

JavaScript for PL/SQL Developers

ā'pěks alpe adria

APEX Alpe Adria

21.04.2023

A regional Oracle APEX conference for everyone

#aaapeks23

Lino Schildenfeld

@LinoSchild
lschilde.blogspot.com

APEX Office hours

April 20, 2023 14:00 - 15:00 UTC

Start Times Around the World

Subscribe

 Actions

Part 1: What's new in Oracle APEX 23.1

In this session, we will present and demonstrate the marquee features of APEX 23.1.

Catch up on the recording for our [last](#) session!

Don't miss our [next](#) live session!

This session: <https://asktom.oracle.com/pls/apex/asktom.search?oh=20407>

Oracle Database 23c Free Version Now Available to Developers





Mentor and Speaker Hub

- Our goal is to *connect* speakers with mentors to assist in *preparing* technical sessions and *improving* presentation skills

Interested? Read more and get in touch

<https://mashprogram.wordpress.com>

AUSOUG APEX News



<https://www.ausoug.org.au/news/apex-special-series-2023/>

My story



AUSOUG



- AUSOUG APEX webinars
- NZ APEX meetup organizer

FOEX



APEX RD
research & development

- APEX World Member of the Month
- Conference speaker
- Blogger

@LinoSchilde

www.linkedin.com/in/lorschilde



Oracle ACE
Pro

Motivation - Modern apps

- User-friendly
- Engaging apps
- User have expectations
- Provide immediate response and feedback
- Put it another way, users expect **web pages to be very interactive**
- They need to work on any device
- This applies to LOW CODE frameworks too

HTML

HTML DOM Basics

```
<!DOCTYPE html>
<html>
  <head>
    <title>Page Title</title>
  </head>

  <body>
    <h1>A Heading</h1>
  </body>
</html>
```

```
<!DOCTYPE html>
<html class="no-js page-1000 app-4500" lang="en"> event scroll
  > <head> ... </head>
  > <body class=" apex-theme-standard has-envBanner--left"> event ...
      <!--
        [if lte IE 9]><div id="outdated-browser">You are using an out
        web browser. For a list of supported browsers, please referen
        Oracle Application Express Installation Guide.</div><![endif]
      -->
      <noscript>You must run this product with JavaScript enabled.
      </noscript>
    > <div class="a-EnvBanner a-EnvBanner--accent-14" role="region"
          label="DEMO APEX.ORACLE.COM"> ... </div>
```

```
window = {
  document: {
    head: {
      ...
    },
    body: {
      ...
    },
  },
};
```

- To work with an HTML page in JavaScript, we create a representation of this structure - Document Object Model (DOM).
- **DOM** is a JavaScript object
- It lets us access and manipulate the elements in an HTML page
- APEX is no exception

DOM manipulation

- Add/remove classes
- Modify attributes
- Element manipulation
- Remove or add elements
- Change css styles

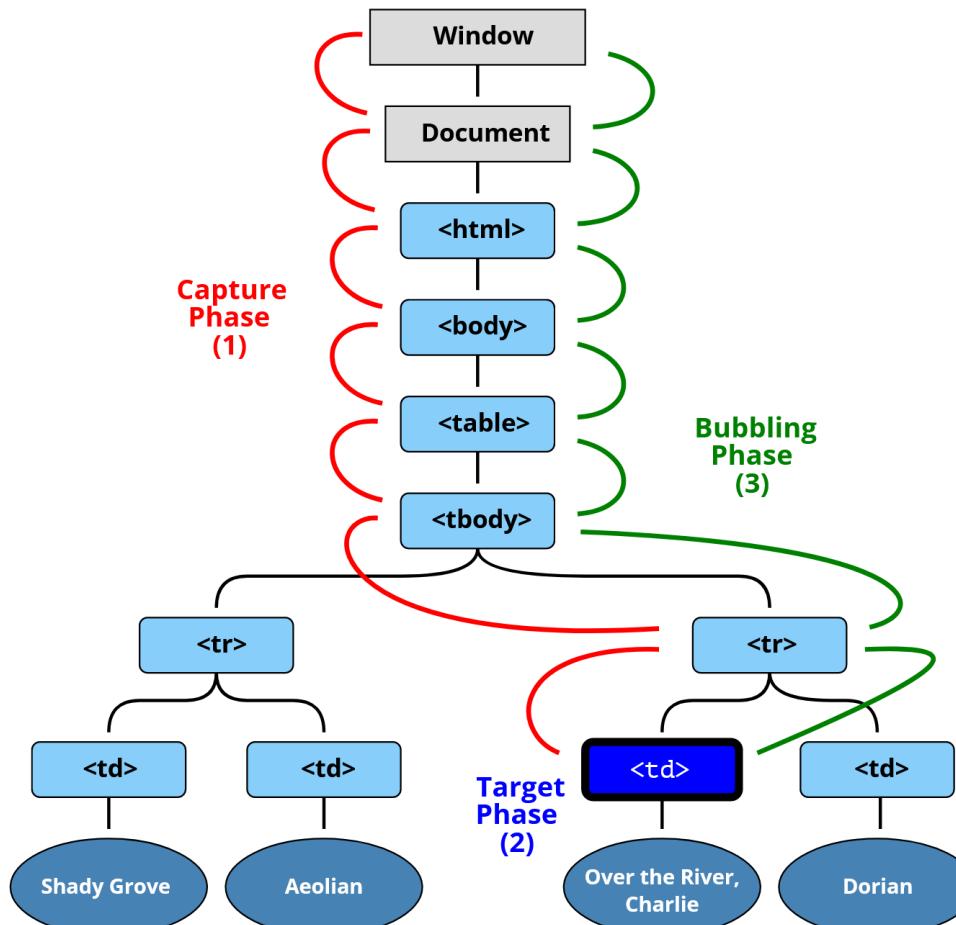
```
<html>
  <head>
    <script>
      // run this function when the document is loaded
      window.onload = function() {

        // create a couple of elements in an otherwise empty HTML page
        const heading = document.createElement("h1");
        const heading_text = document.createTextNode("Big Head!");
        heading.appendChild(heading_text);
        document.body.appendChild(heading);
      }
    </script>
  </head>
  <body>
  </body>
</html>
```

