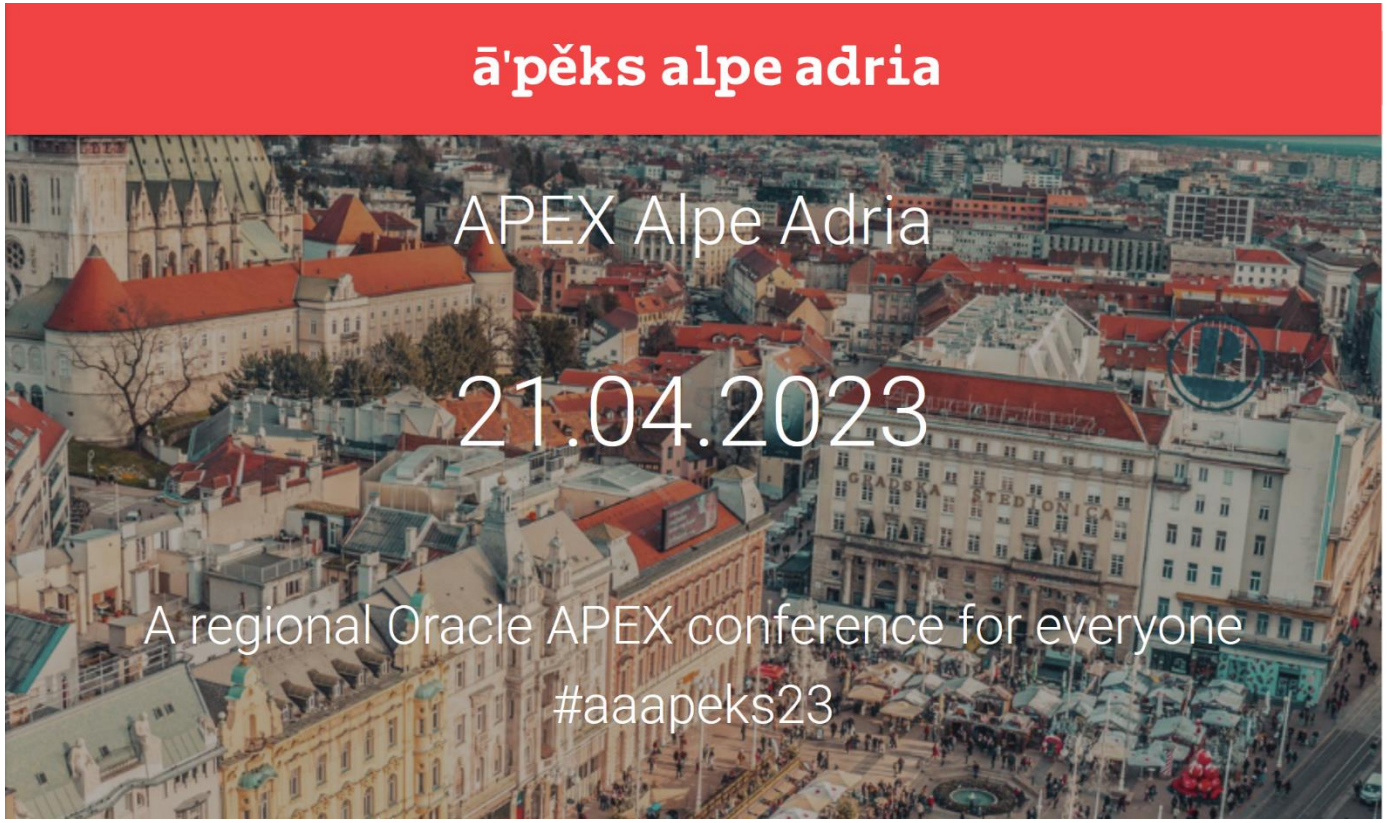
JavaScript for PL/SQL Developers



Lino Schildenfeld
@LinoSchilde
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

apex.oracle.com/officehours

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

SKILLBUILDERS

AUSOUG

fabē reduce • rescue • reconnect

FOEX

AOP

APEX R&D research & development

- AUSOUG APEX webinars
- NZ APEX meetup organizer
- APEX World Member of the Month
- Conference speaker
- Blogger

