Uninstall Applications

This action occurs after an application is uninstalled from a sandbox.

Action Type	Embedded
Full Action ID	embedded.platform.applications.uninstall
Runs multiple custom functions?	Yes

JavaScript File Structure

Action files share the following basic structure:

```
module.exports = function(context, callback) {
  // Your custom code here
  callback();
};
```

When you code the custom function for an action, you have access to two arguments:

`callback` —This argument follows the established JavaScript callback pattern: it takes an error as the first argument (or null if there is no error) and a result as the second argument (if required).

`context` —This argument provides the function access to relevant objects and methods that interface with Kibo.

Context: Platform

The following methods and objects are available to this action through the use of the `context` argument.

Microservice Operation
This action corresponds to the microservice that installs and uninstalls applications.

Get Methods

- get.applicationKey
- get.exports
- get.installationState
- get.nameSpace

Exec Methods

- exec.saveInstallationState

Context Objects Available to All Actions

- apiContext
- configuration

Get

get.applicationKey

Returns the application key.

Parameters	Type	Description
N/A	N/A	N/A

Example:

```
context.get.applicationKey();
```

Response:

```
"string"
```

get.exports

Returns the exports for the application.

Parameters	Type	Description
N/A	N/A	N/A

Example:

```
context.get.exports();
```

Response:

```
"object"
```

get.installationState

Returns the installation state of the application.

Parameters	Type	Description
N/A	N/A	N/A

Example:

```
context.get.installationState();
```

Response:

```
"object"
```

get.nameSpace

Returns the namespace of the application.

Parameters	Type	Description
N/A	N/A	N/A

Example:

```
context.get.nameSpace();
```

Response:

```
"string"
```

Exec

exec.saveInstallationState

Saves the supplied application installation state to Kibo.

Parameters	Type	Description
installationState	object	The application installation state. You can retrieve this object using the corresponding GET method.

Example:

```
context.exec.saveInstallationState(installationState);
```

Response: N/A

Context Objects Available to All Actions

apiContext

Accesses tenant information.

Properties	Type	Description
baseUrl	string	The base URL for the site.
basePciUrl	string	The base PCI URL for the site.
tenantPod	string	The name of the tenant pod in which the tenant resides.
appClaims	string	The application claims token.
appKey	string	The application key.
tenantId	integer	Unique identifier for the tenant.
siteId	integer	Unique identifier for the site. This ID is used at all levels of a store, catalog, and tenant to associate objects to a site.
masterCatalogId	integer	Unique identifier for the master catalog.
catalogId	integer	The unique identifier for the product catalog. Catalogs are part of a master catalog.
currencyCode	string	The default three-letter ISO currency code for monetary amounts.
previewDate	date/time	The date and time that the content is being viewed. This might be a future date if the content is previewed with an active date range set in the future.
localeCode	string	The locale code per the country code provided. This code determines the localized content to use and display.
correlationId	string	The unique identifier of the API request associated with the event action, which might contain multiple actions.
isAuthorizedAsAdmin	Boolean	Indicates whether the Dev Account user is authorized as an admin.
userClaims	string	The user claims token.

Example:

```
context.apiContext.baseUrl;
```

configuration

Receives a JSON response that contains information about the configuration data set in the Action Management JSON editor.

Properties	Type	Description
Varies	object	Custom fields and values that you can set in the Action Management JSON Editor.

Example:

```
context.configuration.customData;
```