Common terms

Object	Description
window	Global object (browser window)
document	part of the DOM API
console	Our dbms_output
JSON	Methods provided for JSON manipulation
setTimeout	or timing the actions
.....	
apex.items	Some APEX stuff
apex.submit('DELETE');

APEX page too



jQuery web API

- Makes it more easy to manipulate a page of HTML
- Library we include on the page
- Fast, small, and feature-rich JavaScript library
- It is included in APEX too
- Wrappers
- Simplified APIs

`addClass,removeClass,attr,removeAttr, val, html, text,
append,prepend,remove,empty`

```
<input id="input-test" type="input" name="input">  
<script>  
    $('#input-test').on('change', function() {  
        console.log('it changed!');  
    });  
</script>
```

SELECTORS

- Like:

```
$ ('#id') - ID selector  
$ ('.class') - class selector  
$ ('td') - element selector  
$ ("input[name~='man']") - attribute selector
```

...

...

```
1 <!doctype html>  
2 <html lang="en">  
3   <head>  
4     <meta charset="utf-8">  
5     <title>attributeContains demo</title>  
6     <script src="https://code.jquery.com/jquery-3.5.0.js"></script>  
7   </head>  
8   <body>  
9  
10  <input name="man-news">  
11  <input name="milkman">  
12  <input name="letterman2">  
13  <input name="newmilk">  
14  
15  <script>  
16  $( "input[name*='man']" ).val( "has man in it!" );  
17  </script>  
18  
19  </body>  
20  </html>
```

DOM vs JQuery

Example:

ID selector

DOM `document.getElementById('myReport')`

JQUERY `$('#myReport')`

JavaScript

PLSQL vs JS

- JavaScript is the only language to change data on client side in Browser.
- SQL & PLSQL manipulate the data
- It adds lots to overall “feeling” in modern apps
- JavaScript is everywhere

LOWCODE

- We do not have to be gurus
- It take only few lines to make things work
- Or do we?

PLSQL vs JS

```
1 declare
2     foo NUMBER;
3 begin
4     foo := 2;
5     dbms_output.put_line(foo);
6
7 declare
8     foo_local number := foo;
9 begin
10    dbms_output.put_line('Entering nested block');
11    dbms_output.put_line(foo_local);
12    foo_local := 3;
13    dbms_output.put_line(foo_local);
14    dbms_output.put_line('Exiting nested block');
15 end;
16 dbms_output.put_line(foo);
17
18 --wrong type will not work
19 foo := 'It does not work';
20
21 end;
```

PLSQL scope example

```
2
Entering nested block
2
3
Exiting nested block
2
```

PLSQL vs JS

JS scope example

```
1  var globalTest = 'Greetings';
2
3  ▶ function outer(){
4      var maxVal = 100;
5
6  ▶   function inner(){
7      var val;|
8
9      val = maxVal;
10     console.log(val);
11
12     val = 'No issues with this';
13     console.log(val);
14   }
15 }
```

PLSQL vs JS

PLSQL

- Scope are PLSQL blocks
- Not case sensitive
- Data type does not change
- Variables are always in declaration section
- Procedures and functions supported
- Each variable needs a name and a type
- Many data types

clob, blob, records,
collections, number, varchar2,
boolean, char, date,
timestamp...

JS

- Scopes are functions
- It is case sensitive
- Variables can be defined anywhere
- There are no procedures
- Data type can be swapped at any time
- Variable only needs a name

JS primitive vs object types

```
1  var myString = 'Test';
2  var myNumber = 2;
3
4  var myBoolean = true;
5
6  var myArray = ['a', 'b'];
7  var myObject = {name:'Lino', role:'Admin'};
8
9  //using NEW constructor
10 var myDate = new Date(2021, 2, 2);
11
```

JS data types

Data Types	Description	Example
String	represents textual data	'hello' , "hello world!" etc
Number	an integer or a floating-point number	3 , 3.234 , 3e-2 etc.
BigInt	an integer with arbitrary precision	900719925124740999n , 1n etc.
Boolean	Any of two values: true or false	true and false
undefined	a data type whose variable is not initialized	let a;
null	denotes a null value	let a = null;
Symbol	data type whose instances are unique and immutable	let value = Symbol('hello');
Object	key-value pairs of collection of data	let student = { };

JS functions

- Functions are like any other data type
- Can be assigned to a variable
- Passed around as parameters
- Returned from other functions

```
1  ▼ function myTest1(fun){  
2      console.log('Yeah!!!!');  
3      fun();  
4  };  
5  
6  ▼ function myTest2(){  
7      console.log('Neah!!!');  
8  };  
9  
10 myTest1(myTest2);
```

Yeah!!!!

Neah!!!

