Vuforia™

VUFORIA STUDIO ENTERPRISE – ANGULAR JS EXAMPLES

June 27, 2016

ANGULAR JS

- The JS code can be added by selecting the **Home.js** menu under **Home** menu in the navigation pane.
- Resources:
  - http://ionicframework.com/docs/api/
  - http://jquery.com/
  - https://angularjs.org/
  - http://tutorials.jenkov.com/angularjs/index.html

$SCOPE

Vuforia™

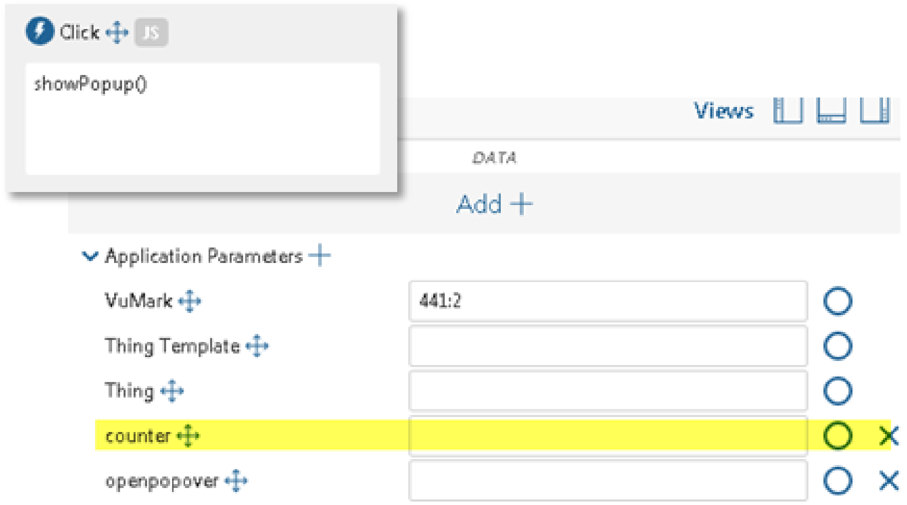
- **$scope** is the application object (the owner of application variables and functions). The controller creates two properties (variables) in the scope (firstName and lastName). The ng-model directives bind the input fields to the controller properties (firstName and lastName).
  - Example – activation by button from view:

  Write code in Button edit and then call function in JS:

```
// Triggered on a button click, or some other target
$scope.showPopup = function() {
//Write your code
};
```

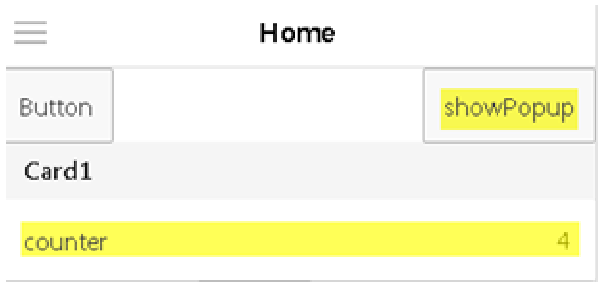
Example

```
//Assign value to Application parameter defined in DATA
$scope.app.params[‘counter'] = 0;

//Simple counter of the clicks on the button
$scope.showPopup = function() {
        $scope.app.params['counter'] = $scope.app.params['counter'] + 1;
 };
```



```
showPopup()
```

VuMark	441:2
Thing Template	
Thing	
counter	
openpopover	



counter	4

$SCOPE$WATCH

- **$scope$watch –** A watch means that AngularJS watches changes in the variable on the $scope object. The framework is "watching" the variable. Watches are created using the $scope.$watch() function which I will cover later in this text. When you register a watch you pass two functions as parameters to the $watch() function: 1)A value function 2)A listener function
  - Example:

  ```
  $scope.$watch(function() {},
            function() {}
           );
  ```

  The first function is the value function and the second function is the listener function.

  The value function should return the value which is being watched. AngularJS can then check the value returned against the value the watch function returned the last time. That way AngularJS can determine if the value has changed. Here is an example:

  ```
  $scope.$watch(function(scope) { return scope.data.myVar },
            function() {}
           );
  ```

  Notice how the value function takes the *scope* as parameter (without the $ in the name). Via this parameter the value function can access the *$scope* and its variables. The value function can also watch global variables instead if you need that, but most often you will watch a *$scope* variable.

$SCOPE$DIGEST AND $SCOPE$APPLY

Vuforia™

- **$scope$digest –** This function iterates through all watches and checks if any of the watched variables have changed. If a watched variable has changed, a corresponding listener function is called.

- **$scope.$apply()** function takes a function as parameter which is executed, and after that $scope.$digest() is called internally. That makes it easier for you to make sure that all watches are checked, and thus all data bindings refreshed. Here is an $apply() example:

```
$scope.$apply(function() {
   $scope.data.myVar = "Another value";
});
```

  The function passed to the *$apply()* function as parameter will change the value of *$scope.data.myVar.* When the function exits AngularJS will call the *$scope.$digest()* function so all watches are checked for changes in the watched values.

JS – USE TIMER SERVICES IN ANGULARJS

- **$timeout -** service can be used to call another JavaScript function after a given time delay. The *$timeout* service only schedules a single call to the function. For repeated calling of a function, see *$interval* later in this text. To use the *$timeout* service you must first get it injected into a controller function. Here is an example that injects the *$timeout* service into a controller function:

```
var myapp = angular.module("myapp", []);
myapp.controller("MyController", function($scope, $timeout){
});
```

  Notice the *$timeout* parameter of the controller function. Into this parameter the *$timeout* service will be injected by AngularJS, just like any other AngularJS service you would want to use in your controller function. Once the *$timeout* service is injected into your controller function, you can use it to schedule function calls. Here is an example on the *$scope* object that used the *$timeout* service to schedule a function call 3 seconds later:

```
var myapp = angular.module("myapp", []);
myapp.controller("DIController", function($scope, $timeout){
  $scope.callAtTimeout = function() {
    console.log("$scope.callAtTimeout - Timeout occurred");
  }
  $timeout( function(){ $scope.callAtTimeout(); }, 3000);
});
```

  Notice the function passed to the $timeout service. This function calls the callAtTimeout() function on the $scope object.

JS – USE TIMER SERVICES IN ANGULARJS

Vuforia™

- **$interval -** service is similar in function to the *$timeout* service, except it schedules a function for repeated execution with a time interval in between. To use the service you must have it injected into a controller function. Here is an example that injects the *$interval* service into a controller function:

```
var myapp = angular.module("myapp", []);
myapp.controller("MyController", function($scope, $interval){
});
```

  Once the $interval service is injected into your controller function, you can use it to schedule repeated function calls. Here is an example on the $scope object that used the $interval service to schedule a function call every 5 seconds:

```
var myapp = angular.module("myapp", []);
myapp.controller("DIController", function($scope, $interval){
   $scope.callAtInterval = function() {
      console.log("$scope.callAtInterval - Interval occurred");
   }
   $interval( function(){ $scope.callAtInterval(); }, 3000);
});
```

  The function passed to the $interval service calls the callAtInterval() function on the $scope object.

EXECUTING $DIGEST() AFTER THE SCHEDULED FUNCTION CALL

- If the function you schedule for execution makes changes to variables in the *$scope* object, or make changes to any other variable which your application is watching, your application needs to execute *$scope.$digest()* after the scheduled function call finishes. By default AngularJS already calls *$digest()* after the scheduled function call finishes, so you don't have to do that explicitly. You can, however, specify if AngularJS should not call *$digest(*) after the scheduled function call. If, for instance, your scheduled function call only updates an animation but does not change any $scope variables, then it is a waste of CPU time to call *$digest()* after the function finishes.

- Both *$timeout* and *$interval* have a third, optional parameter which can specify if the *$digest()* method is to be executed after the scheduled function finishes. Actually, the third parameter specifies if the call to the scheduled function should be done inside an *$apply()* call. Here is an example of how to use this third parameter:

  ```
  $interval( function(){ $scope.callAtInterval(); }, 3000, true);
  $interval( function(){ $scope.callAtInterval(); }, 3000, false);
  ```

  These two *$interval* examples both have a third parameter passed to the *$interval* service. This parameter can be either true or false. A value of true means that the scheduled function should be called inside an *$apply()* call. A value of false means that it should not be called inside an *$apply()* call (meaning *$digest()* will not get called after the scheduled function finishes).

