# Widget Context Technical Guide

## Sitecore Discover

V 1.0.11 20210218

# Table of Contents

# Chapter 1 Introduction

Sitecore Discover works best when it has precise context information regarding which Discover widgets are shown to the website user. Examples of context information includes skus that are included on the page, locale of the page, the store id that the user has selected, and so on...

This document describes the interface that lets you provide widget context information.

Following are some examples of use cases where you might want to use a widget context:

- Specify the locale selected by the user.
- Specify the store selected by the user.
- Specify SKU(s) on displayed on the page (for example, the product details, cart or order confirm pages).
- Specify the category selected by user when user hovers on a menu item for specific widget in the menu to show products for that category.
- Specify specific sku(s) on a PDP page in order to refresh the recommended product shown for widgets on that page.
- Specify specific sku(s) on Cart page as user modifies the cart, in order to refresh the recommended product shown for widgets on that page.
- Specify specific skus on quickview page as user makes selection for specific skus, in order to refresh the recommended product shown for widgets on that page.

# Chapter 2 Specifying updateContext

You can specify the context for a page or specific widget using the following push script format:

```
        var rfk = window.rfk = window.rfk || [];
        rfk.push(["updateContext", <UpdateContext Object 1>, <UpdateContext Object 2>, …]);
```

## 2.1 UpdateContext Object

The updateContext object defines the widget(s) that the context applies to, the specific user, store, campaign or page contexts to set, and the filter context to set.

Format:

```
        {
            "widget": {
                "rfkids": ["<rfkid 1>", "<rfkid 2>", ...],
            },
            "context": {
                "user": {...},
                "store": {...},
                "campaign": {...},
                "page": {
                    "<key 1>": "<value 1>",
                    "<key 2>": "<value 2>",
                    ...
                }
            },
            "filter": {
                "<type 1>": ["<value 1>", "<value 2>", ...],
                ...
            },

            "apply_once": true|false
        }
```

The following table list the keys:

| Key | Value Type | Description |
|---|---|---|
| widget | object | Specifies widgets that the context applies to. If the widget object doesn't exist, then the context applies to all widgets on the page. |
| context | object | Represents the context to apply. A context encapsulates various types of contexts like user, page and store. |
| filter | object | Represents the filters to apply. |
| apply_once | bool. | Usually, once you set the content, the context persists for the full duration of the page interaction. However, there are times when you want the context you set to be non-persistent. For example, if you want to set the context only one time, and once the widget refreshes you want the content to be removed automatically, then set this flag to true.<br>Default false. |

### 2.1.1 The Widget object

The widget context object supports the following keys:

| Key | Value Type | Description |
|---|---|---|
| `rfkids` | List of strings | List of rfkids. Each rfkid refers to the widget on the page that this context applies to. |

## 2.1.2 The Context object

The `context` object supports the following keys:

| Key | Value Type | Description |
|---|---|---|
| `user` | object | Object defining the user context. |
| `store` | object | Object defining the Store context |
| `campaign` | object | Object describing the campaign context |
| `page` | context | Object describing the context of the page |
| `fitment` | object | Object describing the fitment context |

### The User

The `user` context object supports the following keys:

| Key | Value Type | Description |
|---|---|---|
| `groups` | List of strings | Custom groups or segments assigned to user. B2B websites should not use this to identify restrictions groups. See Context - Store for restriction groups. |

### The Store object

The `store` context object supports the following keys:

| Key | Value Type | Description |
|---|---|---|
| `id` | string | Store id. This id should be consistent with the id shared as part of the full product feed.<br><br>B2B websites use this to identify a restriction id associated with a customer. If a restriction is based on the customer id, specify the customer id. If you have a restriction id that is different than customer id, use the specific restriction id. |

| | | |
|---|---|---|
| `group_id` | string | group_id. If your stores are organized in two level hierarchy (for example as zones and stores), you can pass the top level organization identifier as group_id. This id should be consistent with group_id shared as part of product feed.<br><br>For B2B websites, if your restrictions are organized in two-levels, then pass the restriction group id as group_id. If restriction is based on the customer group id, specify the customer group id. If you have a restriction group id that is different then customer group id, use the specific restriction group id. |

## The Campaign object

The `campaign` context object supports the following keys:

| Key | Value Type | Description |
|---|---|---|
| `utm_source` | string | utm_source campaign parameter |
| `utm_campaign` | string | utm_campaign campaign parameter |

## The Page object

The `page` context object supports the following keys:

| Key | Value Type | Description |
|---|---|---|
| `sku` | list of strings | List of product SKUs |
| `category_uri` | string | URI representing a category.<br><br>NOT YET SUPPORTED. |
| `ccid` | string | A category id representing a category provided as part of category feed. This field is ignored if category_uri is specified.<br><br>NOT YET SUPPORTED. |
| `category_id` | integer | Discover’s category id representing a category. This is an internal id assigned by Sitecore Discover. This field is ignored if category_uri or ccid is specified.<br><br>NOT YET SUPPORTED. |
| `locale_country` | string | The country specified as part of the locale context based on ISO 3166 Alpha 2 country code (e.g. us, ca, …). |

| | | |
|---|---|---|
| `locale_language` | string | The language specified as part of the locale context based on ISO 639-1 standard language codes (e.g. en, fr, ...). |