AJAX

- It stands for Asynchronous JavaScript and XML
- Web pages can send HTTP requests to the server and receive a response
- Without needing to reload the entire page
- We use it in APEX on many places
- Report refresh being a typical example

```
// Async examples: https://javascript.info/fetch
async function getJson() {
    const url = "f?p=&APP_ID.:0:&SESSION.:APPLICATION_PROCESS=app_get_clob:NO:RP:::"
    const response = await fetch(url);
    const text = await response.text();
    gJSON = text;
    //alert(gJSON).
```

DEVELOPER TOOLS

- Good text editor and browser
- Similar to SQL Developer
- Open it with F12 or right click Inspect (Ctrl + Shift + I)
- Inspecting elements

The screenshot shows a developer tools interface with a sidebar on the left containing navigation links like 'Dev Tools', 'Work Orders', 'Warehouse', 'Items', 'Item Types', 'Item Categories', and 'Create Multiple Items'. The main area displays a list of items with columns for 'Work Order', 'Build Id', 'Asset Tag', 'Vote Id', 'Destination Location', 'Description', 'Item Count', 'Complete', 'Sealed', 'Current Location', 'Parent Build', and 'Incomplete tasks'. Two items are selected: 'Sealed' and 'Complete', both marked with a star icon. Below the list is a toolbar with buttons for 'View Debug', 'Debug', 'Page Info', 'Quick Edit/A', 'Theme Roller', and settings. The bottom of the screen features a toolbar with icons for 'Inspector', 'Console', 'Debugger', 'Network', 'Style Editor', 'Performance', 'Memory', 'Storage', 'Accessibility', and 'Application'. A search bar at the bottom left says 'Search HTML'. The bottom right shows an open 'Inspector' panel with tabs for 'Layout', 'Computed', 'Changes', and 'Fonts'. The 'Computed' tab is active, displaying CSS rules for the selected elements. One rule is highlighted: 'element { touch-action: pan-y; }'. Other visible rules include 'body { background-color: #f5f5f5; color: #363636; font-family: 'Helvetica Neue', 'Segoe UI', Helvetica, Arial, sans-serif; }'.

Oracle Developers

Perfect candidates

30%

Search for books, videos and tutorials to learn APEX and get a personal Workspace on apex.oracle.com. **It's free!**

ORACLE DATABASE

60%

Learn the basic principles of the Oracle Database, SQL, and PL/SQL.

80%

You can develop in APEX without knowing HTML, CSS, and JavaScript but you'll discover a whole new world after learning it.

!FINAL STEP!

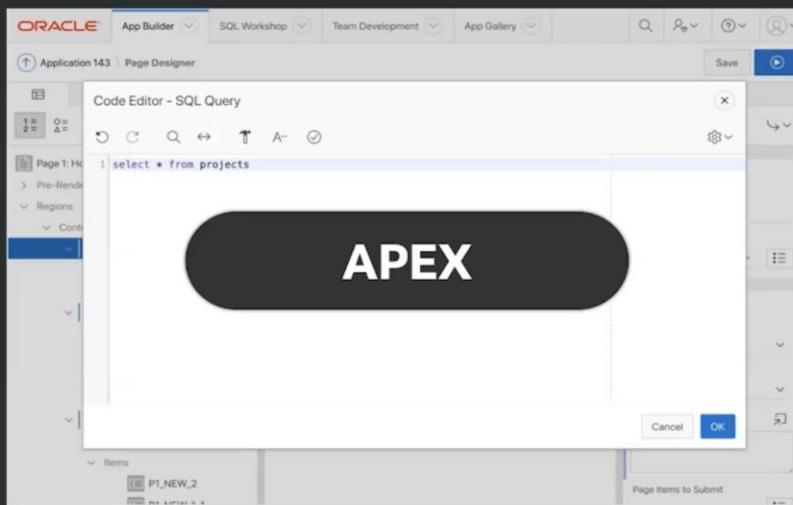
100%

To achieve everything you can think of, you should learn jQuery and Dynamic Actions.
There are no limits for you!

LOWCODE = APEX

- Focus on business solutions
- Improve productivity
- Deliver quality and consistency
- Directly leverage scalability, security attributes on Oracle DB
- Utilize all of your Oracle skills

Oracle APEX lets you build enterprise apps 20x faster with 100x less code



VS



APEX and JavaScript

- Inline on a page
- Static application file or as application theme referenced files
- Anywhere where we can do HTML in APEX

Where can we find JS in APEX:

- Dynamic actions
- Interactive grids configurations (JET Chart, Cards and other regions)
- Actions interface
- Inclusion of additional JS library like Numeral.js
- Plugins
- Client side validations
- AJAX

APEX developer perspective

The screenshot shows the Oracle Application Express (APEX) Page Designer interface. The top navigation bar includes tabs for 'Layout', 'Page Search', and 'Help'. The main area displays the 'Error debug' page configuration.