ANGULAR.ELEMENT

Vuforia™

- **angular.element** is an alias for the jQuery function. If jQuery is not available, angular.element delegates to Angular's built-in subset of jQuery, called "jQuery lite" or jqLite. It is a function in module ng which wraps a raw DOM element or HTML string as a jQuery element.
  - Keep in mind that this function will not find elements by tag name / CSS selector. For lookups by tag name, try instead *angular.element(document).find(...)*

  Examples:

  angular.element(document.getElementById('3DModel-2')).scope().stop();

  Or

  angular.element(document.querySelector("[widget-id=button-6] button")).removeClass("play-button").addClass("pause-button");

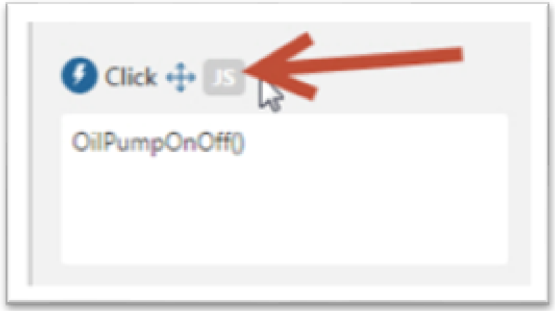
- Angular's jqLite provides only the following jQuery methods: (more info on http://api.jquery.com/)

- addClass()
- after()
- append()
- attr() - Does not support functions as parameters
- bind() - Does not support namespaces, selectors or eventData
- children() - Does not support selectors
- clone()
- contents()
- css() - Only retrieves inline-styles, does not call getComputedStyle(). As a setter, does not convert numbers to strings or append 'px', and also does not have automatic property prefixing.
- data()
- detach()
- empty()
- eq()
- find() - Limited to lookups by tag name
- hasClass()
- html()
- next() - Does not support selectors
- on() - Does not support namespaces, selectors or eventData
- off() - Does not support namespaces, selectors or event object as parameter
- one() - Does not support namespaces or selectors
- parent() - Does not support selectors
- prepend()
- prop()
- ready()
- remove()
- removeAttr()
- removeClass()
- removeData()
- replaceWith()
- text()
- toggleClass()
- triggerHandler() - Passes a dummy event object to handlers.
- unbind() - Does not support namespaces or event object as parameter
- val()
- wrap()

APPLICATION PARAMETERS

CALLING A FUNCTION FROM A BUTTON

Vuforia™



OilPumpOnOff()

- Create JavaScript Function and add call from Button Widget. Add and configure 2D Button Widget. Click on the JavaScript (JS) button to expose text entry box. Enter the JavaScript function name.

Example

```
//Function Called from button
$scope.OilPumpOnOff = function()
{
   if (OILPUMPVISIBLE)
   {
      OILPUMPVISIBLE = false;
      …
   }
   else
   {
      OILPUMPVISIBLE = true;
      …
   }
}
```

NAVIGATING TO A VIEW

Vuforia™

- //Navigate to a view

```
    $scope.HomeNav = function() {
            $scope.app.fn.navigate(‘Enter View Name Here');
    }
```

POPUP WINDOW ACTIVATION - ALERT

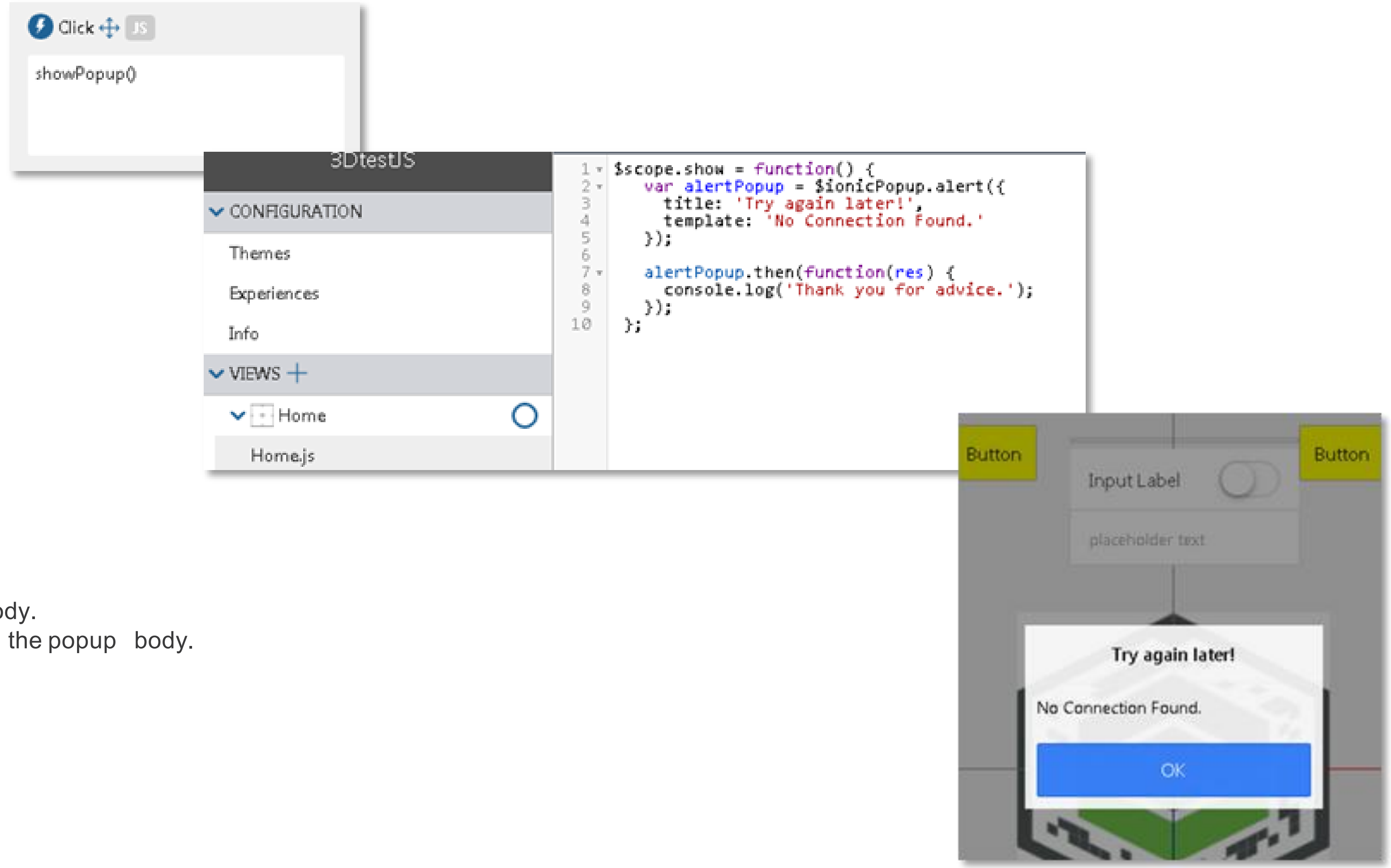
Vuforia™

- Popup service enables you to create popup windows which needs some action from user to continue. Basically there are three main popups:
  - alert box,
  - confirm box
  - prompt box.

- We can customize ionic **alert** popup to set following options.

```
{
  title: '', // String. The title of the popup.
  cssClass: '', // String, The custom CSS class name
  subTitle: '', // String (optional). The sub-title of the popup.
  template: '', // String (optional). The html template to place in the popup body.
  templateUrl: '', // String (optional). The URL of an html template to place in the popup   body.
  okText: '', // String (default: 'OK'). The text of the OK button.
  okType: '', // String (default: 'button-positive'). The type of the OK button.
}
```

Activation on click – Alert box



showPopup()

```
$scope.show = function() {
   var alertPopup = $ionicPopup.alert({
     title: 'Try again later!',
     template: 'No Connection Found.'
   });

   alertPopup.then(function(res) {
     console.log('Thank you for advice.');
   });
 };
```

Try again later!

No Connection Found.

