI");  actionAPI.putString("pyRepeatCustomTemplateName", "[Template name]");  actionAPI.putString("pyRowPromptClass", "[Row class]");  actionAPI.putString("pyPosition", "[First / Last]");  actionAPI.putString("pyEditFlowAction", "[flow action name]");  actionAPI.putString("pyRepeatLayoutType", "RDL");  ((PegaAPI) tools).getUIEngine().getUIAction(uiActionPage).register();  uiActionPage.removeFromClipboard();  **Template table**  ClipboardPage uiActionPage = ((PegaAPI) tools).getUIEngine().getUIActionPage("addRow");  ClipboardPage actionAPI = uiActionPage.getPage("pyActionAPI");  actionAPI.putString("pyRepeatCustomTemplateName", "[Template name]");  actionAPI.putString("pyRowPromptClass", "[Row class]");  actionAPI.putString("pyRepeatDPName", "[Datapage name]");  actionAPI.putString("pyPosition", "[First / Last]");  actionAPI.putString("pyEditFlowAction", "[flow action name]");  actionAPI.putString("pyRepeatLayoutType", "Grid");  ((PegaAPI) tools).getUIEngine().getUIAction(uiActionPage).register();  uiActionPage.removeFromClipboard(); |
| Delete Item | **Non template table**  No public API  **Repeating dynamic layout**  removeFromRepeatSource(event, showConfirmBox)  **Template table**  No public API | **Non template table**  ReloadSection  **Repeating dynamic layout**  @baseclass.pzRemoveFromRepeatSource  ReloadSection  **Template table**  @baseclass.pzDoListPreAddItem  ProcessAction  SubmitModalFlowAction | **Non template table**  Same as "Reload section" action  **Repeating dynamic layout**  ClipboardPage uiActionPage = ((PegaAPI) tools).getUIEngine().getUIActionPage("deleteRow");  ((PegaAPI) tools).getUIEngine().getUIAction(uiActionPage).register();  uiActionPage.removeFromClipboard();  **Template table**  ClipboardPage uiActionPage = ((PegaAPI) tools).getUIEngine().getUIActionPage("deleteRow");  ClipboardPage actionAPI = uiActionPage.getPage("pyActionAPI");  actionAPI.putString("pyNextGenGridDeleteConfirm", "[true / false]");  ((PegaAPI) tools).getUIEngine().getUIAction(uiActionPage).register();  uiActionPage.removeFromClipboard(); |
| Edit Item | **Non template table**  No public API  **Template table**  No public API | **Non template table**  pzPerformGridAction  **Template table**  @baseclass.pzDoListAction  SubmitModalFlowAction  ProcessAction | **Non template table**  Registered by default  **Template table**  Registered by default |
| Open Local Action | **Non template table**  No public API | **Non template table**  pzPerformGridAction | **Non template table**  Registered by default |
| Open Selected Item | **Non template table**  No public API | **Non template table**  Dynamically executed | **Non template table**  Register one of the action  openWorkByHandle, openWorkItem, openAssignment, openRuleByKeys, openRuleByClassAndName, openRule |
| Refresh Current Row | **Non template table**  No public API | **Non template table**  ReloadSection | **Non template table**  Same as "Reload section" action |
| Refresh List | **Non template table**  No public API | **Non template table**  ReloadSection | **Non template table**  Same as "Reload section" action |
| Set Focus |  | No Server Interaction |  |

**Other / Misc**

|  |  |  |  |
| --- | --- | --- | --- |
| **Action** | **API** | **Activity** | **Registration Code** |
| Invoke Action |  |  |  |
| Log Off | pega.u.d.LogOff | LogOff | Universal Registration |
| Open Rule | pega.desktop.openRule | doUIAction  WBOpen  @baseclass.pzGetFormType | ClipboardPage UIActionPage = ((PegaAPI) tools).getUIEngine().getUIActionPage("openRule");  ClipboardPage actionAPI = UIActionPage.getPage("pyActionAPI");  actionAPI.putString("pyHandle", "[Rule pzins key]");  actionAPI.putString("pyPromptResolveType", "[open\_specific/rule\_resolve]");  actionAPI.putString("pyOpenSpecificVersion", "[true/false]");  actionAPI.putString("pyWhen", "[when rule name]");  ((PegaAPI) tools).getUIEngine().getUIAction(UIActionPage).register();  UIActionPage.removeFromClipboard(); |
| Open Rule By Keys | pega.desktop.openRuleByKeys | doUIAction  WBOpen  pzGetRuleInfo  @baseclass.pzGetFormType  @baseclass.pzGetRuleInfo | ClipboardPage UIActionPage = ((PegaAPI) tools).getUIEngine().getUIActionPage("openRuleByKeys");  ClipboardPage actionAPI = UIActionPage.getPage("pyActionAPI");  actionAPI.putString("pyObjClass", "[Rule pxObjClass class]");  ClipboardPage keyset = actionAPI.getProperty("pyKeySet").getPageValue(ClipboardProperty.LIST\_APPEND);  keyset.putString("pyName", "key name");  keyset.putString("pyValue", "key value");  ((PegaAPI) tools).getUIEngine().getUIAction(UIActionPage).register();  UIActionPage.removeFromClipboard(); |
| Open Rule By Name | pega.desktop.openRuleByClassAndName | doUIAction  WBOpen  @baseclass.pzGetFormType  @baseclass.pzGetRuleInfo | ClipboardPage UIActionPage = ((PegaAPI) tools).getUIEngine().getUIActionPage("openRuleByKeys");  ClipboardPage actionAPI = UIActionPage.getPage("pyActionAPI");  actionAPI.putString("pyInsName", "[Rule ins key]");  actionAPI.putString("pyObjClass", "[Rule ins key]");  ((PegaAPI) tools).getUIEngine().getUIAction(UIActionPage).register();  UIActionPage.removeFromClipboard(); |
| Run Activity | pega.u.d.asyncRequest  pega.api.ui.actions.runActivity  Any server call whose SafeURL has been converted by pega.u.d.convertToRunActivityAction | pzRunActionWrapper | pega\_rules\_utilities.pzRegisterActivity(tools,activityName); |
| Run Data Transform | pega.api.ui.actions.runDataTransform | pzRunActionWrapper | ClipboardPage actionAPI = ((PegaAPI)tools).getUIEngine().getUIActionPage("runDataTransform");  actionAPI.getPage("pyActionAPI").putString("pyPreDataTransform.pyName",dataTransformName);        ((PegaAPI)tools).getUIEngine().getUIAction(actionAPI).register();        actionAPI.removeFromClipboard(); |
| Run Script | No server call |  |  |
| section | pega.control.Actions.prototype.opensection  (or)  pega.c.UIElement.Actions.opensection | pzShowSection | ClipboardPage actionPage = pega.getUIEngine().getUIActionPage("section");  actionPage.getPage("pyActionAPI").putString("pySectionSource", "name");  actionPage.getPage("pyActionAPI").putString("pyACName","acprimary");  actionPage.getPage("pyActionAPI").putString("pySectionName","MileStoneWrapper");  actionPage.getPage("pyActionAPI").putString("pyPageContext","datapage");  actionPage.getPage("pyActionAPI").putString("pyUsingPage","MilestoneDetailsDisplay");  pega.getUIEngine().getUIAction(actionPage).register();  actionPage.removeFromClipboard(); |