Page Identification:

- Name: Error debug
- Alias: (empty)
- Title: Error debug
- Page Group: - Select -

Appearance:

- User Interface: DESKTOP
- Page Mode: Normal
- Page Template: Standard
- Template Options: Use Template Defaults
- CSS Classes: (empty)
- Media Type: (empty)

Navigation Menu:

- Override User Interface Level: (checkbox)

Page Components (Content Body):

- Info
- Maintain Employee
- Region Body
 - P55_ROWID
 - P55_EMPNO
 - P55_ENAME
 - P55_JOB
 - P55_MGR
 - P55_HIREDATE
 - P55_SAL
- P55_BONUS
- P55_COMM
- P55_DEPTNO
- P55_LOCATION
- P55_NO_OF_EMPLOYEES



Edit Page 3



js/demo.js

Cancel

Delete

Save Changes

✓ File saved and minified X

Directory

js ⋮

* File Name

demo.js



Reference

#APP_FILES#js/demo#MIN
#.js

Mime Type

text/javascript



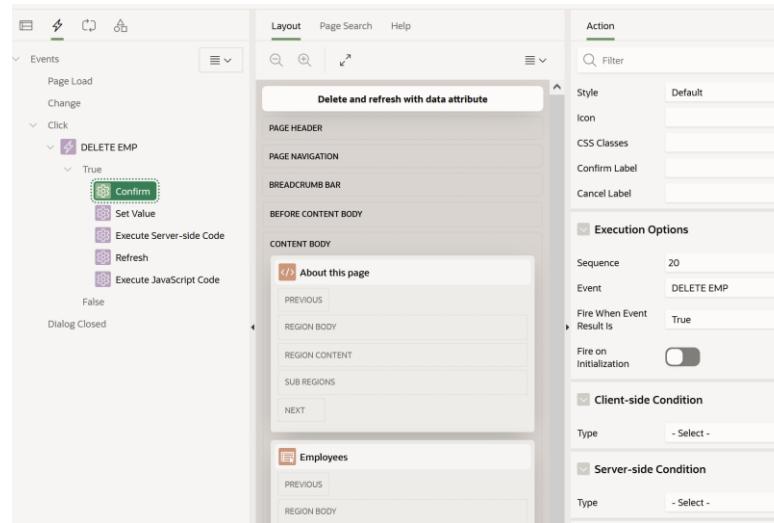
```
1 var value = apex.item( "P1_ITEM" ).element;  
2  
3 apex.message.showPageSuccess("Employee #"+ $v("P3_EMPNO")+" removed.");
```

Including JS on a page

>21.2 APEX version

DYNAMIC ACTIONS

- Easiest way to include JS in APEX
- Declarative => little coding with little to no knowledge of JS
- **Most likely will not cover everything JavaScript can do**
- When complexity grows put it into files
- Code Sharing between pages
- We can now create and edit static JS files from APEX Builder

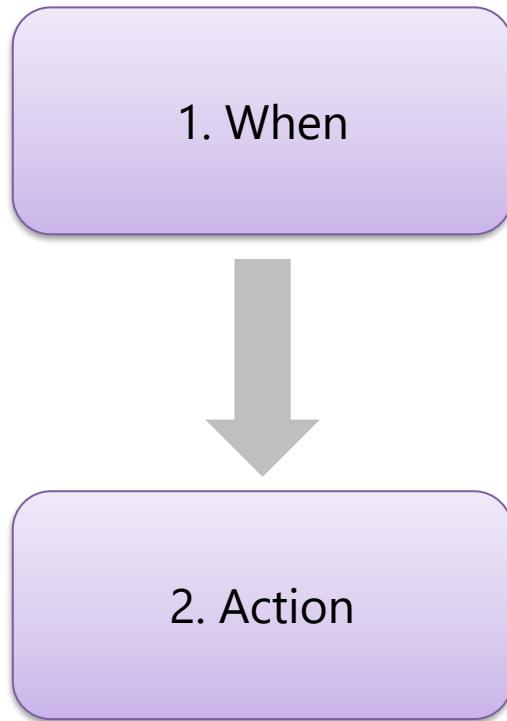


DYNAMIC ACTIONS

```
1 <html>
2 <body>
3
4 <h2>JavaScript addEventListener()</h2>
5
6 <button id="myBtn">Try it</button>
7
8 <script>
9 document.getElementById("myBtn").addEventListener("click", myFunction);
10
11 function myFunction() {
12 | alert ("Hello World!");  

13 }
14 </script>
15
16 </body>
17 </html>
```

DYNAMIC ACTIONS 101



1. When the Dynamic Action should happen or "fire"

- Browser Events
- Framework Events
- Component Events

OPTIONAL client side condition

2. What you want to do when an event takes place

- True Actions
- False Actions

3*. Affected elements or Scope

DYNAMIC ACTIONS

@JuergenSchuster



Juergen @JuergenSchuster · Nov 19

I'm afraid I have become a #orclapex Dynamic Action junkie 🏙 100 DAs on one page, anybody else who has this addiction?

The screenshot shows a list of dynamic actions defined on a single page. The actions are categorized into several groups:

- Parent Search: Employee, Parent, Parent Search: New Parent, Parent Search: New Minioness, Parent Search: New Passions, Parent Search: New Organisation, Parent Search: New Impassions, Parent Search: New Passions/Minioness Change, Parent Search: Organisation: Passions, Parent Search: Organisation: New Pass, Play: Pig Trigger, Endless: Endless, Endless: Shulle, Endless: New Endless Layout, Endless: New Endless Layouts, Minion: DPH, Early: New Minioness, Early: UnpassMinion, Live more - Mark the current Submenu of Last visited action, Change:Builder:Start on Price:Class, Change:Price:Class on Builder:Class, Set O_REPORTING_REPORTNAME to New Resource Type 262, Parent Search: Blue:Op:Blue, During:Create, Parent Search: NewEnd:American, Page:Change:Op:Blue.