### The Fitment object

The `fitment` context object supports the following keys:

| Key | Value Type | Description |
|---|---|---|
| `ids` | array of string | Represents a fitment ids. |
| `items` | array of fitment objects | For rare case when you do not have fitment ids associated with fitments in your system. `ids` should be used where possible.<br>Note: You provide only `ids` or `items`, but not both. If both are provided, `ids` are honored, and `items` are ignored. |

## 2.1.3 The Filter object

The `filter` context object supports the following keys:

| Key | Value Type | Description |
|---|---|---|
| `<type>` | list of string | A list of filters to apply |

## 2.2 Examples

This section provides the examples of how to pass context information:

### 2.2.1 Passing product skus context for all widgets

The following sample shows you can pass product skus context for all widgets on a page:

```
        var rfk = window.rfk = window.rfk || [];

        rfk.push(["updateContext", {
            "context": {
                "page": {
                    "sku": ["12345", "34536"]
                }
            }
        }]);
```

### 2.2.2 Passing product skus context only for widgets rfkid_4 and rfkid_5

The  following shows a typical example of how to pass context SKUs for any page requiring a product/sku context like cart page and confirmation page widgets.

```
        var rfk = window.rfk = window.rfk || [];

        rfk.push(["updateContext", {
            "widget": {
                "rfkids": ["rfkid_4", "rfkid_5"]
            },
            "context": {
                "page": {
                    "sku": ["12345", "34536"]
                }
            }
        }]);
```

### 2.2.3 Passing category URI context only for widgets rfkid_4

**Note:**
This is not yet supported

The following shows a typical example of how to pass a category URI context for showing product recommendations when a website user hovers on a specific category a the menu. You can use any of the alternate ways to specify a category, but you must only use one way.

```
        var rfk = window.rfk = window.rfk || [];

        rfk.push(["updateContext", {
            "widget": {
                "rfkids": ["rfkid_4"]
            },
            "context": {
                "page": {
                    "category_uri": "/shoes"
                }
            }
        }]);
```

### 2.2.4 Passing locale context and product skus context only for a widget rfkid_5

The following shows a typical example of how to pass locale context and product skus context only for a widget:

```
        var rfk = window.rfk = window.rfk || [];

        rfk.push(["updateContext", {
            "widget": {
                "rfkids": ["rfkid_5"]
            },
            "context": {
               "page": {
                  "locale_country": "eu",
                  "locale_language": "en"
                  "sku": ["12345", "34536"]
                }
            }
        }]);
```

### 2.2.5 Passing locale context for all widgets in the page and product skus context only for specific widget

The following shows a typical example of how to pass context for all widgets in the page and product skus context only for widget rfkid_5:

```
        var rfk = window.rfk = window.rfk || [];

        rfk.push(["updateContext",
            {
              "context": {
                 "page": {
                    "locale_country": "eu",
                    "locale_language": "en"
                  }
              }
            },
            {
              "widget": {
                "rfkids": ["rfkid_5"]
              },
              "context": {
                "page": {
                    "sku": ["12345", "34536"]
                }
              }
            }
        ]);
```

### 2.2.6 Passing storeId context for all widgets on the page

The following shows a typical example of how to pass storeId context for all widgets on the page:

```
        var rfk = window.rfk = window.rfk || [];

        rfk.push(["updateContext", {
            "context": {
                "store": {
                    "id": "1"
                }
            }
        }]);
```

## 2.2.7 Passing user group context for all widgets on the page

The following shows a typical example of how to pass user group context for all widgets on the page:

```
        var rfk = window.rfk = window.rfk || [];

        rfk.push(["updateContext", {
            "context": {
                "user": {
                    "groups": ["1"]
                }
            }
        }]);
```

## 2.2.8 Passing campaign context for all widgets on the page

The following shows a typical example of how to pass campaign context for all widgets on the page:

```
        var rfk = window.rfk = window.rfk || [];

        rfk.push(["updateContext", {
            "context": {
                "campaign": {
                    "utm_source": "google",
                    "Utm_campaign": "test"
                }
            }
        }]);
```

## 2.2.9 Passing filter context for all widgets on the page

The following shows a typical example of how to pass filter context for all widgets on the page:

```
        var rfk = window.rfk = window.rfk || [];

        rfk.push(["updateContext", {
            "filter": {
                "color": ["Red", "Light Blue"],
                "gender": ["Female"]
            }
        }]);
```