**Call Site Registration for BAC Mitigation**

Application users who have built a part of their UI using custom code, in almost all cases would be relying on the well known API -**pega.u.d.asyncRequest**  , to make their AJAX calls.

If not using  , such users should migrate to use this API\* as it provides a lost of additional functionality like queuing , change tracking etc and now in 8.5 also provides an easy way to fix BAC issues.

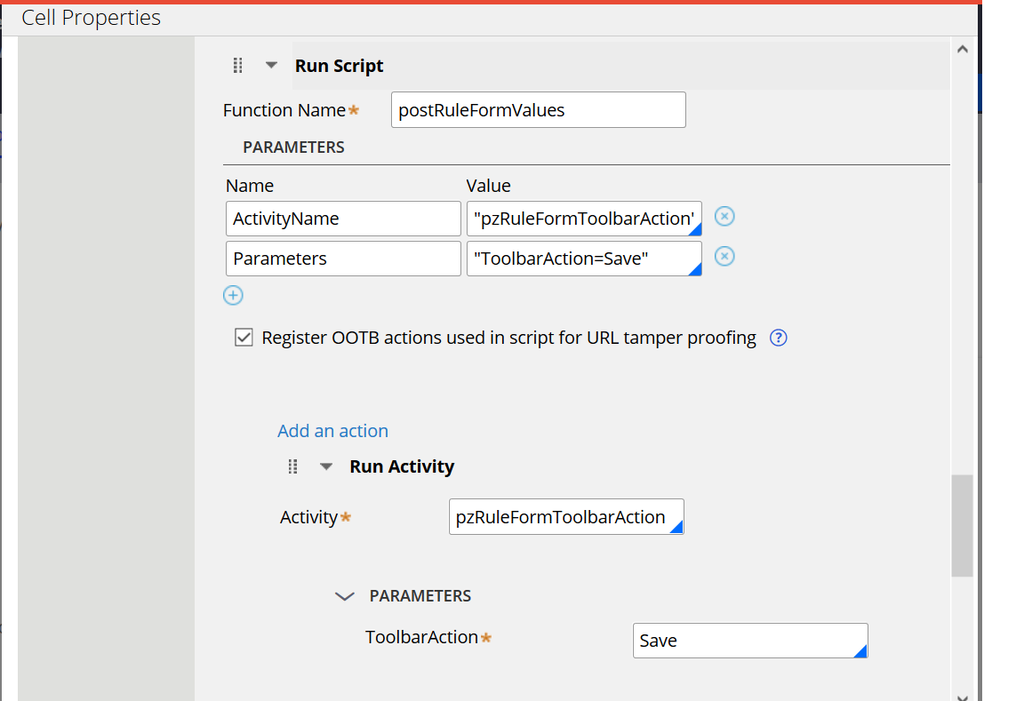
In 8.5 any activity call that uses this API would be dispatched using a dispatcher activity (as opposed to being called directly). ).

Suppose user has a button or link that has an event handler which internally calls an application activity using pega.u.d.asyncRequest .

Use call site registration , to register such activity.

Go to the runscript action in the property panel which calls the event handler, and under the action there is checkbox option : Register OOTB actions used in script for URL tamper proofing

* Under that select runActivity as the action and enter your activity name there. Thats all there is to it.



* In case you invoke activities during page load etc, then add a proxy hidden(Visibility = Never) button in your UI to register the activity there.
* Alternatively , if you have a control that has non auto generated html, you can use the simple RUF to register in code  - pega\_rules\_utilities.pzRegisterActivity(tools,**activityName**)

**Important**: Register your activity as local and close to the call site as possible. Donot keep it at a global place as that would mean the activity can be invoked from any context.

\*Check this gist for ways to modify your safe url object to make it compatible with Registration.

<https://gist.github.com/bokks/0d25a37d51c7ddeadb664c5f94b304d1>