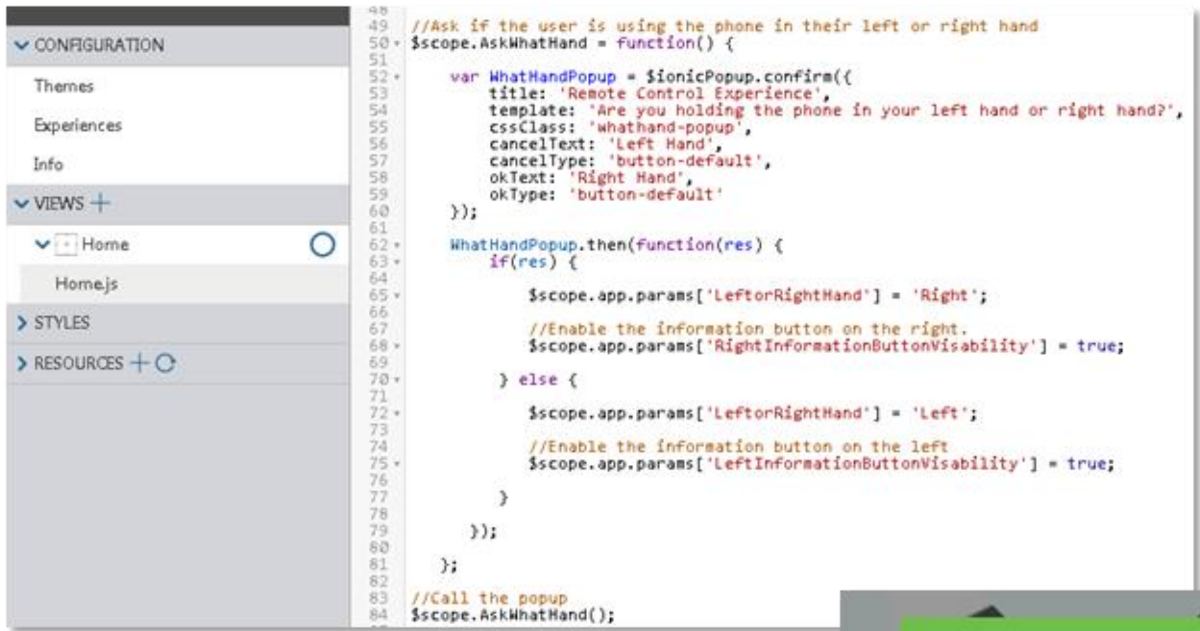
OK

POPUP WINDOW ACTIVATION - CONFIRM

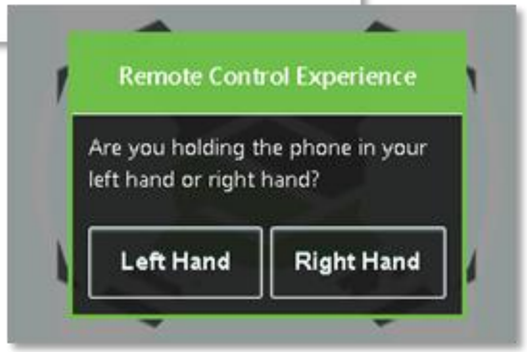
- Ionic confirm show a simple popup with "Cancel" and "Ok" button. We can get user selection on promise true if the user presses the OK button, and false if the user presses the Cancel button.
- We can customize ionic **confirm** popup to set following options.

```
{
   title: '', // String. The title of the popup.
   cssClass: '', // String, The custom CSS class name
   subTitle: '', // String (optional). The sub-title of the popup.
   template: '', // String (optional). The html template to place in the popup body.
   templateUrl: '', // String (optional). The URL of an html template to place in the popup   body.
   cancelText: '', // String (default: 'Cancel'). The text of the Cancel button.
   cancelType: '', // String (default: 'button-default'). The type of the Cancel button.
   okText: '', // String (default: 'OK'). The text of the OK button.
   okType: '', // String (default: 'button-positive'). The type of the OK button.
}
```

Activation on Start – Confirm box



```
//Ask if the user is using the phone in their left or right hand
$scope.AskWhatHand = function() {

    var WhatHandPopup = $ionicPopup.confirm({
        title: 'Remote Control Experience',
        template: 'Are you holding the phone in your left hand or right hand?',
        cssClass: 'whathand-popup',
        cancelText: 'Left Hand',
        cancelType: 'button-default',
        okText: 'Right Hand',
        okType: 'button-default'
    });

    WhatHandPopup.then(function(res) {
        if(res) {

            $scope.app.params['LeftorRightHand'] = 'Right';

            //Enable the information button on the right.
            $scope.app.params['RightInformationButtonVisability'] = true;

         } else {

            $scope.app.params['LeftorRightHand'] = 'Left';

            //Enable the information button on the left
            $scope.app.params['LeftInformationButtonVisability'] = true;

         }

      });

   };

//Call the popup
$scope.AskWhatHand();
```



Remote Control Experience

Are you holding the phone in your left hand or right hand?

POPUP WINDOW ACTIVATION - PROMT

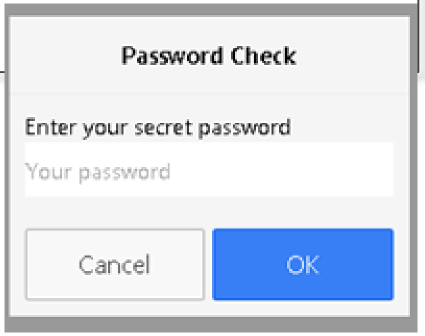
- Prompt - Ionic prompt show a simple prompt popup, with input, OK button, and Cancel button. We can get user action on promise when user insert any value into text box and click OK button. But if user presses Cancel button then promise return undefined.

- We can customize ionic **confirm** popup to set following options.

```
{
  title: '', // String. The title of the popup.
  cssClass: '', // String, The custom CSS class name
  subTitle: '', // String (optional). The sub-title of the popup.
  template: '', // String (optional). The html template to place in the popup body.
  templateUrl: '', // String (optional). The URL of an html template to place in the popup body.
  inputType: // String (default: 'text'). The type of input to use
  defaultText: // String (default: ''). The initial value placed into the input.
  maxLength: // Integer (default: null). Specify a maxlength attribute for the input.
  inputPlaceholder: // String (default: ''). A placeholder to use for the input.
  cancelText: // String (default: 'Cancel'. The text of the Cancel button.
  cancelType: // String (default: 'button-default'). The type of the Cancel button.
  okText: // String (default: 'OK'). The text of the OK button.
  okType: // String (default: 'button-positive'). The type of the OK button.
}
```

Prompt box

```
$scope.show = function() {
            $ionicPopup.prompt({
                        title: 'Password Check',
                        template: 'Enter your secret password',
                        inputType: 'password',
                        inputPlaceholder: 'Your password'
            }).then(function(res) {
            console.log('Your password is', res);
});
};
```



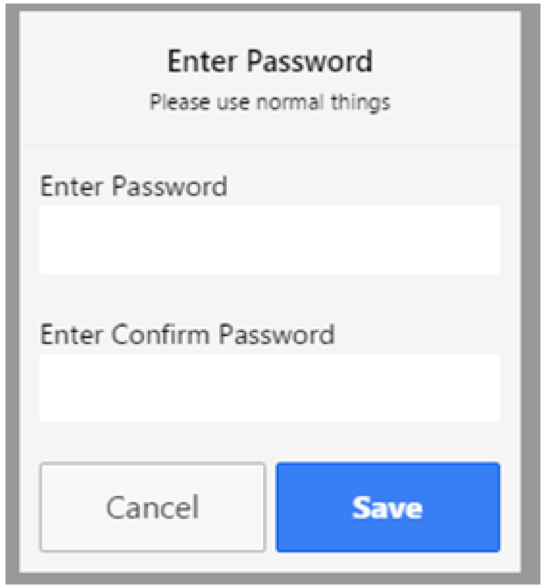
POPUP WINDOW ACTIVATION - SHOW

Vuforia™

- Show a complex popup. This is the master show function for all popups.
- We can customize ionic **show** popup to set following options.

```
{
  title: '', // String. The title of the popup.
  cssClass: '', // String, The custom CSS class name
  subTitle: '', // String (optional). The sub-title of the popup.
  template: '', // String (optional). The html template to place in the popup body.
  templateUrl: '', // String (optional). The URL of an html template to place in the popup   body.
  scope: null, // Scope (optional). A scope to link to the popup content.
  buttons: [{ // Array[Object] (optional). Buttons to place in the popup footer.
    text: 'Cancel',
    type: 'button-default',
    onTap: function(e) {
      // e.preventDefault() will stop the popup from closing when tapped.
      e.preventDefault();
    }
  }, {
    text: 'OK',
    type: 'button-positive',
    onTap: function(e) {
      // Returning a value will cause the promise to resolve with the given value.
      return scope.data.response;
    }
  }]
}
```

- A complex popup has a *buttons* array, with each button having a *text* and *type* field, in addition to an *onTap* function. The *onTap* function, called when the corresponding button on the popup is tapped, will by default close the popup and resolve the popup promise with its return value. If you wish to prevent the default and keep the popup open on button tap, call *event.preventDefault()* on the passed in tap event.
- Returns: *object* A promise which is resolved when the popup is closed. Has an additional *close* function, which can be used to programmatically close the popup.