@LinoSchilde

Oracle ACE Pro

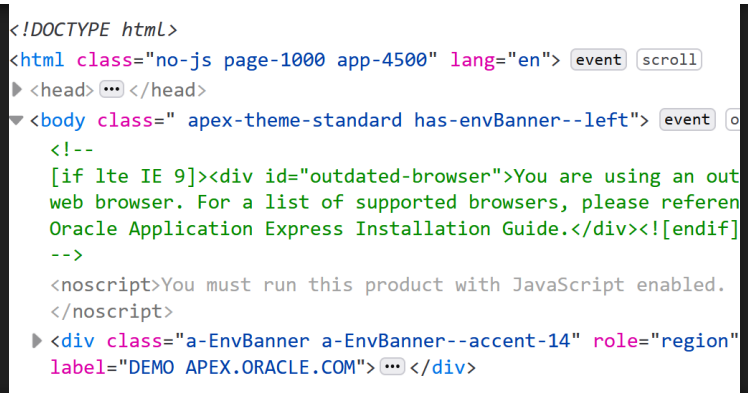
www.linkedin.com/in/lschilde

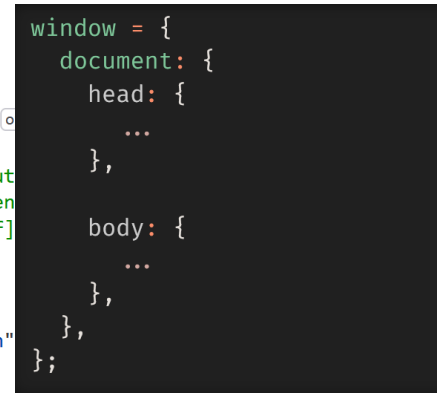
Motivation - Modern apps

- 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





- 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

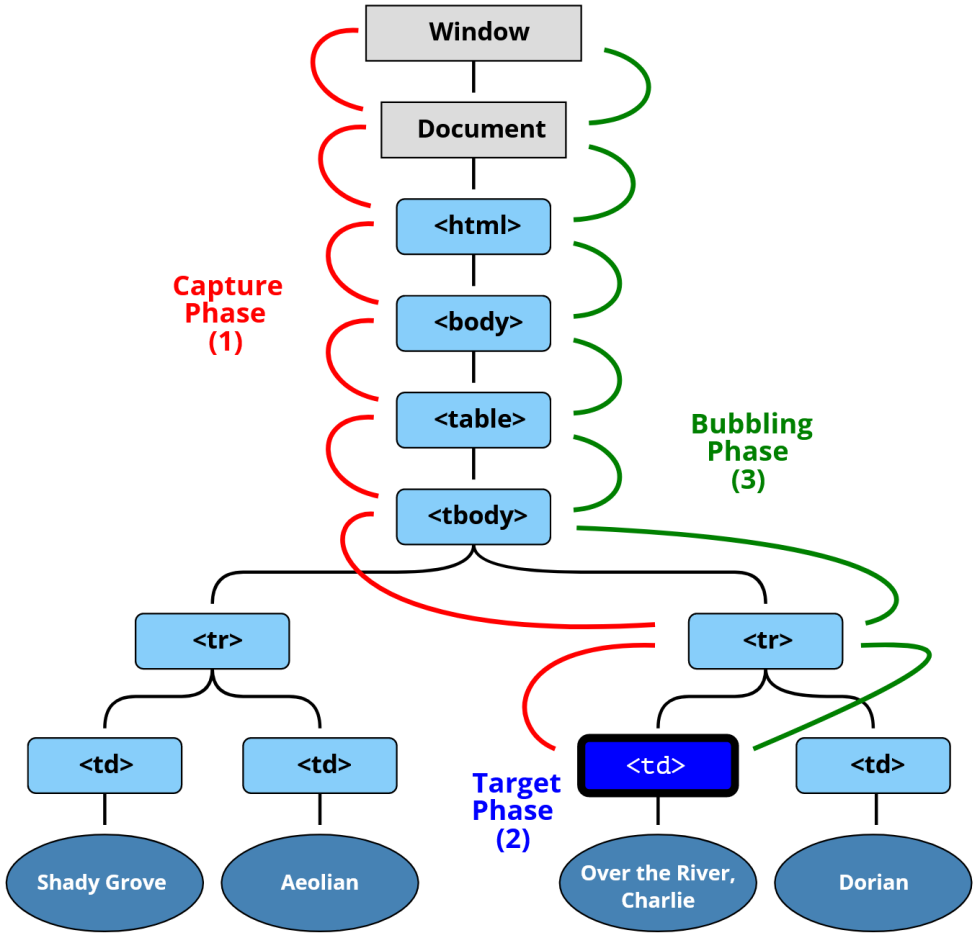
- Add/remove classes
- Modify attributes
- Element manipulation
- Remove or add elements
- Change css styles

```
<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



https://javascript.info/bubbling-and-capturing

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,prepand,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
```

…

…

```
<!doctype html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <title>attributeContains demo</title>
  <script src="https://code.jquery.com/jquery-3.5.0.js"></script>
</head>
<body>

<input name="man-news">
<input name="milkman">
<input name="letterman2">
<input name="newmilk">

<script>
$( "input[name*='man']" ).val( "has man in it!" );
</script>

</body>
</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

PLSQL scope example

```
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;
```

```
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,
bololean, 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);
```

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 = { };`

https://www.programiz.com/javascript/data-types

JS functions

- Functions are like any other data type
- Can be assign 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!!!
```