Q 10

↑ 1

♡ 20



@Matt FOEX

```
1 select da_count_tab.*  
2   from (select aap.page_id  
3         ,(select count(*)  
4           from apex_application_page_da aapd  
5             where aapd.application_id = 200  
6               and aapd.page_id = aap.page_id) as count_da  
7        ,(select count(*)  
8           from apex_application_page_da acts aapda  
9             where aapda.application_id = 200  
10                and aapda.page_id = aap.page_id) as count_da_actions  
11      from apex_application_pages aap  
12        where aap.application_id = 200) da_count_tab  
13   where da_count_tab.count_da > 100  
14   order by da_count_tab.count_da desc
```

PAGE_ID	COUNT_DA	COUNT_DA_ACTIONS
1	350	819
2	379	995
3	457	996
4	369	928
5	478	973
6	468	937
7	477	952
8	490	954
9	469	961
10	493	946
11	461	959
12	494	950
13	474	979
14	501	483
15	473	943
16	486	934
17	497	922
18	487	883
19	489	931
20	472	935
21	491	926
22	492	929
23	1500	517
24	488	868
25	495	902
26	496	904
27	302	488

DYNAMIC ACTIONS

- Consolidate
- Do not be afraid to use manual code

```
/**  
 * @function show  
 **/  
show: function () {  
    if (fab.e.navigation.back.stack.length > 0 ||  
        !apex.item("P0_BACK_URL").isEmpty()) {  
        $(".fabe-back-all").show();  
        $(".brand-logo img").hide();  
    }  
  
    if (!apex.item("P0_BACK_URL_MOBILE").isEmpty()) {  
        $(".fabe-back-mobile").show();  
        $(".brand-logo img").addClass("hide-on-med-and-down");  
    }  
},  
  
/**  
 * @function hide  
 * @example  
 * fab.e.navigation.back.hide();  
 **/  
hide: function () {  
    console.log(fab.e.navigation.back.stack.length === 0,  
               apex.item("P0_BACK_URL").isEmpty());  
  
    if (fab.e.navigation.back.stack.length === 0 &&  
        apex.item("P0_BACK_URL").isEmpty()) {  
        $(".fabe-back-all").hide();  
        $(".brand-logo img").show();  
    }  
  
    if (fab.e.navigation.back.stack.length === 0 &&  
        apex.item("P0_BACK_URL_MOBILE").isEmpty()) {  
        $(".fabe-back-mobile").hide();  
        $(".brand-logo img").removeClass("hide-on-med-and-down");  
    }  
},
```

P61_DEPTNO

Dynamic Actions

New

True

- Alert
- Alert
- Alert
- Show
- Show
- Show
- Disable
- Disable
- Disable
- Execute JavaScript Code
- Execute JavaScript Code
- Execute JavaScript Code

False

Modularize JS code

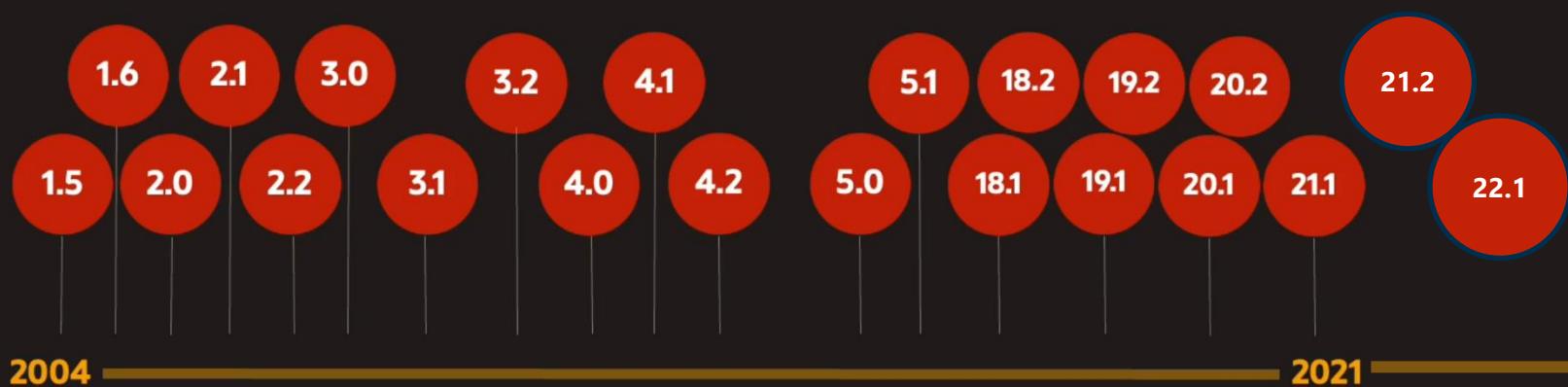
- Gives context to JavaScript functions
- Example one JavaScript module per APEX page

```
1  /**
2  * @namespace page1
3  */
4  const page1 = {};
5
6  /**
7  * @module demo
8  */
9  page1.demo = {
10    /**
11     * @function doSomething
12     * @example page1.demo.doSomething();
13     */
14     doSomething:function(){
15       //write your JS source here
16     }
17   };
18
```

page1.demo.doSomething();