POPOVER WINDOW ACTIVATION

- The Popover is a view that floats above an app’s content. Popovers provide an easy way to present or gather information from the user and are commonly used in the following situations:
  - Show more info about the current view
  - Select a commonly used tool or configuration
  - Present a list of actions to perform inside one of your views

```
// .fromTemplate() method
  var template = '<ion-popover-view><ion-header-bar> <h1 class="title">My Popover Title</h1> </ion-header-bar> <ion-content> Hello! </ion-content></ion-popover-view>';

  $scope.popover = $ionicPopover.fromTemplate(template, {
    scope: $scope
  });

  // .fromTemplateUrl() method
  $ionicPopover.fromTemplateUrl('my-popover.html', {
    scope: $scope
  }).then(function(popover) {
    $scope.popover = popover;
  });


  $scope.openPopover = function($event) {
    $scope.popover.show($event);
  };
  $scope.closePopover = function() {
    $scope.popover.hide();
  };
  //Cleanup the popover when we're done with it!
  $scope.$on('$destroy', function() {
    $scope.popover.remove();
  });
  // Execute action on hide popover
  $scope.$on('popover.hidden', function() {
    // Execute action
  });
  // Execute action on remove popover
  $scope.$on('popover.removed', function() {
    // Execute action
  });
});
```

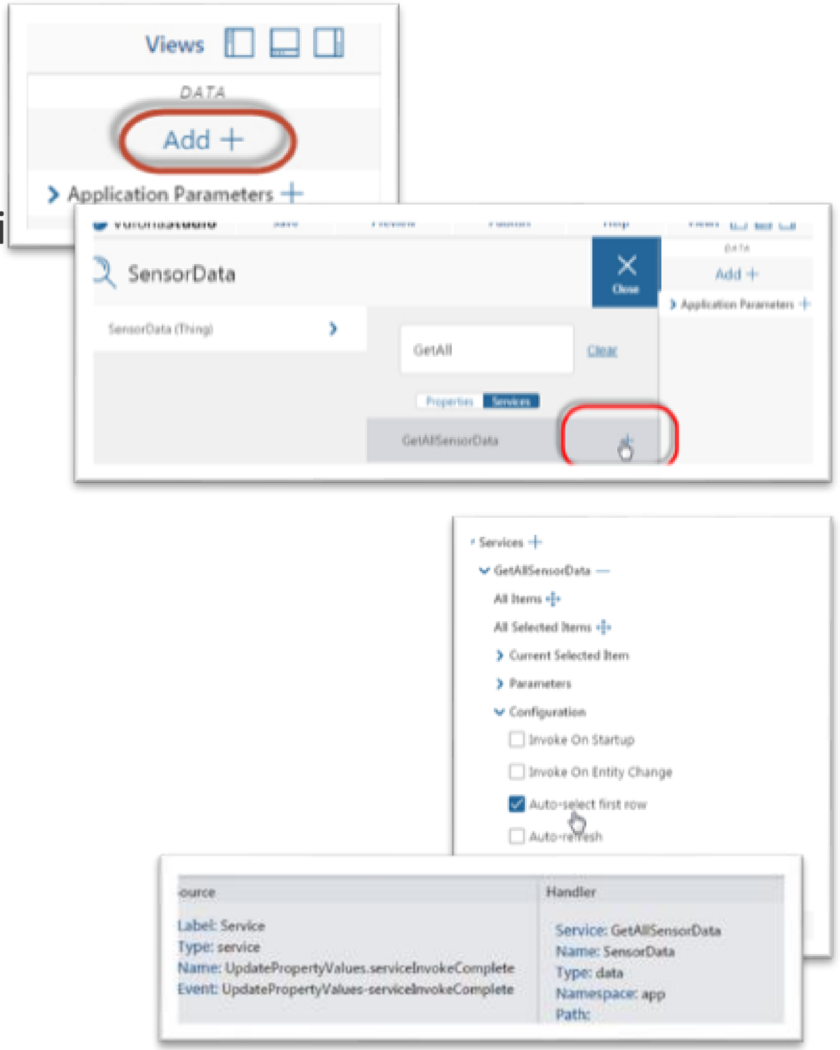
NEED TO ADD HOW TO ASSIGN POPOVER TO SPECIFIC OBJECT

- The info is in the iot overflow site

ADD SERVICE INTEGRATION

- **svc** (ThingWorx service) can be added and referenced. Add and Select ThingWorx Thing. Select Service to add. Configure or bind for execution. Events of other Services can BIND for service execution.

  A ThingWorx InfoTable result set can be loaded into a JSON structure for use i Java script.



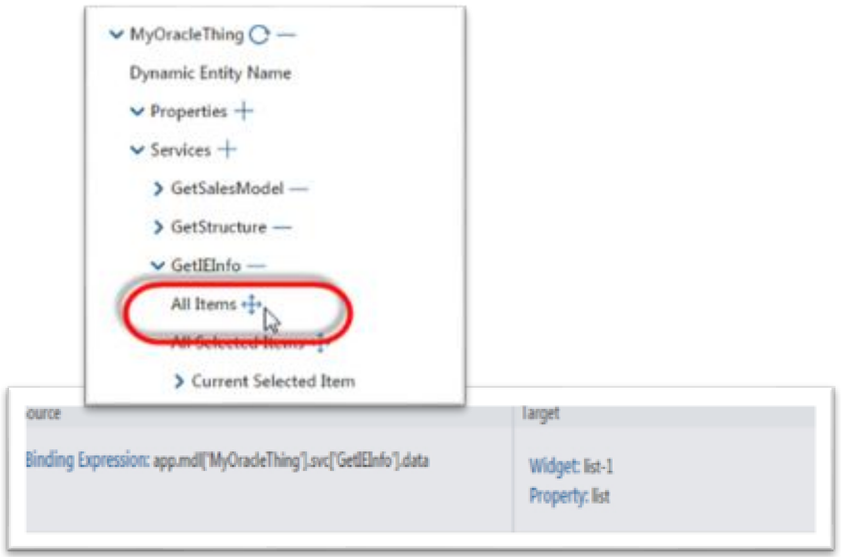
ource	Handler
Label: Service Type: service Name: UpdatePropertyValues.serviceInvokeComplete Event: UpdatePropertyValues-serviceInvokeComplete	Service: GetAllSensorData Name: SensorData Type: data Namespace: app Path:

Example

```
//Load Service InfoTable into JSON
var tmptext = $scope.app.mdl['SensorData'].svc['GetAllSensorData'];
var mjson = angular.fromJson (tmptext);

Loop Through JSON
for (var i = 0; i < mjson.data.length; ++i) {
    var detailrow = mjson.data[i];
    …
    …
    …
}
```

LOAD SERVICE RESULT INTO WIDGET & REFERENCE SERVICE VALUE

- **svc** (ThingWorx service) InfoTable result set can be Bound to a Widget for quick loading. Load service and Bind "All Items" to Widget.
- **svc** (ThingWorx service) InfoTable result set can be referenced in Javascript. Data can be referenced with the "data" tag. Current data elements are referenced with tag "current" followed by the name of the InfoTable result set.



ource	Target
Binding Expression: app.mdl['MyOracleThing'].svc['GetIEInfo'].data	Widget: list-1 Property: list

Example

```
//Get Result set attribute "Name" for currently selected data
$scope.app.mdl['MyOracleThing'].svc['GetSalesModel'].data.current['Name']
```

2D WIDGETS

BAR CHART

Vuforia™

BUTTONS – (TEXT, SHOW, HIDE, DISABLE, ENABLE) (ALL VERIFIED)

- Change the button text

```
//Change the text (replace ‘button-1’ with the ID of the button to be controlled)
$scope.view.wdg['button-1']['text'] = 'Enter text here';
```

- Show a button

```
//Show the button (replace ‘button-1’ with the ID of the button to be controlled)
$scope.view.wdg['button-1']['visible'] = true;
```

- Hide a button

```
//Hide the button (replace ‘button-1’ with the ID of the button to be controlled)
$scope.view.wdg['button-1']['visible'] = false;
```

- Disable a button

```
//Disable the button (replace ‘button-1’ with the ID of the button to be controlled)
$scope.view.wdg['button-1']['disabled'] = true;
```