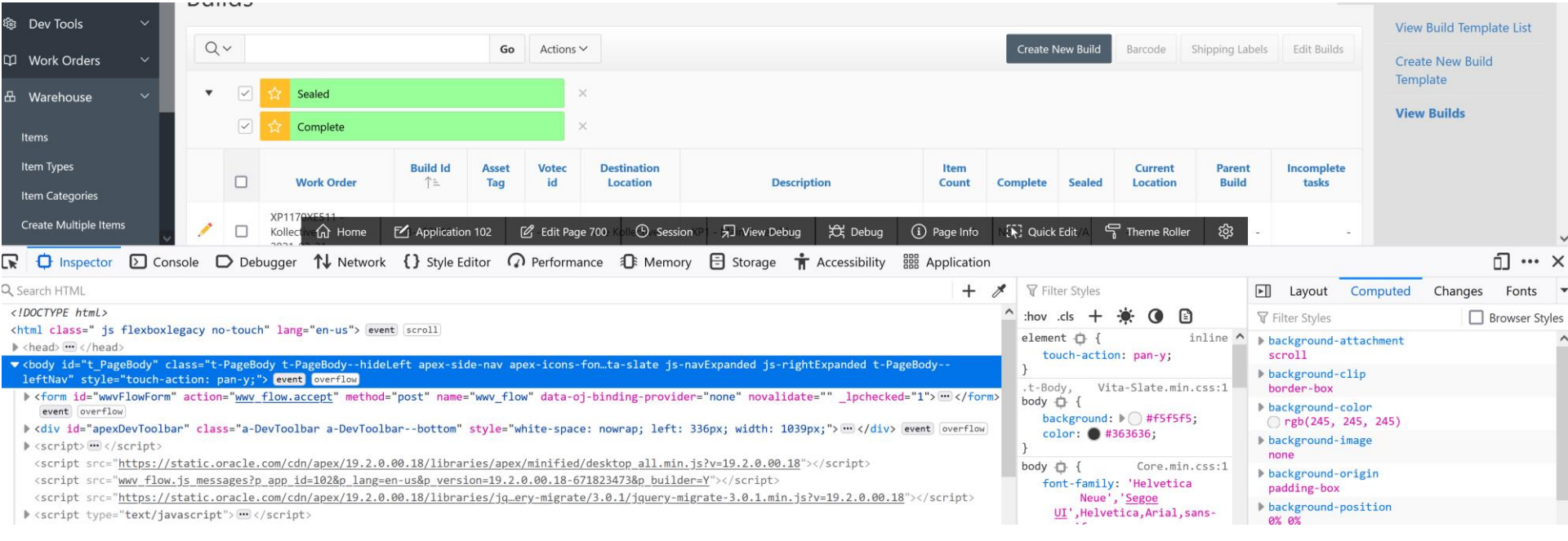
AJAX

- 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);
```

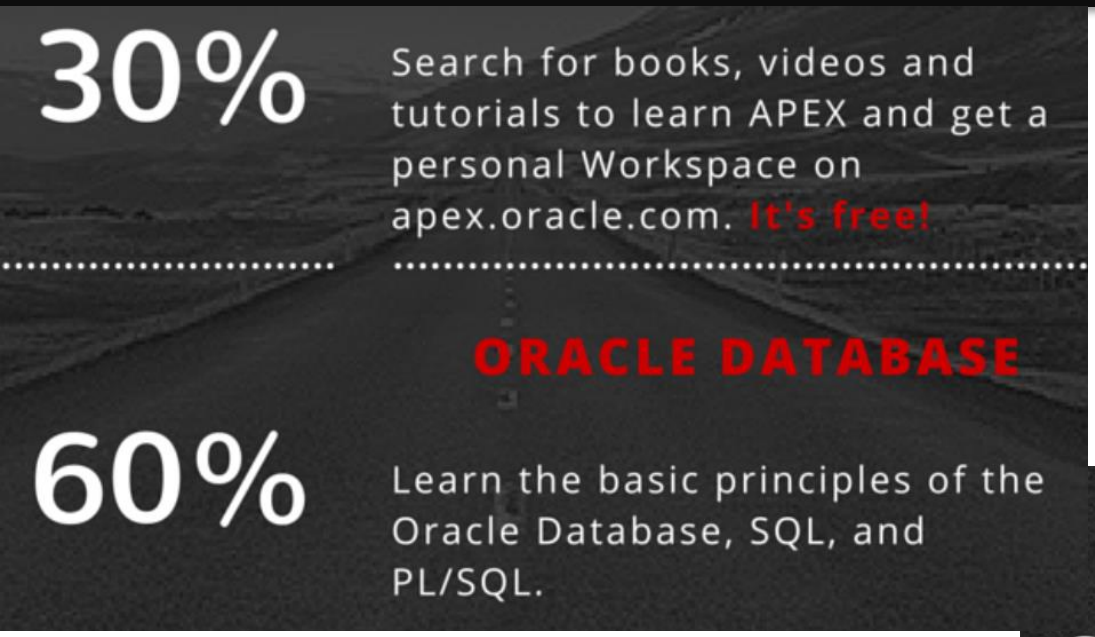
DEVELOPER TOOLS

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



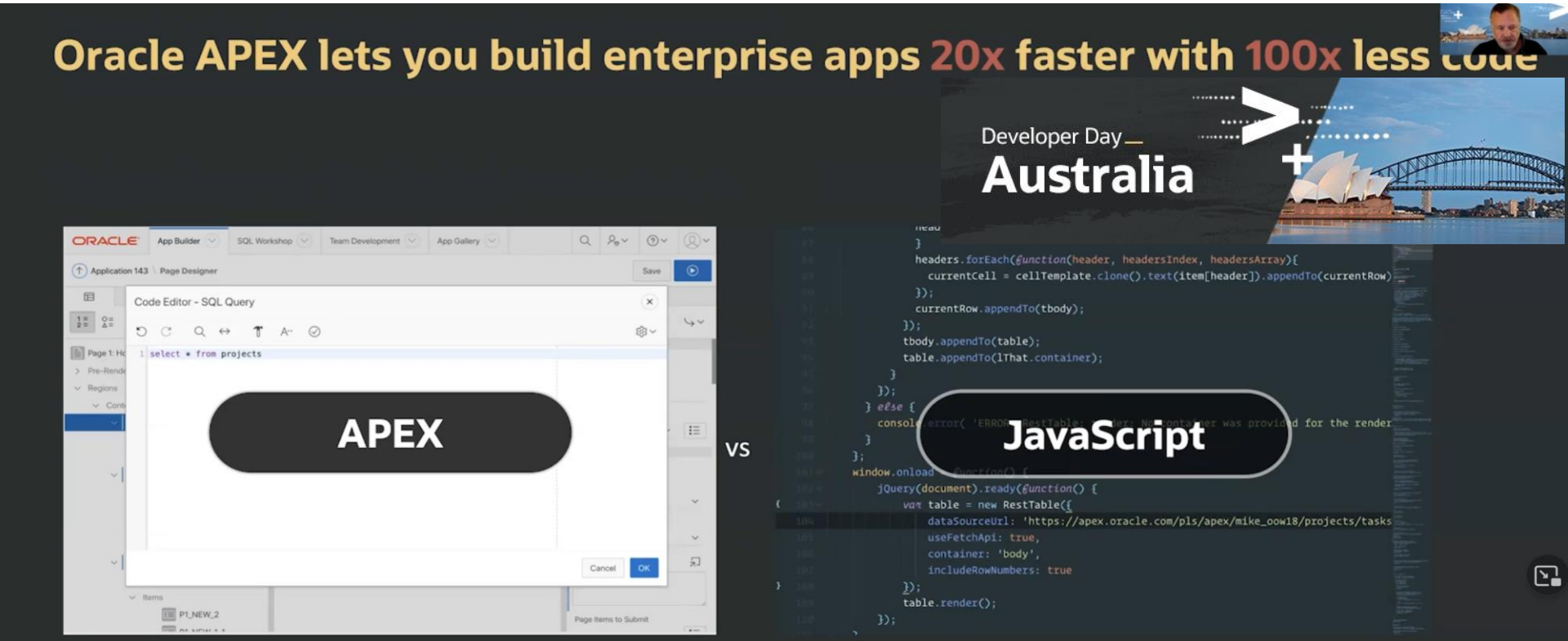
Oracle Developers

Perfect candidates



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



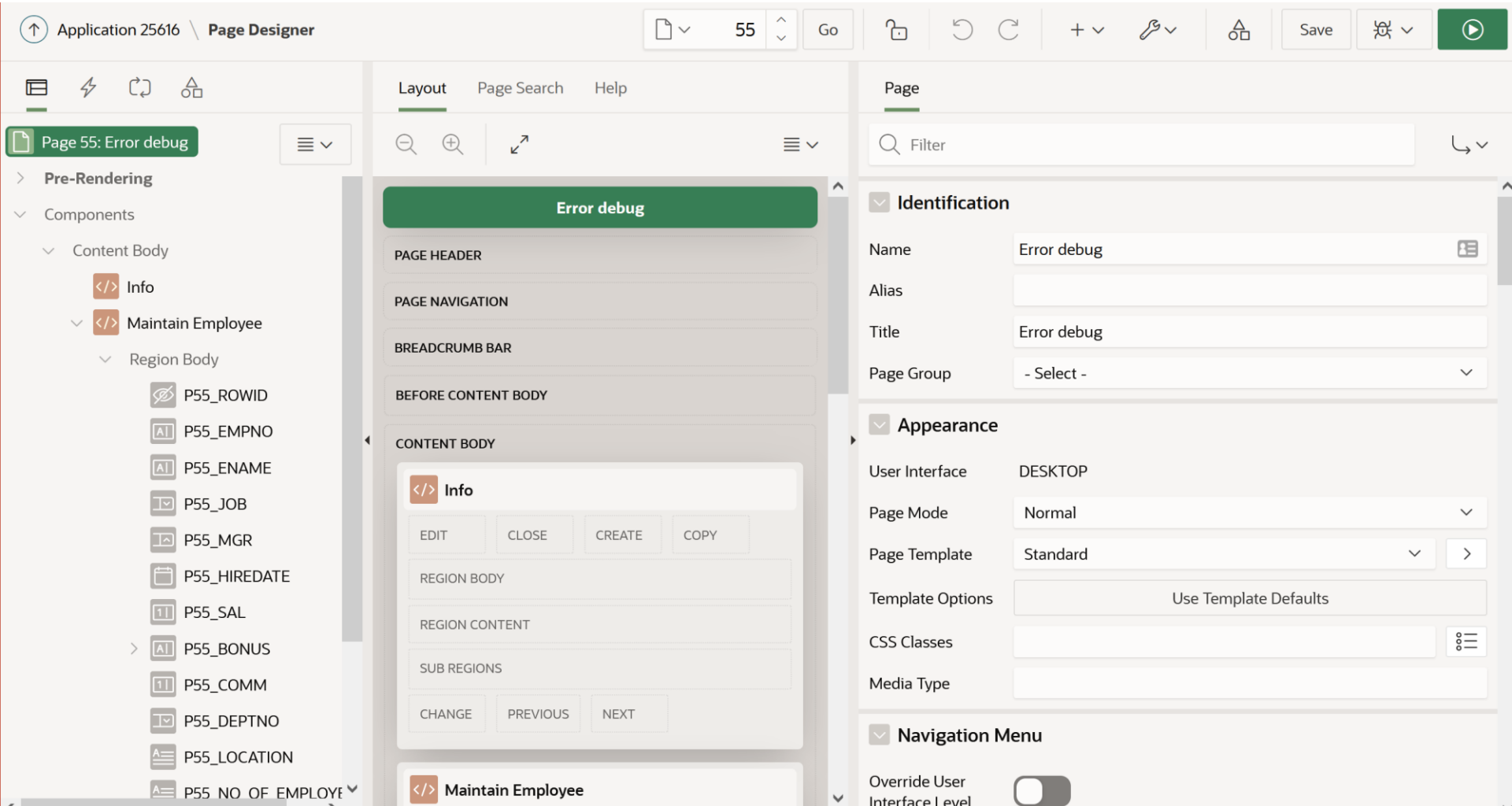
APEX and JavaScript

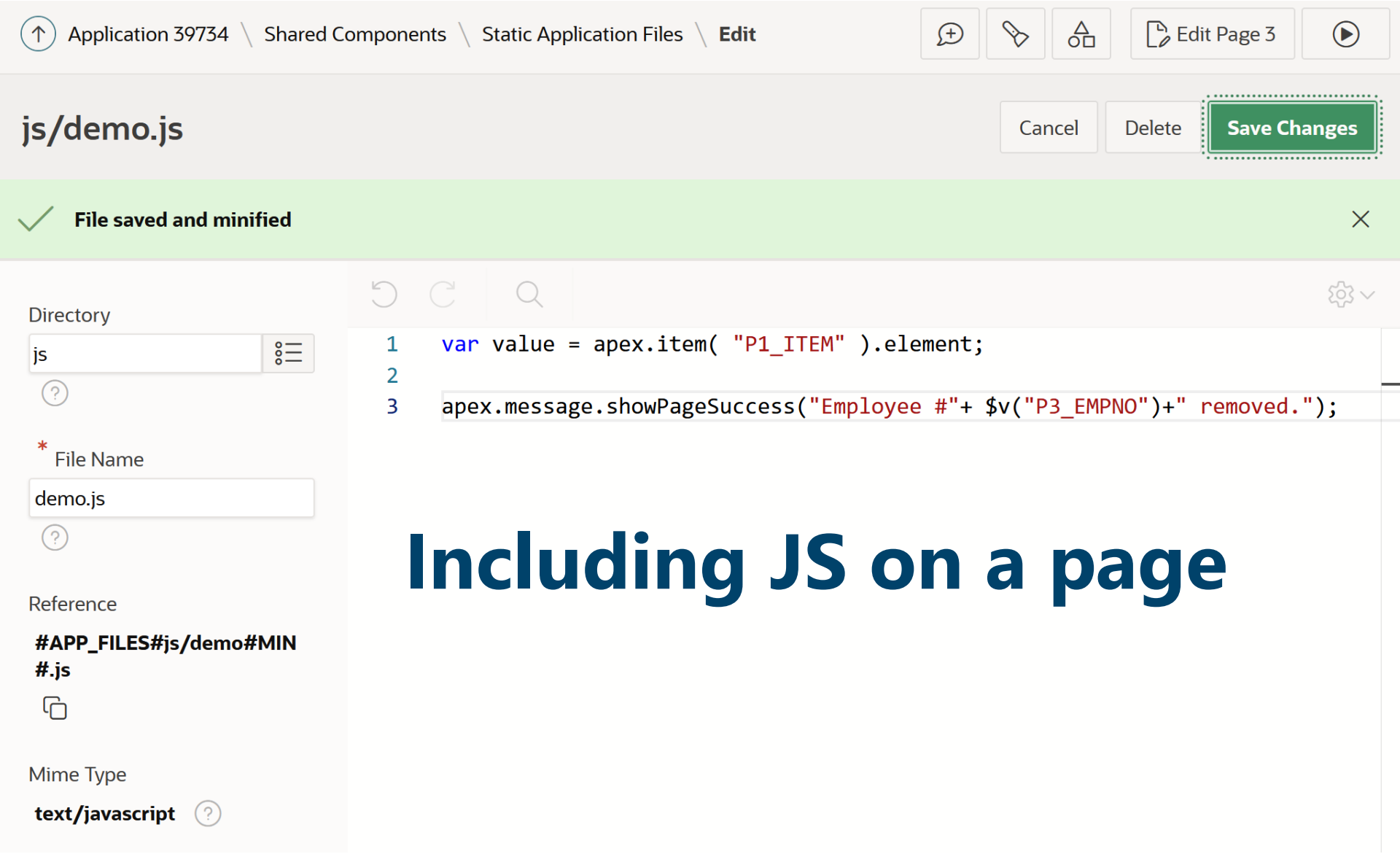
- Inline on a page
- Static application file or as application theme referenced files
- Anywhere where we can do HTML in APEX

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

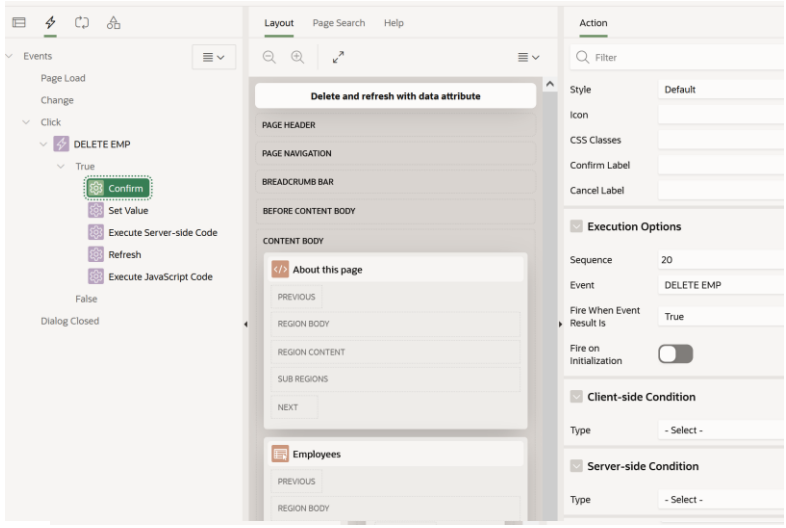




>21.2 APEX version

DYNAMIC ACTIONS

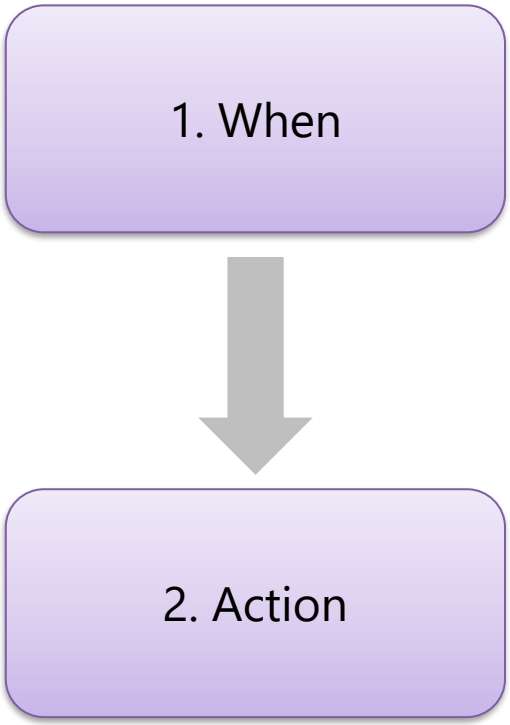
- Easiest way to include JS in APEX