APEX versions – new APIs

APEX Product Timeline



20
Releases

250+
Major features

0
required rewrites

APEX JS APIs

Shortcut to documentation

- apex.oracle.com/jsapi
- Namespaces
- Interfaces
- Widgets

Example

```
// Displays a page-level success message 'Changes saved!'.
apex.message.showPageSuccess( "Changes saved!" );
```

```
apex.debug.error( "Update Failed" );
```

```
apex.page.submit( "DELETE" );
```

```
> apex.items.P26_AREA_DESCRIPTION.value
< 'Ore Preperation'
```

```
apex.server.process( "MY_PROCESS", {
    x01: "test",
    pageItems: "#P1_DEPTNO,#P1_EMPNO"
}, {
    success: function( data ) {
        // do something here
    },
    error: function( jqXHR, textStatus, errorThrown ) {
        // handle error
    }
});
```

```
apex.env
```

```
▼ Object { APP_USER: "LSCHILDE@GMAIL.COM", APP_ID: "83349", API_APEX_FILES: "https://static.oracle.com/cdn/apex/21.2.0/" }
  APEX_FILES: "https://static.oracle.com/cdn/apex/21.2.0/"
  APEX_VERSION: "21.2.0"
  APP_FILES: "lschilde/r/83349/files/static/v1/"
  APP_ID: "83349"
  APP_PAGE_ID: "13"
  APP_SESSION: "1474258414065"
  APP_USER: "LSCHILDE@GMAIL.COM"
  WORKSPACE_FILES: "lschilde/r/files/static/v103/"
▶ <prototype>: Object { ... }
```

```
var dateString = apex.date.format( myDate, "YYYY-MM-DD HH24:MI" );
// output: "2021-06-29 15:30"
```

```
apex.pwa.openInstallDialog();
```

VALIDATIONS - Client side

The screenshot shows a user interface for adding a new customer. On the left, there is a vertical sidebar with icons for Home, Customers (31), Orders (10), Invoices (10), Reports, Settings, and Help. The main area has a header with a Cancel button and a blue Add Customer button. The form contains fields for First Name, Last Name, Street Address, Line 2, City, State, Zip Code, Credit Limit, Phone Number, Alternate Number, Email, URL, and Tags. Most fields have red asterisks indicating they are required. The Phone Number field contains the value "999-999-9999".

Cancel Add Customer

First Name * ≡

Last Name *

Street Address

Line 2

City

State * - Choose State - ▾

Zip Code *

Credit Limit *

Phone Number Alternate Number

Email

URL

Tags

VALIDATIONS - Client side

- Better user experience
- Data quality
- Only notified after submitting the page
- **Important to give the users immediate feedback**
- Validate items, forms, IGS
- We should do it also on server side

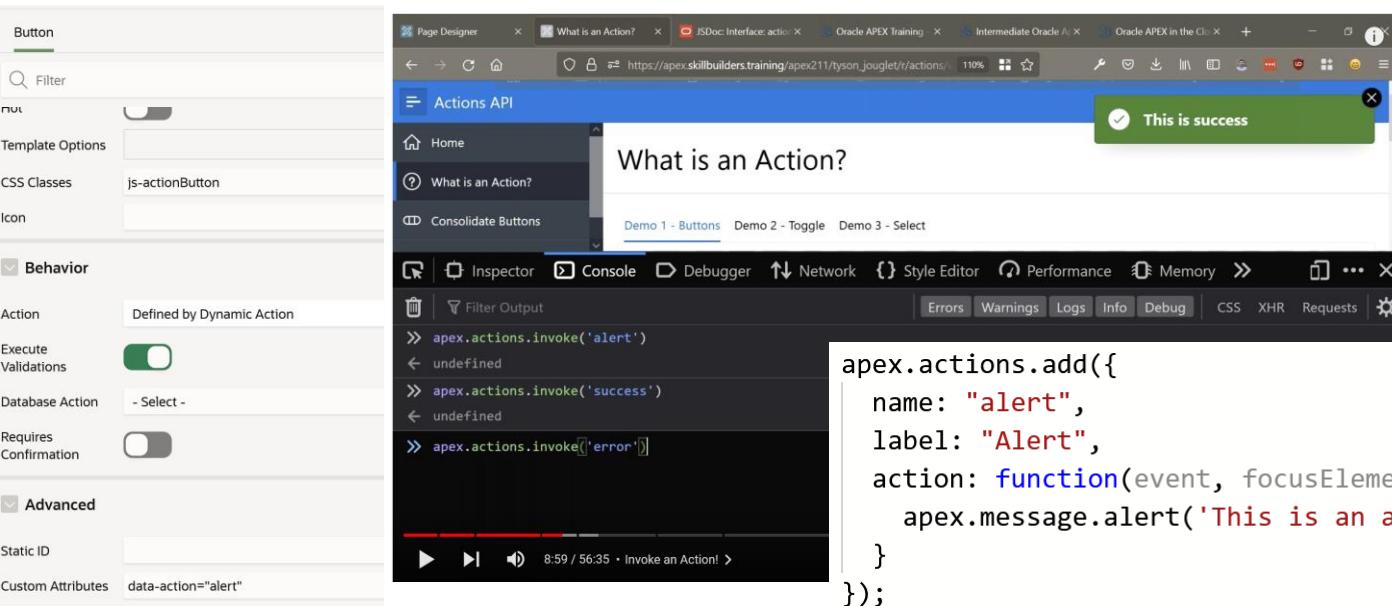
Prevent invalid user inputs
Email addresses
Number
Password complexity
Phone numbers

```
1  function isEmpty(pValue) {
2    var isEmpty = false;
3    if ($.trim(pValue) === "") {
4      isEmpty = true;
5    }
6    return isEmpty;
7  }
8
9  function isPositiveInteger(pValue) {
10   // an integer is a number that can be written without a fractional or decimal component
11   var isPositiveInteger = false;
12   var positiveIntegerRegex = /\^\d+$/;
13
14   if (pValue.match(positiveIntegerRegex)) {
15     isPositiveInteger = true;
16   }
17   return isPositiveInteger;
18 }
19
20 function isValidDate(pValue) {
21   var isValidDate = false;
22   // date format is DD/MM/YYYY
23   var dateFormatRegex = new RegExp("^(0[1-9]|1[0-2])/(0[1-9]|1[0-2])/(0[1-9]|1[0-2])$");
24
25   if (pValue.match(dateFormatRegex)) {
26     // seems that the date format is correct, but can we parse the date to a date object?
27     var dateArray = pValue.split("/");
28     var year = parseInt(dateArray[2]);
29     var month = parseInt(dateArray[1], 10);
30     var day = parseInt(dateArray[0], 10);
31     var date = new Date(year, month - 1, day);
32
33     if (((date.getMonth() + 1) === month) && (date.getDate() === day) && (date.getFullYear() === year)) {
34       isValidDate = true;
35     }
36   }
37   return isValidDate;
38 }
```