## 2.2.10 Passing filter context for a specific widget

The following shows a typical example of how to pass filter context for widget rfkid_7:

```
        var rfk = window.rfk = window.rfk || [];

        rfk.push(["updateContext", {
            "widget": {
                "rfkids": ["rfkid_7"]
            },
            "filter": {
                "color": ["Red", "Light Blue"],
                "gender": ["Female"],
            }
        }]);
```

### 2.2.11 Passing a range (price) as filter context for a specific widget

The following shows a typical example of how to pass a range (price) as filter context for widget rfkid_7:

```
        var rfk = window.rfk = window.rfk || [];

        rfk.push(["updateContext", {
            "widget": {
                "rfkids": ["rfkid_7"]
            },
            "filter": {
                "final_price": {"from": 1616.94, "to": 9005},
            }
        }]);
```

### 2.2.12 Passing fitment Ids in context for specific widget

The following shows a typical example of how to pass fitment Ids in context for widget rfkid_7:

```
        var rfk = window.rfk = window.rfk || [];

        rfk.push(["updateContext", {
            "widget": {
                "rfkids": ["rfkid_7"]
            },
            "filter": {
                "final_price": {"from": 1616.94, "to": 9005},
            }
        }]);
```

### 2.2.13 Passing fitment objects in context for a specific widget

The following shows a typical example of how to pass fitment objects in context for widget rfkid_7:

**Note:**
Fitment IDs are preferred. You should only pass fitment objects (as provided in the fitment data feed) if you cannot provide proper fitment IDs.

```
        var rfk = window.rfk = window.rfk || [];

        rfk.push(["updateContext", {
            "widget": {
                "rfkids": ["rfkid_7"]
            },
            "context": {
               "fitment": {
                  "items": [
                     {
                       "make": "BMW",
                       "model": "F900R",
                       "year": "2018"
                     }
                  ]
               }
            }
        }]);
```

### 2.2.14 Passing fitment as filter context for a widget (Deprecated)

**Note:**
Sitecore Discover no longer supports fitments as filter.

```
        var rfk = window.rfk = window.rfk || [];

        rfk.push(["updateContext", {
            "widget": {
                "rfkids": ["rfkid_7"]
            },
            "filter": {
                "year": ["2005"],
                "make": ["Toyota"],
                "model": ["Camry"]
            }
        }]);
```

### 2.2.15 Passing filter context to make a given widget non-persistent

When a context is non-persistent, it is removed as soon as the widget refreshes. The following shows a typical example of how to pass filter context to make widget rfkid_7 non-persistent:

```
        var rfk = window.rfk = window.rfk || [];

        rfk.push(["updateContext", {
            "widget": {
                "rfkids": ["rfkid_7"]
            },
            "filter": {
                "color": ["Red", "Light Blue"],
                "gender": ["Female"]
            },
            "apply_once": true
        }]);
```

# Chapter 3 Clearing context

You can clear a context previously set for a page or a specific widget. To clear a context, you set its value to `null`.

For example, you can clear a store id context when a user logs out. If the store id is specified as context when a user logs in, typically, it must be removed from the context when the user logs out. The following are examples of multiple ways can clear context, based on the scenario.

You can clear the context with a push script that sets the the value of `"id"` inside `"store"` to `null`:

```
var rfk = window.rfk = window.rfk || [];

rfk.push(["updateContext", {
    "context": {
        "store": {
            "id": null
        }
    }
}]);
```

If there are no other keys inside the store context object, you can set the value of `"store"` to `null`:

```
var rfk = window.rfk = window.rfk || [];

rfk.push(["updateContext", {
    "context": {
        "store": null
    }
}]);
```

If there are no other keys inside the context object, you can set the value of `"context"` to `null`:

```
var rfk = window.rfk = window.rfk || [];

rfk.push(["updateContext", { "context": null }]);
```

**Note:**
If an object within `context` only contains `null` values, the object itself is removed from the context. In the following example, the two context objects are equivalent and both are removed:

| Example 1 | Example 2 |
|---|---|
| <pre>{<br>    "context": {<br>        "store": {<br>            "id": null<br>        }<br>    }<br>}</pre> | <pre>{<br>    "context": {<br>        "store": null<br>    }<br>}</pre> |

# Chapter 4 Timing for making rfk.push() calls

For specifying static context (content that does not change based on user interactions), Sitecore recommends placing the `rfk.push` script above the `Initjs` beacon. This ensures that the proper context is set before Sitecore Discover starts processing the widgets. Otherwise, Discover renders the widgets without the proper context, and once the context is specified using `rfk.push` script, content of the widgets will get refreshed.

# Chapter 5 Limitations

Sitecore Discover system usually assures products displayed in multiple recommendations widgets are not duplicated when the widgets are refreshed together. If you refresh individual widgets, then the widgets might display duplicate products.

This is only a consideration when you are displaying related widgets together, specifically, in the product description page, where you might display Similar Items and People also Viewed. If you refresh only one of these widgets, then these two widgets might show duplicate products. To avoid this, you must set a context to apply to for both the widgets.