- Declarative => little coding with little to no knowledge of JS
- **Most likely will not cover everything JavaScript** can do
- When complexity grows put it into files
- Code Sharing between pages
- We can now create and edit static JS files from APEX Builder



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

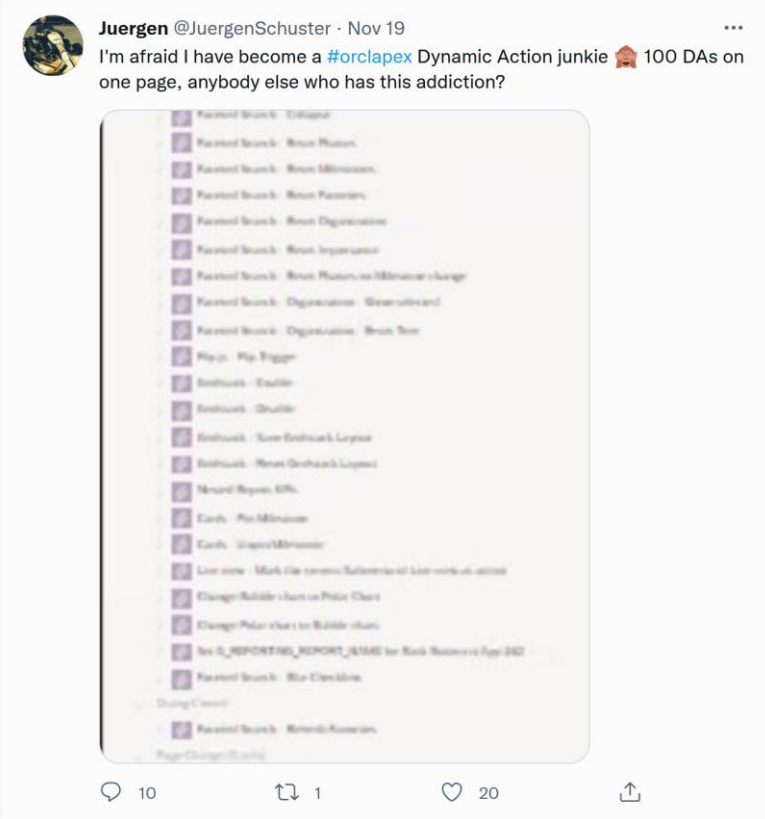
2. What you want to do when an event takes place

- True Actions
- False Actions

3*. Affected elements or Scope

DYNAMIC ACTIONS

@JuergenSchuster



@Matt_FOEX

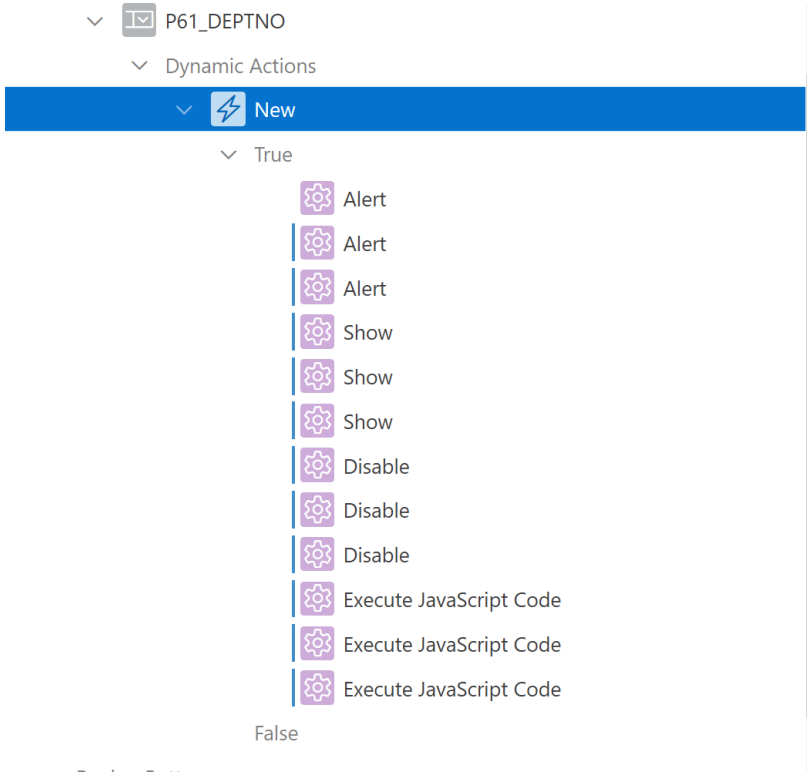
```
select da_count_tab.*
  from (select aap.page_id
              ,(select count(*)
                  from apex_application_page_da aapd
                 where aapd.application_id = 200
                   and aapd.page_id = aap.page_id) as count_da
              ,(select count(*)
                  from apex_application_page_da_acts aapda
                 where aapda.application_id = 200
                   and aapda.page_id = aap.page_id) as count_da_actions
          from apex_application_pages aap
         where aap.application_id = 200) da_count_tab
 where da_count_tab.count_da > 100
 order by da_count_tab.count_da desc
```

	PAGE_ID	COUNT_DA	COUNT_DA_ACTIONS
1	350	301	819
2	379	275	995
3	457	270	996
4	369	262	928
5	478	255	973
6	468	255	937
7	477	253	952
8	490	252	954
9	469	251	961
10	493	246	946
11	461	246	959
12	494	246	950
13	474	245	979
14	501	245	483
15	473	238	943
16	486	237	934
17	497	237	922
18	487	236	883
19	489	236	931
20	472	233	935
21	491	232	926
22	492	231	929
23	1500	230	517
24	488	228	868
25	495	228	902
26	496	226	904
27	302	225	488

DYNAMIC ACTIONS

- Consolidate
- Do not be afraid to use manual code