- Enable a button

```
//Enable the button (replace ‘button-1’ with the ID of the button to be controlled)
$scope.view.wdg['button-1']['disabled'] = false;
```

BUTTONS – (MARGIN, ADD OR REMOVE CSS CLASS) (ALL VERIFIED)

- Change the margin

```
//Change the margin (replace ‘button-1’ with the ID of the button to be controlled)
$scope.view.wdg['button-1']['margin'] = '5px';
```

- Remove a css class

```
//Remove the class (replace ‘button-1’ with the ID of the button to be controlled)
angular.element(document.querySelector("[widget-id=button-1]  button")).removeClass("ExpandButton");
```

- Add a css class

```
//Remove the class (replace ‘button-1’ with the ID of the button to be controlled)
angular.element(document.querySelector("[widget-id=button-1]  button")).addClass("ExpandButtonSelected");
```

- Remove a css class and add a different class

```
//Remove the class (replace ‘button-1’ with the ID of the button to be controlled)
angular.element(document.querySelector("[widget-id=button-1]  button")).removeClass("ExpandButton").addClass("ExpandButtonSelected");
```

Not Documented Here: Studio ID, Friendly Name, Click

CARD – (HEADER, FOOTER, SHOW, HIDE, MARGIN, PADDING) (ALL VERIFIED)

Vuforia™

- Change header text
```
//Change the header text (replace ‘card-1’ with the ID of the card to be controlled)
$scope.view.wdg['card-1']['header'] = 'Enter text here';
```

- Change footer text
```
//Change the footer text (replace ‘card-1’ with the ID of the card to be controlled)
$scope.view.wdg['card-1']['footer'] = 'Enter text here';
```

- Show a card
```
//Show the card (replace ‘card-1’ with the ID of the card to be controlled)
$scope.view.wdg['card-1']['visible'] = true;
```

- Hide a card
```
//Hide the card (replace ‘card-1’ with the ID of the card to be controlled)
$scope.view.wdg['card-1']['visible'] = false;
```

- Change the margin
```
//Change the margin (replace ‘card-1’ with the ID of the card to be controlled)
$scope.view.wdg['card-1']['margin'] = '5';
```

- Change the padding
```
//Change the padding (replace ‘card-1’ with the ID of the card to be controlled)
$scope.view.wdg['card-1']['padding'] = '5';
```

CARD – (ADD OR REMOVE CSS CLASS) (ALL VERIFIED)

Vuforia™

- Remove a css class

  //Remove the class (replace ‘card-1’ with the ID of the card to be controlled)

  angular.element(document.querySelector("[widget-id=card-1] .card")).removeClass(“red");

- Add a css class

  //Remove the class (replace ‘card-1’ with the ID of the card to be controlled)

  angular.element(document.querySelector("[widget-id=card-1] .card")).addClass(“blue");

- Remove a css class and add a different class

  //Remove the class (replace ‘button-1’ with the ID of the button to be controlled)

  angular.element(document.querySelector("[widget-id=card-1] .card")).removeClass(“red").addClass(“blue");

Not Documented Here: Studio ID, Friendly Name

CHECKBOX – (ADD OR REMOVE CSS CLASS, CHECKED, NOT CHECKED, LABEL) (ALL VERIFIED)

Vuforia™

- Remove a css class
  //Remove the class (replace ‘checkbox-1’ with the ID of the checkbox to be controlled)
  angular.element(document.querySelector("[widget-id=checkbox-1] .checkbox")).removeClass(“red");

- Add a css class
  //Remove the class (replace ‘checkbox-1’ with the ID of the checkbox to be controlled)
  angular.element(document.querySelector("[widget-id=checkbox-1] .checkbox")).addClass(“blue");

- Remove a css class and add a different class
  //Remove the class (replace ‘checkbox-1’ with the ID of the checkbox to be controlled)
  angular.element(document.querySelector("[widget-id=checkbox-1] .checkbox")).removeClass(“red").addClass(“blue");

- Set the checkbox to be checked
  //Set to be checked (replace ‘checkbox-1’ with the ID of the checkbox to be controlled)
  $scope.view.wdg['checkbox-1']['value'] = true;

- Set the checkbox to not be checked
  //Set to not be checked (replace ‘checkbox-1’ with the ID of the checkbox to be controlled)
  $scope.view.wdg['checkbox-1']['value'] = false;

- Change the checkbox label
  //Show the button (replace ‘checkbox-1’ with the ID of the checkbox to be controlled)
  $scope.view.wdg['checkbox-1']['label'] = 'Enter text here';

CHECKBOX – (SHOW, HIDE, DISABLE, ENABLE, MARGIN, PADDING) (ALL VERIFIED)

- Show a checkbox
```
//Show the checkbox (replace ‘checkbox-1’ with the ID of the checkbox to be controlled)
$scope.view.wdg['checkbox-1']['visible'] = true;
```

- Hide a checkbox
```
//Hide the checkbox (replace ‘checkbox-1’ with the ID of the checkbox to be controlled)
$scope.view.wdg['checkbox-1']['visible'] = false;
```

- Disable a checkbox
```
//Disable the checkbox (replace ‘checkbox-1’ with the ID of the checkbox to be controlled)
$scope.view.wdg['checkbox-1']['disabled'] = true;
```

- Enable a checkbox
```
//Enable the checkbox (replace ‘checkbox-1’ with the ID of the checkbox to be controlled)
$scope.view.wdg['checkbox-1']['disabled'] = false;
```

- Change the margin
```
//Change the margin (replace ‘checkbox-1’ with the ID of the checkbox to be controlled)
$scope.view.wdg['checkbox-1']['margin'] = '5';
```

- Change the padding
```
//Change the padding (replace ‘checkbox-1’ with the ID of the checkbox to be controlled)
$scope.view.wdg['checkbox-1']['padding'] = '5';
```

Not Documented Here: Studio ID, Friendly Name, Value Changed, Selected, Deselected

FOOTER

GAUGE – (SHOW, HIDE, MIN, MAX, VALUE)

Vuforia™

- Show a gauge - verified

```
//Show the gauge (replace ‘gauge-1’ with the ID of the gauge to be controlled)
$scope.view.wdg['gauge-1']['visible'] = true;
```

- Hide a gauge - verified

```
//Hide the gauge (replace ‘gauge-1’ with the ID of the gauge to be controlled)
$scope.view.wdg['gauge-1']['visible'] = false;
```

- Set the minimum value – The label doesn’t update correctly

```
//Set the minimum value (replace ‘gauge-1’ with the ID of the gauge to be controlled)
$scope.view.wdg['gauge-1']['min'] = 25;
```

- Set the maximum value - verified

```
//Set the maximum value (replace ‘gauge-1’ with the ID of the gauge to be controlled)
$scope.view.wdg['gauge-1']['max'] = 125;
```

- Set the value - verified

```
//Set the value (replace ‘gauge-1’ with the ID of the gauge to be controlled)
$scope.view.wdg['gauge-1']['value'] = 85;
```

Not Documented Here: Class, Gauge Title, Width, Height, Studio Id, Friendly Name

GRID LAYOUT - (SHOW, HIDE) (ALL VERIFIED)

Vuforia™

- Show a grid

```
//Show the grid (replace ‘gridLayout-1’ with the ID of the grid to be controlled)
$scope.view.wdg['gridLayout-1']['visible'] = true;
```

- Hide a grid

```
//Hide the grid (replace ‘gridLayout-1’ with the ID of the label to be controlled)
$scope.view.wdg['gridLayout-1']['visible'] = false;
```

GRID LAYOUT – (ADD CSS CLASS TO COLUMNS IN GRID) (ALL VERIFIED)

Vuforia™

- Add a css class from column 1

  //Add the class (replace ‘gridLayout-1’ with the ID of the grid to be controlled)
  ```
  angular.element(document.querySelector('[widget-id="gridLayout-1"]  .col:nth-child(1)')).addClass("ExpandButtonSelectedBackground");
  ```

- Add a css class from column 2

  //Add the class (replace ‘gridLayout-1’ with the ID of the grid to be controlled)
  ```
  angular.element(document.querySelector('[widget-id="gridLayout-1"]  .col:nth-child(2)')).addClass("ExpandButtonSelectedBackground");
  ```

- Add a css class from column 3

  //Add the class (replace ‘gridLayout-1’ with the ID of the grid to be controlled)
  ```
  angular.element(document.querySelector('[widget-id="gridLayout-1"]  .col:nth-child(3)')).addClass("ExpandButtonSelectedBackground");
  ```