<https://www.youtube.com/watch?v=-fn4mHZC-AA>

ACTIONS interface

- Declarative way to improve your apps with built in APEX JS capability
- Reduce number of DAs
- Reduce number of buttons
- **Less replication of your code**
- Define action shortcuts



The screenshot shows the Oracle APEX Page Designer interface. On the left, a sidebar titled "Button" displays various configuration options: "Template Options" (set to "js-actionButton"), "Icon" (empty), "Behavior" (set to "Defined by Dynamic Action"), "Execute Validations" (switched on), "Database Action" (dropdown menu), "Requires Confirmation" (switched off), and "Advanced" (dropdown menu). At the bottom, "Custom Attributes" is set to "data-action='alert'". To the right, the main content area shows a browser window with the URL "https://apex.skillbuilders.training/apex211/tysen_joulet/r/actions". The page title is "What is an Action?". Below the title, there are three tabs: "Demo 1 - Buttons" (selected), "Demo 2 - Toggle", and "Demo 3 - Select". A green success message box is visible at the top right of the page. The browser's developer tools are open, specifically the "Console" tab, which contains the following JavaScript code:

```
apex.actions.add({
    name: "alert",
    label: "Alert",
    action: function(event, focusElement) {
        apex.message.alert('This is an action!');
    }
});
```

<https://skillbuilders.com/course/how-the-oracle-apex-actions-interface-make-apps-better-free-tutorial/>

PL/SQL and passing values

- React to an event
- Pass required data in
- Execute PL/SQL code
- Return the result
- Notify the user

```
1 declare
2   p_project_id number := APEX_APPLICATION.g_x01;
3   --p_action varchar2(50) := APEX_APPLICATION.g_x02;
4 begin
5   delete from eba_demo_da_emp where empno = p_project_id;
6   apex_json.open_object;
7   apex_json.write('success', true);
8   apex_json.close_object;
9 exception
10  when others then
11    apex_json.open_object;
12    apex_json.write('error', false);
13    apex_json.write('message', sqlerrm);
14    apex_json.close_object;
15 end;
```

3

```
1 <button type="button" title="My Button" aria-label="My Button" class="t-Button t-Button--noLabel"
2   onclick="apex.event.trigger(document,'deleteEmp',[{action:'rejecting', id:'#ID#'}]);">
3     <span aria-hidden="true" class="t-Icon fa fa-trash" ></span>
4   </button>
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
1 var sentID = this.data.id;
2 var action = this.data.action;
3 //var lSpinner$ = apex.util.showSpinner();
4 apex.server.process("DELETE_EMP",
5   { x01: pID,
6     x02: pAction,
7   },
8   { loadingIndicatorPosition : "page",
9     success: function(pData) {
10       // If the AJAX is successful set the value or the returned items
11       if (pData.success === true){
12         apex.region( "employees" ).refresh();
13         apex.message.showPageSuccess( "Sucessfully deleted ID = " + pID + "!" );
14       }
15     },
16     error: function(request, status, error) {
17       apex.message.clearErrors();
18       // Now show new errors
19       apex.message.showErrors([
20     ])
```

1

2

DEBUGGING?

- Browser Tools
 - F12
 - Right click inspect
 - Ctrl + Shift + I
- Get familiar with features
- Console log & Network tab
 - Make sure you enable APEX debug
 - Check the name of the DA
 - Check parameters sent in
 - Check the payload returned
 - apex.debug vs apex_debug

The screenshot shows a portion of an Oracle Apex application with three input fields: 'Employee Number', 'Name', and 'Job' (set to 'CLERK'). Below the application is a browser's developer tools interface, specifically the 'Console' tab of the Network panel.

Apex Application Screenshot:

- Employee Number
- Name
- Job: CLERK

Developer Tools Console Tab:

- Toolbar: DEMO APEX.ORACLE.CO..., Home, App 25616, Page 55, Session, Debug, Quick Edit, Customize, Help, Settings.
- Bottom tabs: Inspector, Console (selected), Debugger, Network, Style Editor, Performance, Memory, Storage, etc.
- Output pane:

```
Dynamic Action Fired: null (NATIVE_SET_VALUE)
▶ Object { triggeringElement: input#P55_BONUS.text_field.apex-item-text, affectedElements: [...], action: [...],
  browserEvent: [...], data: undefined, resumeCallback: s(t) ↵ }
```

- Log level buttons: Errors, Warnings, Logs, Info, Debug, CSS, XHR, Requests.
- Bottom status bar: desktop_all.min.js:5:1498, f:326:552.

That is not all!!!!