```
/**
 * @function show
 **/
show: function () {
    if (fabe.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
 * fabe.navigation.back.hide();
 **/
hide: function () {
    console.log(fabe.navigation.back.stack.length === 0,
        apex.item("P0_BACK_URL").isEmpty());

    if (fabe.navigation.back.stack.length === 0 &&
        apex.item("P0_BACK_URL").isEmpty()) {
        $(".fabe-back-all").hide();
        $(".brand-logo img").show();
    }

    if (fabe.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");
    }
},
```



Modularize JS code

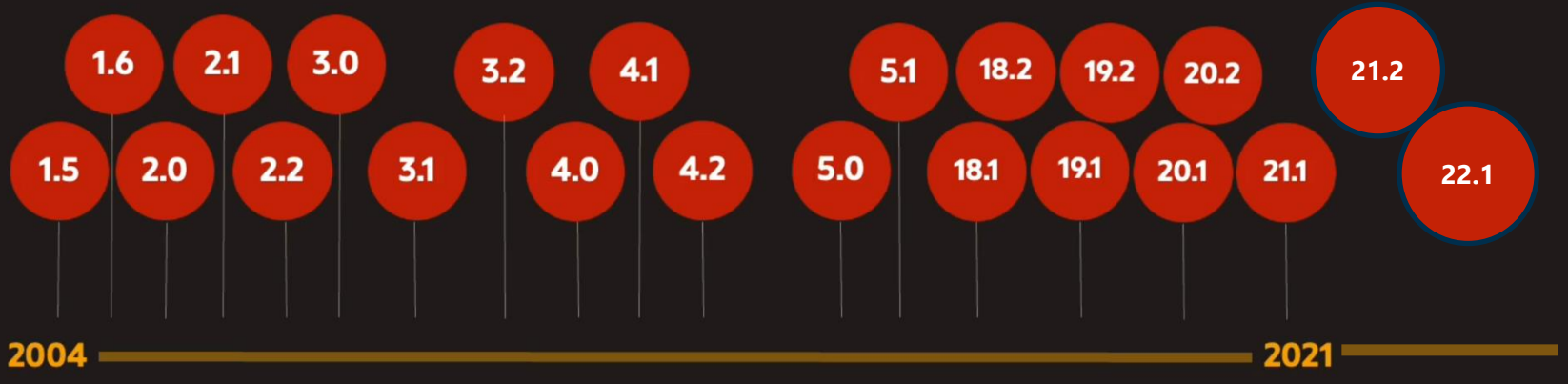
- Gives context to JavaScript functions
- Example one JavaScript module per APEX page

```
/**
* @namespace page1
**/
const page1 = {};

/**
* @module demo
**/
page1.demo = {
  /**
  * @function doSomething
  * @example page1.demo.doSomething();
  **/
   doSomething:function(){
    //write your JS source here
  }
};

```

```
page1.demo.doSomething();
```

APEX versions – new APIs

APEX Product Timeline



20	250+	0
Releases	Major features	required rewrites

APEX JS APIs

Shortcut to documentation

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

