# DEVEXTREME

## 1 Basics of DevExtreme

### 1.1 What is DevExpress and DevExtreme?

* DevExpress is component vendor for variety of controls for different technologies.

### 1.2 Nuget Package installation

* Install-Package DevExtreme -Version 21.1.3

### 1.3 jQuery widgets

* Any devextreme UI component must be placed inside a container.
* dxButton is a jQuery plugin provided by devextreme to each UI component.
* To configure the UI component u have to pass **an object** to the plugin. The props of this object act as mirror to the props of the UI component.
* $(“#container”).dxButton() is different from $(“#conatiner”).dxButton(“instance”).
* At first place dx create a dx-button-content class div inside the conatiner and returns the conatiner jQuery element.
* At second, dx returns the button widget instance using which u can call dx’s method or access its properties.

### 1.4 Call methods of dx widgets

* There are several ways to invoke a dx UI component methods.

#### 1.4.1 Using the widget instance:

*myWidget.methodName();*

#### 1.4.2 Using the jQuery interface:

*$(“#myWidgetElement”).dxMyWidget(“methodName”);*

#### 1.4.3 Using DOM Element:

* First get raw widget instance,
* Then get the dx instance of that widget,
* And the it is as same as 1.4.1.

*var widgetElement = document.getElementById(“myWidgetElement”);*

*var widgetInstance = DevExpress.ui.getWidgetInstance(widgetElement);*

*widgetInstance.methodName();*

#### 1.4.3 Using global widget instance method

For some widget, there are global method available on ‘DevExpress.ui’, which can be used to interact with widgets.

E.g., DevExpress.ui.notify to show notification for some widget.

### 1.5 DevExpress.ui namespace

* DevExpress.ui is a namespace that contains global utility methods and properties related to UI components.
* Task performed by DevExpress.ui:
  1. Notification
  2. Widget Registraction
  3. Widget Lookup
  4. Widget Theme

### 1.6 Accessing custom handlers

*let buttonInstance = $("#container").dxButton("instance");*

*buttonInstance.\_options.\_optionManager.\_options.onClick*

*document.getElementById(“#container”).jQuery351039286975644576462.dxButton.\_options.\_optionManager.\_options.onClick*

* Another way to register an event in devextreme is using .on method. It allows you to subscribe to events at runtime and even to attach several handlers to a single event. E.g., myWidgetInstance.on(“click”, function(){  
   alert(“click1”);  
  });

myWidgetInstance.on(“click”, function(){  
 alert(“click2”);  
});

* To manually locate this handlers:

*document.getElementById(“#container”).jQuery351039286975644576462*.dxButton.\_eventsStrategy.\_events.click.\_list

#### Unsubscribing from an event handler

To unsubscribe from an event handler attached using .on() method. You have to invoke .off() method. (Both invokes \_eventStrategy methods only).

myWidgetInstance.off(“click”) => empty the \_list attached to click event in \_eventStrategy instance.

myWidgetInstance.off(“click”, handlerRef) => pops out the handlerRef from the \_list attached to click event in \_eventStrategy instance.

To unsubscribe from handler attached using .onEventName property, u have to do, myWidgetInstance.option(“eventName”, undefined);

## 2 Editors

### 2.1 Checkbox

* A box which when selected by end user => shows that a particular feature has been enabled or chosen.

#### Configuration object for checkbox

accessKey – what key should set focus on the UI component.

accessKeyEnabled – when user interacts with the UI component then UI component should change its state.

Disabled – whether UI Component responds to user.

FocusStateEnabled – whether UI component can be focused using kb navigation.

onInitialized – when the widget has been fully initialized but before its content is rendered or ready for interaction. After this UI component is rendered on the screen. E.g, apply filters.

onContentReady – is executed when the widget is ready for interaction i.e., Ui is painted on the screen. E.g., highlight row.