- Add a css class from column 4

  //Add the class (replace ‘gridLayout-1’ with the ID of the grid to be controlled)
  ```
  angular.element(document.querySelector('[widget-id="gridLayout-1"]  .col:nth-child(4)')).addClass("ExpandButtonSelectedBackground");
  ```

- Add a css class from column 5

  //Add the class (replace ‘gridLayout-1’ with the ID of the grid to be controlled)
  ```
  angular.element(document.querySelector('[widget-id="gridLayout-1"]  .col:nth-child(5)')).addClass("ExpandButtonSelectedBackground");
  ```

GRID LAYOUT – (REMOVE CSS CLASS FROM COLUMNS IN GRID) (ALL VERIFIED)

Vuforia™

- Remove a css class from column 1

  //Remove the class (replace 'gridLayout-1' with the ID of the grid to be controlled)

  ```
  angular.element(document.querySelector('[widget-id="gridLayout-1"]  .col:nth-child(1)')).removeClass("ExpandButtonSelectedBackground");
  ```

- Remove a css class from column 2

  //Remove the class (replace 'gridLayout-1' with the ID of the grid to be controlled)

  ```
  angular.element(document.querySelector('[widget-id="gridLayout-1"]  .col:nth-child(2)')).removeClass("ExpandButtonSelectedBackground");
  ```

- Remove a css class from column 3

  //Remove the class (replace 'gridLayout-1' with the ID of the grid to be controlled)

  ```
  angular.element(document.querySelector('[widget-id="gridLayout-1"]  .col:nth-child(3)')).removeClass("ExpandButtonSelectedBackground");
  ```

- Remove a css class from column 4

  //Remove the class (replace 'gridLayout-1' with the ID of the grid to be controlled)

  ```
  angular.element(document.querySelector('[widget-id="gridLayout-1"]  .col:nth-child(4)')).removeClass("ExpandButtonSelectedBackground");
  ```

- Remove a css class from column 5

  //Remove the class (replace 'gridLayout-1' with the ID of the grid to be controlled)

  ```
  angular.element(document.querySelector('[widget-id="gridLayout-1"]  .col:nth-child(5)')).removeClass("ExpandButtonSelectedBackground");
  ```

Not Documented Here: Studio Id, Friendly Name

HEADER

Vuforia™

LABEL – (TEXT, SHOW, HIDE, WRAP) (ALL VERIFIED)

Vuforia™

- Enter text in a label
  ```
  //Enter a text string in a label (replace ‘label-1’ with the ID of the label to be controlled)
  $scope.view.wdg['label-1']['text'] = 'Enter the text string';
  ```
- Show a label
  ```
  //Show the label (replace ‘label-1’ with the ID of the label to be controlled)
  $scope.view.wdg['label-1']['visible'] = true;
  ```
- Hide a label
  ```
  //Hide the label (replace ‘label-1’ with the ID of the label to be controlled)
  $scope.view.wdg['label-1']['visible'] = false;
  ```
- Set label text to wrap
  ```
  //Wrap label text (replace ‘label-1’ with the ID of the label to be controlled)
  $scope.view.wdg['label-1']['wrap'] = true;
  ```
- Set label text to not wrap
  ```
  //Disable wrap text (replace ‘label-1’ with the ID of the label to be controlled)
  $scope.view.wdg['label-1']['wrap'] = false;
  ```

Not Documented Here: Class, Padding, Margin, Studio Id, Friendly Name

LIST

REPEATER

RESOURCE IMAGE – (SOURCE, BACKGROUND COLOR, SHOW, HIDE)

- Set the image location - verified

  ```
  //Set the image location (replace ‘resourceImage-1’ with the ID of the image to be controlled)
  $scope.view.wdg['resourceImage-1']['imgsrc'] = 'app/resources/Default/vu_alert1.svg';
  ```

- Set the background color- the color doesn’t update in the preview

  ```
  //Set the background color(replace ‘resourceImage-1’ with the ID of the image to be controlled)
  $scope.view.wdg['resourceImage-1']['backgroundcolor'] = 'red';
  ```

- Show the image - verified

  ```
  //Show the image (replace ‘resourceImage-1’ with the ID of the image to be controlled)
  $scope.view.wdg['resourceImage-1']['visible'] = true;
  ```

- Hide the image - verified

  ```
  //Hide the image (replace ‘resourceImage-1’ with the ID of the image to be controlled)
  $scope.view.wdg['resourceImage-1']['visible'] = false;
  ```

Not Documented Here: Class, Width, Height, Alignment, Padding, Studio ID, Friendly Name

SELECT

Vuforia™

SLIDER – (MIN, MAX, STEP, VALUE) (ALL VERIFIED)

- Set the minimum value

```
//Set the slider minimum value (replace ‘slider-1’ with the ID of the slider to be controlled)
$scope.view.wdg['slider-1']['min'] = 40;
```

- Set the maximum value

```
//Set the slider maximum value (replace ‘slider-1’ with the ID of the slider to be controlled)
$scope.view.wdg['slider-1']['max'] = 90;
```

- Set the step value

```
//Set the slider step value (replace ‘slider-1’ with the ID of the slider to be controlled)
$scope.view.wdg['slider-1']['step'] = 5;
```

- Set the value

```
//Set the slider value (replace ‘slider-1’ with the ID of the slider to be controlled)
$scope.view.wdg['slider-1']['value'] = 88;
```

SLIDER – (SHOW, HIDE, DISABLE, ENABLE) (ALL VERIFIED)

Vuforia™

- Show a slider

```
//Show the slider (replace ‘slider-1’ with the ID of the slider to be controlled)
$scope.view.wdg['slider-1']['visible'] = true;
```

- Hide a slider

```
//Hide the slider (replace ‘slider-1’ with the ID of the slider to be controlled)
$scope.view.wdg['slider-1']['visible'] = false;
```

- Disable a slider

```
//Disable the slider (replace ‘slider-1’ with the ID of the slider to be controlled)
$scope.view.wdg['slider-1']['disabled'] = true;
```

- Enable a slider

```
//Enable the slider (replace ‘slider-1’ with the ID of the slider to be controlled)
$scope.view.wdg['slider-1']['disabled'] = false;
```

Not Documented Here: Class, Icon Left of Slider, Icon Right of Slider, Margin, Padding, Studio ID, Friendly Name, Value Changed

SPINNER – (SHOW, HIDE) (ALL VERIFIED)

Vuforia™

- Show a spinner
```
//Show the spinner (replace ‘spinner-1’ with the ID of the spinner to be controlled)
$scope.view.wdg['spinner-1']['visible'] = true;
```

- Hide a spinner
```
//Hide the spinner (replace ‘spinner-1’ with the ID of the spinner to be controlled)
$scope.view.wdg['spinner-1']['visible'] = false;
```

Not Documented Here: Spinner Type, Studio ID, Friendly Name

TAB STRIP – (SHOW, HIDE)

- Show the tab strip

```
//Show the tab strip (replace ‘tabs-1’ with the ID of the tab strip to be controlled)
$scope.view.wdg['tabs-1']['visible'] = true;
```

- Hide the tab strip

```
//Hide the tab strip (replace ‘tabs-1’ with the ID of the spinner to be controlled)
$scope.view.wdg['tabs-1']['visible'] = false;
```

Not Documented Here: Class, Tab Orientation, Tab Padding, Tab Strip Class, Margin, Studio ID, Friendly Name, Tab Click

TEXT AREA – (TEXT, PLACEHOLDER, LABEL, # OF ROWS, MAX LENGTH) (ALL VERIFIED)

Vuforia™

- Set the text value

```
//Set the text (replace 'textArea-1' with the ID of the textArea to be controlled)
$scope.view.wdg['textArea-1']['text'] = 'Enter the text here';
```

- Set the placeholder text

```
//Set the placeholder (replace 'textArea-1' with the ID of the textArea to be controlled)
$scope.view.wdg['textArea-1']['placeholder'] = 'Enter the placeholder text here';
```

- Set the label text

```
//Set the label (replace 'textArea-1' with the ID of the textArea to be controlled)
$scope.view.wdg['textArea-1']['label'] = 'Enter the label text here';
```

- Set the number of rows

```
//Set the number of rows (replace 'textArea-1' with the ID of the textArea to be controlled)
$scope.view.wdg['textArea-1']['rows'] = 4;
```

- Set the maximum length

```
//Set the maximum number of characters that can be entered (replace 'textArea-1' with the ID of the
textArea to be controlled)
$scope.view.wdg['textArea-1']['maxlength'] = 15;
```