```
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
    }
} );
```

Example

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

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

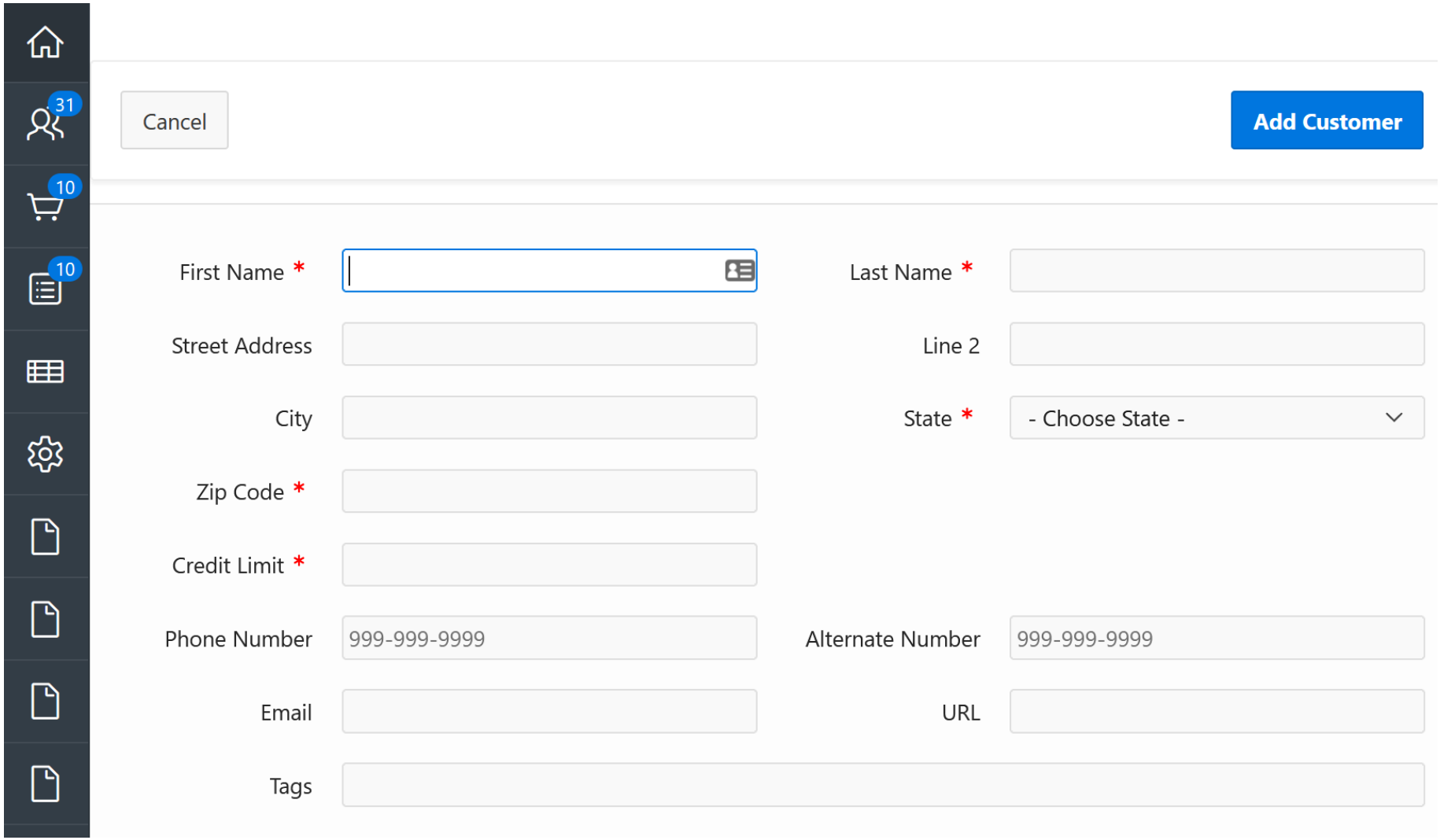
```
apex.page.submit( "DELETE" );
```

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

```
apex.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



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

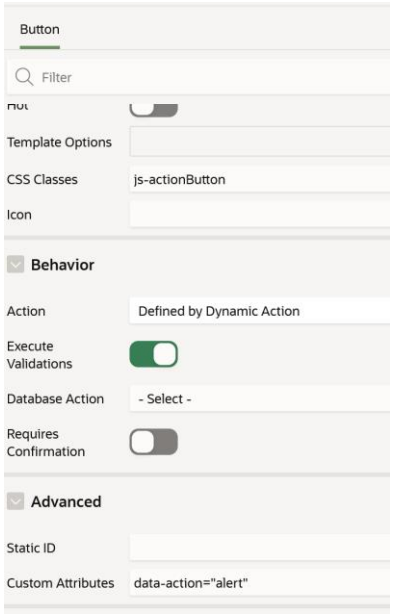
Prevent invalid user inputs
Email addresses
Number
Password complexity
Phone numbers

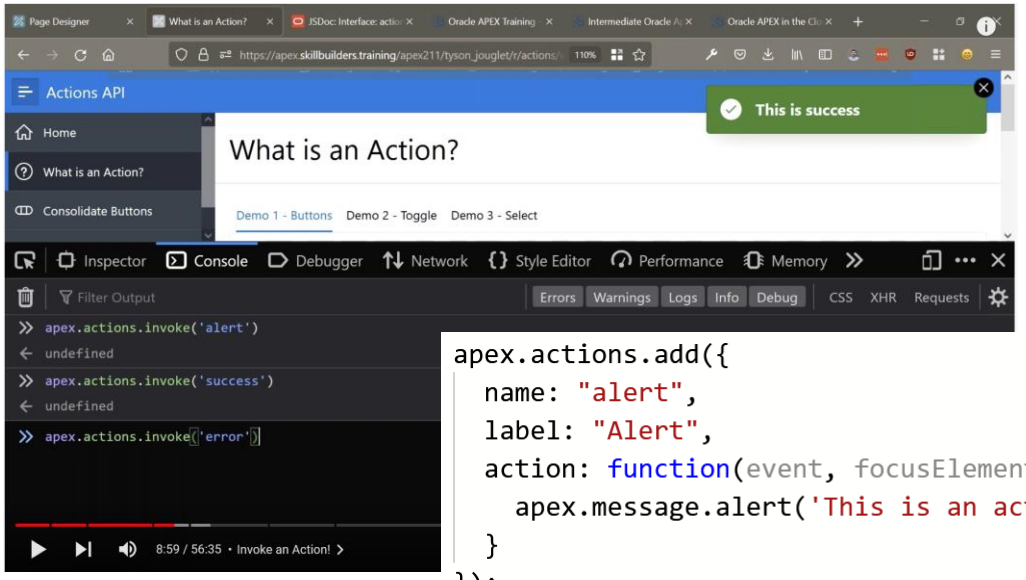
```
function isEmpty(pValue) {
  var isEmpty = false;
  if ($.trim(pValue) === "") {
    isEmpty = true;
  }
  return isEmpty;
}

function isPositiveInteger(pValue) {
  // an integer is a number that can be written without a fractional or decimal component
  var isPositiveInteger = false;
  var positiveIntegerRegex = /^\d+$/;

  if (pValue.match(positiveIntegerRegex)) {
    isPositiveInteger = true;
  }
  return isPositiveInteger;
}

function isValidDate(pValue) {
  var isValidDate = false;
  // date format is DD/MM/YYYY
  var dateFormatRegex = new RegExp("^(3[01]|[12][0-9]|0?[1-9])/(1[0-2]|0?[1-9])/(?:[0-9]{2})?[0-9]{

  if (pValue.match(dateFormatRegex)) {
    // seems that the date format is correct, but can we parse the date to a date object?
    var dateArray = pValue.split("/");
    var year = parseInt(dateArray[2]);
    var month = parseInt(dateArray[1], 10);
    var day = parseInt(dateArray[0], 10);
    var date = new Date(year, month - 1, day);

    if (((date.getMonth() + 1) === month) && (date.getDate() === day) && (date.getFullYear() === ye
      isValidDate = true;
    }
  }
  return isValidDate;
}
```

