

Why sap.ui.getCore().byId() Returns Undefined in SAP UI5?

 inui.io  UI5  Last updated on July 14, 2019

`sap.ui.getCore().byId()` returns undefined when it could not find a control with the to `sap.ui.getCore().byId()` passed ID. For example, `sap.ui.getCore().byId("bar")` returns undefined if the SAP UI5 application does not contain a control with the set ID *bar*.

But what if there is actually in an XML view a control with an set ID of *bar* like:

```
01 <!-- in your XML view -->
02
03 <mvc:View
04     id="foo"
05     controllerName="my.app.controller.FooController"
06     xmlns:mvc="sap.ui.core.mvc"
07     xmlns="sap.m">
08
09     <Button id="bar" text="Click this"/>
10
11 </mvc:View>
```

And in the controller of the view is in the `onAfterRendering()` hook a `sap.ui.getCore("bar")`. But it returns *undefined*.

```
01 // in your controller
02
03 sap.ui.define([
04     "sap/ui/core/mvc/Controller"
05 ], function (Controller) {
06     "use strict";
07
08     return Controller.extend("my.app.controller.FooController", {
09
10         onAfterRendering: function() {
11
12             // returns undefined
```

```

12 // Returns undefined
13 var oButton = sap.ui.getCore().byId("bar");
14
15 // works
16 oButton = this.byId("bar");
17
18 }
19
20 });
21
22 });

```

Why does *sap.ui.getCore("bar")* does not get the button with the id property bar? And why does *this.byId("bar")* works?

First of all, *this.byId()* is the way to get a control from a view in its controller. But what if you would like to get in a controller a control from another view then the controller's view? Because *this.byId()* can only get controls from the view which is assigned to the controller.

The first assumption would be to use *sap.ui.getCore().byId("bar")* with *bar* for the id property of the control from the other controller. But that does not work. And here is why:

IDs of Controls Are Prefixed with the IDs of Its Views in SAP UI5

The views prefix the control IDs with its own IDs. This has the advantages of that

1. the same control ID can be used in different views. For example, the ID *foo* for a button control could be used in View1 and in View2.
2. the same view can be used multiple times. For example, if the view is nested in another view or reused in the application.

To avoid ID collisions of controls, each view adds its own ID as a prefix to all its child controls. For example, a view has the ID *foo*. A button control in the view has the ID *bar*. Then the prefix that the view add to each of its controls is *foo-*. Therefore, the ID of the button control of the view is *foo-bar*.

And this is the difference of *this.byId()* and *sap.ui.getCore().byId()*: A view's *byId()* adds the prefix of the view automatically to the requested ID which is passed as argument to the method. But *sap.ui.getCore().byId()* does not add any prefix.

Behind the scenes *this.byId()* calls *sap.ui.getCore().byId()*. But *this.byId()* puts the argument *this.createId(id)* into *sap.ui.getCore().byId()*: *sap.ui.getCore().byId(this.createId(id))*. *id* is the String argument from *byId()*.

For example, *"bar"* in *byId("bar")*: The method *createId()* of the controller takes the ID of the controller's view and prefixes it to the passed ID from *byId()*. Hence, *this.byId("bar")* calls behind the scenes *sap.ui.getCore().byId(this.createId(id))* and ends up in *sap.ui.getCore().byId("foo-bar")*:

```
1  // this is what this.byId() calls behind the scenes
2
3  View.prototype.byId = function(sId) {
4
5      return sap.ui.getCore().byId(this.createId(sId));
6
7  };
```

To Use *sap.ui.getCore().byId()* the ID of the Control Needs to Be Prefixed

Therefore, *sap.ui.getCore().byId()* works only with the view prefix or nested views prefixes of the control added to the control's ID. But in contrast to *this.byId()* works *sap.ui.getCore().byId()* for controls too that are in other views then the controller's view.

```
01  <!-- in your XML view foo -->
02
03  <mvc:View
04      id="foo"
05      controllerName="myApp.controller.FooController"
06      xmlns:mvc="sap.ui.core.mvc"
07      xmlns="sap.m">
08
09      <!-- view ID is foo -->
10      <!-- button control ID is foo--bar -->
11      <Button id="bar" text="Click this"/>
12
13  </mvc:View>
```

```
01  // in your controller of foo view
```

```
01 // in your controller of foo view
```

```
02
03 sap.ui.define([
04     "sap/ui/core/mvc/Controller"
05 ], function (Controller) {
06     "use strict";
07
08     return Controller.extend("myApp.controller.FooController", {
09         onAfterRendering: function() {
10
11             // looks for ID foo--bar
12             // works
13             var oButton = this.byId("bar");
14
15             // looks for ID bar
16             // returns undefined
17             oButton = sap.ui.getCore().byId("bar");
18
19             // looks for ID foo--bar
20             // works
21             oButton = sap.ui.getCore().byId("foo--bar");
22
23         }
24     });
25 });
26
27
```

```
01 // in your controller of another view than foo
```

```
02
03 sap.ui.define([
04     "sap/ui/core/mvc/Controller"
05 ], function (Controller) {
06     "use strict";
07
08     return Controller.extend("myApp.controller.BarController", {
09         onAfterRendering: function() {
10
11             // looks for ID bar--bar
12             // returns undefined
13             var oButton = this.byId("bar");
14
15             // looks for ID bar
16             // returns undefined
17             oButton = sap.ui.getCore().byId("bar");
18
19
```

```

18
19         // looks for ID foo--bar
20         // works
21         oButton = sap.ui.getCore().byId("foo--bar");
22
23     }
24
25     });
26
27 });

```

Nested Controls Require Multiple Prefixes and How to Deactivate Auto Prefixing

If a control is nested in multiple views or further controls then it is necessary to prefix the IDs of those controls as well. For example, a view with the ID *foo* contains a simple form with the ID *bar* and the form contains an input with the ID *input1*. Then the ID of the input would be *foo-bar—input1*.

By the way, it is possible to deactivate the prefixing of IDs in the *manifest.json*:

```

01 // manifest.json
02
03 ...
04
05 sap.ui5 {
06
07     ...
08
09     autoPrefixId: false
10
11 }
12
13 ...

```

However, this is not recommended.



 No Comments

Get Exclusive SAP Tips

Learn about exclusive SAP tips and insights that are only shared with the private newsletter subscribers.

Email Address

Sign Up

LATEST POLL

Will SAP UI5 replace the SAP Web UI?

☐ Yes

☐ No

Submit

View results


ESSENTIAL RESSOURCES

[3 Most Important SAP Full Forms Explained – Do You Know Them?](#)

[77 SAP Acronyms List – What Stands for What in SAP?](#)

Leave a Reply

Your email address will not be published. Required fields are marked *

 Comment *

Fill out this field

 Name *

Fill out this field

 Email *

Please enter a valid email address.

Post Comment