TEXT AREA – (SHOW, HIDE, DISABLE, ENABLE) (ALL VERIFIED)

Vuforia™

- Show the text area field
  ```
  //Show the text area (replace ‘textArea-1’ with the ID of the textArea to be controlled)
  $scope.view.wdg['textArea-1']['visible'] = true;
  ```
- Hide the text area field
  ```
  //Hide the text area (replace ‘textArea-1’ with the ID of the textArea to be controlled)
  $scope.view.wdg['textArea-1']['visible'] = false;
  ```
- Disable the text area field
  ```
  //Disable the text area (replace ‘textArea-1’ with the ID of the textArea to be controlled)
  $scope.view.wdg['textArea-1']['disabled'] = true;
  ```
- Enable the text area field
  ```
  //Enable the text area (replace ‘textArea-1’ with the ID of the textArea to be controlled)
  $scope.view.wdg['textArea-1']['disabled'] = false;
  ```

Not Documented Here: Class, Read Only, Padding, Margin, Studio ID, Friendly Name, Value Changed

TEXT INPUT – (TEXT, PLACEHOLDER, LABEL) (ALL VERIFIED)

- Set the text value
```
//Set the text (replace ‘textInput-1’ with the ID of the textInput to be controlled)
$scope.view.wdg['textInput-1']['text'] = 'Enter the text here';
```
- Set the placeholder text
```
//Set the placeholder (replace ‘textInput-1’ with the ID of the textInput to be controlled)
$scope.view.wdg['textInput-1']['placeholder'] = 'Enter the placeholder text here';
```
- Set the label text
```
//Set the label (replace ‘textInput-1’ with the ID of the textInput to be controlled)
$scope.view.wdg['textInput-1']['label'] = 'Enter the label text here';
```

TEXT INPUT – (SHOW, HIDE, DISABLE, ENABLE) (ALL VERIFIED)

Vuforia™

- Show the text input field
  ```
  //Show the text input (replace ‘textInput-1’ with the ID of the textInput to be controlled)
  $scope.view.wdg['textInput-1']['visible'] = true;
  ```

- Hide the text input field
  ```
  //Hide the text input (replace ‘textInput-1’ with the ID of the textInput to be controlled)
  $scope.view.wdg['textInput-1']['visible'] = false;
  ```

- Disable the text input field
  ```
  //Disable the text input (replace ‘textInput-1’ with the ID of the textInput to be controlled)
  $scope.view.wdg['textInput-1']['disabled'] = true;
  ```

- Enable the text input field
  ```
  //Enable the text input (replace ‘textInput-1’ with the ID of the textInput to be controlled)
  $scope.view.wdg['textInput-1']['disabled'] = false;
  ```

Not Documented Here: Class, Type, Align, Padding, Margin, Studio ID, Friendly Name, Value Changed

TIME SERIES CHART

TOGGLE – (VALUE, LABEL) (ALL VERIFIED)

Vuforia™

- Set the value to true
  //Set the value to true (replace ‘toggle-1’ with the ID of the toggle to be controlled)
  $scope.view.wdg['toggle-1']['value'] = true;

- Set the value to false
  //Set the value to false(replace ‘toggle-1’ with the ID of the toggle to be controlled)
  $scope.view.wdg['toggle-1']['value'] = false;

- Enter text in the toggle label
  //Enter a text string in the toggle label (replace ‘toggle-1’ with the ID of the toggle to be controlled)
  $scope.view.wdg['toggle-1']['label'] = 'Enter the text string';

TOGGLE – (SHOW, HIDE, DISABLE, ENABLE) (ALL VERIFIED)

Vuforia™

- Show a toggle

  ```
  //Show the toggle (replace ‘toggle-1’ with the ID of the toggle to be controlled)
  $scope.view.wdg['toggle-1']['visible'] = true;
  ```

- Hide a toggle

  ```
  //Show the toggle (replace ‘toggle-1’ with the ID of the toggle to be controlled)
  $scope.view.wdg['toggle-1']['visible'] = false;
  ```

- Disable a toggle

  ```
  //Disable the toggle (replace ‘toggle-1’ with the ID of the toggle to be controlled)
  $scope.view.wdg['toggle-1']['disabled'] = true;
  ```

- Enable a toggle

  ```
  //Enable the toggle (replace ‘toggle-1’ with the ID of the toggle to be controlled)
  $scope.view.wdg['toggle-1']['disabled'] = false;
  ```

Not Documented Here: Class, Margin, Studio ID, Click

TOGGLE BUTTON - (PRESSED, IMAGE PRESSED, IMAGE NOT PRESSED) (ALL VERIFIED)

- Set the button to be pressed

```
//Set the button to be pressed (replace ‘toggleButton-1’ with the ID of the button to be controlled)
$scope.view.wdg['toggleButton-1']['pressed'] = true;
```

- Set the button to not be pressed

```
//Set the button to not be pressed (replace ‘toggleButton-1’ with the ID of the button to be controlled)
$scope.view.wdg['toggleButton-1']['pressed'] = false;
```

- Set the image for the button when pressed

```
//Set the image when pressed (replace ‘toggleButton-1’ with the ID of the button to be controlled)
$scope.view.wdg['toggleButton-1']['src'] = 'app/resources/Default/vu_alert1.svg';
```

- Set the image for the button when not pressed

```
//Set the image when not pressed (replace ‘toggleButton-1’ with the ID of the button to be controlled)
$scope.view.wdg['toggleButton-1']['srcnotpressed'] = 'app/resources/Default/vu_alert1.svg';
```

TOGGLE BUTTON – (BACKGROUND COLOR, BACKGROUND COLOR PRESSED, WIDTH, HEIGHT) (ALL VERIFIED)

- Set the background color

  //Set the background color; this field supports RGB color, hexadecimal colors, or color names. (replace ‘toggleButton-1’ with the ID of the button to be controlled)

  $scope.view.wdg['toggleButton-1']['backgroundColor'] = 'red';

- Set the background color when pressed

  //Set the background color when pressed; this field supports RGB color, hexadecimal colors, or color names. (replace ‘toggleButton-1’ with the ID of the button to be controlled)

  $scope.view.wdg['toggleButton-1']['backgroundColorPressed'] = 'blue';

- Set the width of the button

  //Set the width (replace ‘toggleButton-1’ with the ID of the button to be controlled)

  $scope.view.wdg['toggleButton-1']['width'] = '100px';

- Set the height of the button

  //Set the height (replace ‘toggleButton-1’ with the ID of the button to be controlled)

  $scope.view.wdg['toggleButton-1']['height'] = '100px';

TOGGLE BUTTON – (SHOW, HIDE, DISABLE, ENABLE) (ALL VERIFIED)

Vuforia™

- Show a button
  ```
  //Show the button (replace ‘toggleButton-1’ with the ID of the button to be controlled)
  $scope.view.wdg['toggleButton-1']['visible'] = true;
  ```

- Hide a button
  ```
  //Hide the button (replace ‘toggleButton-1’ with the ID of the button to be controlled)
  $scope.view.wdg['toggleButton-1']['visible'] = false;
  ```

- Disable a button
  ```
  //Disable the button (replace ‘toggleButton-1’ with the ID of the button to be controlled)
  $scope.view.wdg['toggleButton-1']['disabled'] = true;
  ```

- Enable a button
  ```
  //Enable the button (replace ‘toggleButton-1’ with the ID of the button to be controlled)
  $scope.view.wdg['toggleButton-1']['disabled'] = false;
  ```

Not Documented Here: Class, Studio ID, Friendly Name, Click, Pressed, Unpressed

URL IMAGE – (URL, BACKGROUND COLOR, SHOW, HIDE) (ALL VERIFIED)

- Set the image url

```
//Set the image url(replace ‘urlImage-1’ with the ID of the image to be controlled)
$scope.view.wdg['urlImage-1']['src'] = 'Enter URL here';
```

- Set the background color

```
//Set the background color; this field supports RGB color, hexadecimal colors, or color names. (replace
‘urlImage-1’ with the ID of the image to be controlled)
$scope.view.wdg['urlImage-1']['backgroundColor'] = 'red';
```

- Show the image

```
//Show the image (replace ‘urlImage-1’ with the ID of the image to be controlled)
$scope.view.wdg['urlImage-1']['visible'] = true;
```

- Hide the image

```
//Hide the image (replace ‘urlImage-1’ with the ID of the image to be controlled)
$scope.view.wdg['urlImage-1']['visible'] = false;
```