- **JavaScript stored procedures**
- Integration of GraalVM
- Execute SQL from within JavaScript
- Leverage a huge set of JavaScript libraries
- Processes, Computations, Validations, and the Execute Server-side code DA



Screenshot of the Oracle APEX developer interface showing a server-side code editor.

Identification:

- Name: [REDACTED]
- Action: Execute Server-side Code

Language: JavaScript (MLE)

Settings:

- Language: JavaScript (MLE)

JavaScript Code:

```
var rows;
function needsSalaryRaise( empno ) {
    // placeholder for some complicated logic to calculate if employee needs a raise
    return true;
}
rows = apex.conn.execute( "select empno from emp where deptno = :deptno", { deptno: apex.env.P2_DEPTNO } ).rows;
for ( var row of rows ) {
    if ( needsSalaryRaise( row.EMPNO ) ) {
        apex.conn.execute( "update emp set sal = sal * 1.03 where empno = :empno", { empno: row.EMPNO } );
    }
}
```

JavaScript - MLE

Language JavaScript (MLE) Environment Default Rows 10 Clear Command Find Tables

Q A::

```
1 function extendProjectTasks( name ) {
2     if (name !== '') {
3         return true;
4     }
5     else {
6         return false;
7     }
8 }
9
10 for ( var row of apex.conn.execute( "select id, name from eba_project_tasks where project_id = :project", { project: "5" } ).rows ) {
11     if ( extendProjectTasks( row.name ) ) {
12         apex.conn.execute( "update eba_project_tasks set end_date = end_date + 1 where id = :id", { id: row.ID } );
13         console.log("The task with the ID: " + row.ID + " and the name " + row.NAME + " has been extended successfully!");
14     }
15 }
16
17 }
```

Results Explain Describe Saved SQL History

The task with the ID: 34 and the name Design production screens for main Partner Portal app has been extended successfully!
The task with the ID: 35 and the name Train Partners has been extended successfully!
The task with the ID: 36 and the name Monitor Partners has been extended successfully!
The task with the ID: 37 and the name Review Beta Feedback from Partners has been extended successfully!
The task with the ID: 38 and the name Improve existing feature functions based on feedback has been extended successfully!
Statement processed.

0.14 seconds

<https://blogs.oracle.com/apex/post/mle-and-the-future-of-server-side-programming-in-oracle-apex>

PWA – push notifications

Notifications

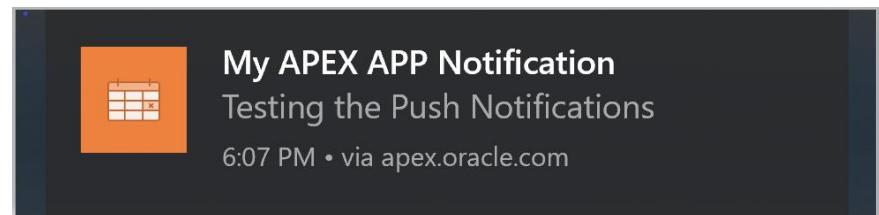
- Receive push notifications directly on your device, featuring a message title, body, icon, and link.

Easy Subscription

- APEX users can subscribe to push notifications from within their applications, while instance administrators can manage a queue of notifications.

Simple NO config using new APEX APIs

- apex_pwa.send_push_notification



PWA – push notifications

Low code push notifications, made available with a single switch.
Easily notify desktop and mobile users who've opted-in to receive
push notifications.

The image shows a collage of several push notification examples:

- A dark grey rounded rectangle containing a red square icon with a white 'O' and the word "FORUMS". Below it is the text "You've been mentioned in a post! Tim K. mentioned you in a comment in".
- A light grey rounded rectangle containing a white square icon with purple "PWA" letters. Below it is the text "APEX Push Notifications Notify any devices and deliver real-time updates to APEX users with web push notifications."
- A dark grey rounded rectangle containing a blue square icon with a white checkmark. Below it is the text "Task Completed Steve has completed the 'Pick up the oranges from the store' via apex.oracle.com".
- A light grey rounded rectangle containing a grey square icon with a white 'G'. Below it is the text "GOOGLE CHROME Event Added apex.oracle.com Added 'Reston Team Meetup' to the now" followed by a blue square icon with two people.
- A dark grey rounded rectangle containing a brown square icon with a white popcorn bucket. Below it is the text "Movie tonight? • oracle.com • now We've got the perfect movie f..." followed by a white downward arrow.
- A small dark grey rounded rectangle containing a Firefox logo and the word "Firefox".

Workshop

codeshare.io/WdK9vd

Summary

JavaScript is here to stay

- Little we know is enough to get us through
- Familiarize your self with JS and available built in features

Consolidate

- if it can trigger on same event or global page
- using actions interface

Minimize custom code

- But don't be afraid to use it
- Structure it well

Think security

- Apply server side security too
- Don't trust user inputs

Q&A

Thank you for attending



Resources

- https://www.youtube.com/watch?v=_Izurjl5Vm0
- https://www.youtube.com/watch?v=02thxqv-m_c
- <https://www.youtube.com/watch?v=-l6E5LuNU3U>
- <https://www.youtube.com/watch?v=uK7vCqfXxNs>
- <https://skillbuilders.com/course/learning-javascript/>
- <https://www.youtube.com/watch?v=phydnTdH81Y>
- <https://medium.com/graalvm/javascript-as-a-server-side-language-in-oracle-apex-20-2-457e073ca4ca>
- <https://technology.amis.nl/database/executing-javascript-from-pl-sql-in-oracle-database/>