Not Documented Here: Class, Width, Height, Alignment, Padding, Studio ID, Friendly Name

VALUE DISPLAY – (VALUE, LABEL, SHOW, HIDE) (ALL VERIFIED)

- Set the value

```
//Set the  value (replace ‘valueDisplay-1’ with the ID of the valueDisplay to be controlled)
$scope.view.wdg['valueDisplay-1']['value'] = 'Enter value';
```

- Set the label

```
//Set the label (replace ‘valueDisplay-1’ with the ID of the valueDisplay to be controlled)
$scope.view.wdg['valueDisplay-1']['label'] = 'Enter value';
```

- Show the display

```
//Show the value display (replace ‘valueDisplay-1’ with the ID of the valueDisplay to be controlled)
$scope.view.wdg['valueDisplay-1']['visible'] = true;
```

- Hide the display

```
//Hide the value display (replace ‘valueDisplay-1’ with the ID of the valueDisplay to be controlled)
$scope.view.wdg['valueDisplay-1']['visible'] = false;
```

- Not Documented Here: Class, Padding, Margin, Type, Studio ID, Friendly Name

3D WIDGETS

3D CONTAINER

3D IMAGE

3D LABEL

3D MODEL - (SOURCE, SCALE, X COORDINATE, Y COORDINATE, Z COORDINATE)

- Set the 3D Model

```
//Set the 3D Model (replace ‘3DModel-1’ with the ID of the 3D Model to be controlled)
$scope.view.wdg['3DModel-1']['src'] = 'path to model';
```

- Set the scale

```
//Set the scale of the 3D Model (replace ‘3DModel-1’ with the ID of the 3D Model to be controlled)
$scope.view.wdg['3DModel-1']['scale'] = 1.0;
```

- Set the X coordinate

```
//Set the 3D Model X coordinate (replace ‘3DModel-1’ with the ID of the 3D Model to be controlled)
$scope.view.wdg['3DModel-1']['x'] = 0;
```

- Set the Y coordinate

```
//Set the 3D Model Y coordinate (replace ‘3DModel-1’ with the ID of the 3D Model to be controlled)
$scope.view.wdg['3DModel-1']['y'] = 0;
```

- Set the Z coordinate

```
//Set the 3D Model Z coordinate (replace ‘3DModel-1’ with the ID of the 3D Model to be controlled)
$scope.view.wdg['3DModel-1']['z'] = 0;
```

3D MODEL – (X ROTATION, Y ROTATION, Z ROTATION, SHOW, HIDE)

Vuforia™

- Set the X rotation

```
//Set the 3D Model X rotation (replace ‘3DModel-1’ with the ID of the 3D Model to be controlled)
$scope.view.wdg['3DModel-1']['rx'] = 0;
```

- Set the Y rotation

```
//Set the 3D Model Y rotation (replace ‘3DModel-1’ with the ID of the 3D Model to be controlled)
$scope.view.wdg['3DModel-1']['ry'] = 0;
```

- Set the Z rotation

```
//Set the 3D Model Z rotation (replace ‘3DModel-1’ with the ID of the 3D Model to be controlled)
$scope.view.wdg['3DModel-1']['rz'] = 0;
```

- Show the 3D model

```
//Show the 3D Model (replace ‘3DModel-1’ with the ID of the 3D Model to be controlled)
$scope.view.wdg['3DModel-1']['visible'] = true;
```

- Hide the 3D model

```
//Hide the 3D Model (replace ‘3DModel-1’ with the ID of the 3D Model to be controlled)
$scope.view.wdg['3DModel-1']['visible'] = false;
```

3D MODEL – (OCCLUDING, ALWAYS ON TOP)

- Enable the “Occluding” property

```
//Enable the occluding property (replace ‘3DModel-1’ with the ID of the 3D Model to be controlled)
$scope.view.wdg['3DModel-1']['occlude'] = true;
```

- Disable the “Occluding” property

```
//Disable the occluding property (replace ‘3DModel-1’ with the ID of the 3D Model to be controlled)
$scope.view.wdg['3DModel-1']['occlude'] = false;
```

- Enable the “Always on top” property

```
//Enable the always on top property (replace ‘3DModel-1’ with the ID of the 3D Model to be controlled)
$scope.view.wdg['3DModel-1']['decal'] = true;
```

- Disable the “Always on top” property

```
//Disable the always on top property (replace ‘3DModel-1’ with the ID of the 3D Model to be controlled)
$scope.view.wdg['3DModel-1']['decal'] = false;
```

3D SENSOR – (TEXT, IMAGE, FONT, FONT SIZE) (ALL VERIFIED)

- Set the text
  ```
  //Set the text shown on the sensor (replace ‘3DSensor-1’ with the ID of the 3D Sensor to be controlled)
  $scope.view.wdg['3DSensor-1']['text'] = 'Enter text here';
  ```

- Set the image
  ```
  //Set the image (replace ‘3DSensor-1’ with the ID of the 3D Sensor to be controlled)
  $scope.view.wdg['3DSensor-1']['src'] = 'app/resources/Default/vu_alert1.svg';
  ```

- Set the font
  ```
  //Set the font on the sensor (replace ‘3DSensor-1’ with the ID of the 3D Sensor to be controlled)
  $scope.view.wdg['3DSensor-1']['font'] = 'Arial';
  ```

- Set the font size
  ```
  //Set the size of the font (replace ‘3DSensor-1’ with the ID of the 3D Sensor to be controlled)
  $scope.view.wdg['3DSensor-1']['fontsize'] = '60px';
  ```

3D SENSOR

THINGMARK – (THINGMARK, X COORDINATE, Y COORDINATE, Z COORDINATE) (ALL VERIFIED)

- Set the ThingMark
  - Best practice is to bind the ThingMark property on the ThingMark widget to the ThingMark application parameter, but here is the syntax to set it programmatically)

```
//Enter the ThingMark domain and id (replace ‘thingMark-1’ with the ID of the ThingMark to be controlled)
$scope.view.wdg['thingMark-1']['markerId'] = 287:1;
```

- Set the X coordinate

```
//Set the ThingMark X coordinate (replace ‘thingMark-1’ with the ID of the ThingMark to be controlled)
$scope.view.wdg['thingMark-1']['x'] = 0;
```

- Set the Y coordinate

```
//Set the ThingMark Y coordinate (replace ‘thingMark-1’ with the ID of the ThingMark to be controlled)
$scope.view.wdg['thingMark-1']['y'] = 0;
```

- Set the Z coordinate

```
//Set the ThingMark Z coordinate (replace ‘thingMark-1’ with the ID of the ThingMark to be controlled)
$scope.view.wdg['thingMark-1']['z'] = 0;
```

THINGMARK – (X ROTATION, Y ROTATION, Z ROTATION, ALWAYS ON TOP) (ALL VERIFIED)

Vuforia™

- Set the X rotation

  ```
  //Set the ThingMark X rotation (replace ‘thingMark-1’ with the ID of the ThingMark to be controlled)
  $scope.view.wdg['thingMark-1']['rx'] = -90;
  ```

- Set the Y rotation

  ```
  //Set the ThingMark Y rotation (replace ‘thingMark-1’ with the ID of the ThingMark to be controlled)
  $scope.view.wdg['thingMark-1']['ry'] = 0;
  ```

- Set the Z rotation

  ```
  //Set the ThingMark Z rotation (replace ‘thingMark-1’ with the ID of the ThingMark to be controlled)
  $scope.view.wdg['thingMark-1']['rz'] = 0;
  ```

- Enable the “Always on top” property

  ```
  //Enable the always on top property (replace ‘thingMark-1’ with the ID of the ThingMark to be controlled)
  $scope.view.wdg['thingMark-1']['decal'] = true;
  ```

- Disable the “Always on top” property

  ```
  //Disable the always on top property (replace ‘thingMark-1’ with the ID of the ThingMark to be controlled)
  $scope.view.wdg['thingMark-1']['decal'] = false;
  ```

THINGMARK – (TRACKING INDICATOR)

- Enable the “Display tracking indicator” property – I think this works
  //Enable the tracking indicator on the ThingMark (replace ‘thingMark-1’ with the ID of the ThingMark to be controlled)
  $scope.view.wdg['thingMark-1']['tracking-indicator'] = true;

- Disable the “Display tracking indicator” property – I think this works
  //Disable the tracking indicator on the ThingMark (replace ‘thingMark-1’ with the ID of the ThingMark to be controlled)
  $scope.view.wdg['thingMark-1']['tracking-indicator'] = false;