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






```
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

Establish communication between client and server side

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

1

```
<button type="button" title="My Button" aria-label="My Button" class="t-Button t-Button--noLabel
onclick="apex.event.trigger(document,'deleteEmp',[{action:'rejecting', id:'#ID#'}]);">
    <span aria-hidden="true" class="t-Icon fa fa-trash" ></span>
</button>
```

2

```
var sentID = this.data.id;
var action = this.data.action;
  //var lSpinner$ = apex.util.showSpinner();
apex.server.process("DELETE_EMP",
                    { x01: pID,
                      x02: pAction,
                    },
                    { loadingIndicatorPosition : "page",
                     success: function(pData) {
                        // If the AJAX is successful set the value or the returned items
                        if (pData.success === true){
                         apex.region( "employees" ).refresh();
                         apex.message.showPageSuccess( "Sucessfully deleted ID = " + pID + "!" );
                        }
                     },
                    error: function(request, status, error) {
                        apex.message.clearErrors();
                        // Now show new errors
                        apex.message.showErrors([
                            {
```

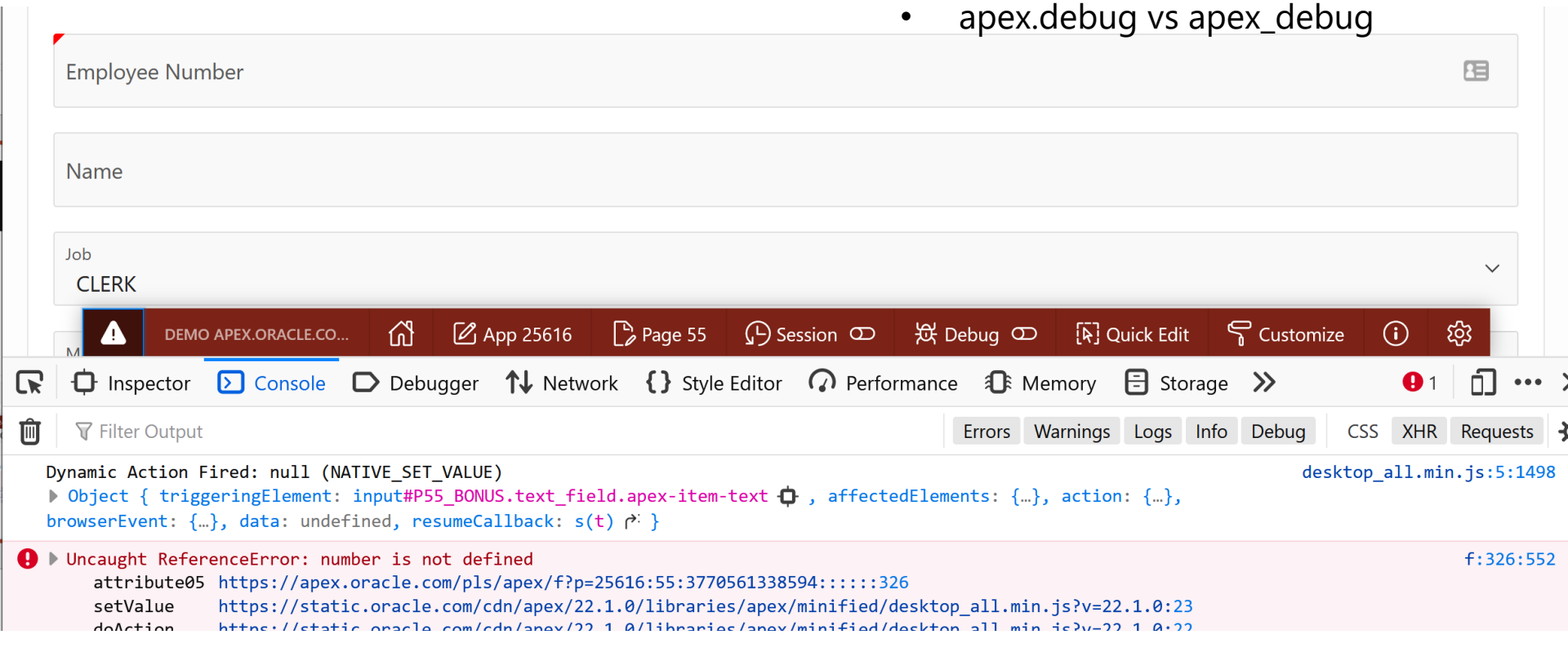
3

```
declare
 p_project_id number := APEX_APPLICATION.g_x01;
 --p_action varchar2(50) := APEX_APPLICATION.g_x02;
begin
   delete from eba_demo_da_emp where empno = p_project_id;
   apex_json.open_object;
   apex_json.write('success', true);
   apex_json.close_object;
exception
   when others then
      apex_json.open_object;
      apex_json.write('error', false);
      apex_json.write('message', sqlerrm);
      apex_json.close_object;
end;
```

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



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

Identification

Name

Action: Execute Server-side Code

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 } );
  }
}
```

Language: JavaScript (MLE) | Environment: Default | Rows: 10

```
1  apex.conn.execute("begin dbms_output.put_line('Hello from plsql'); end;" );
```

JavaScript - MLE

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

```
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

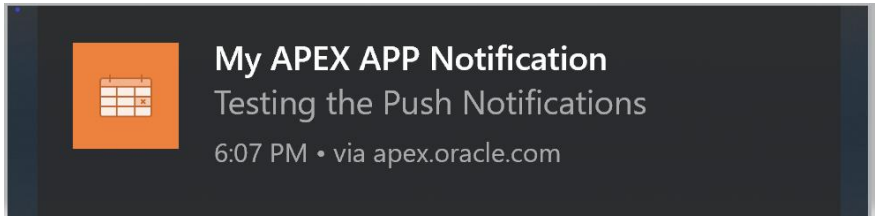
- Receive push notifications directly on your device, featuring a message title, body, icon, and link.

Easy Subscription

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

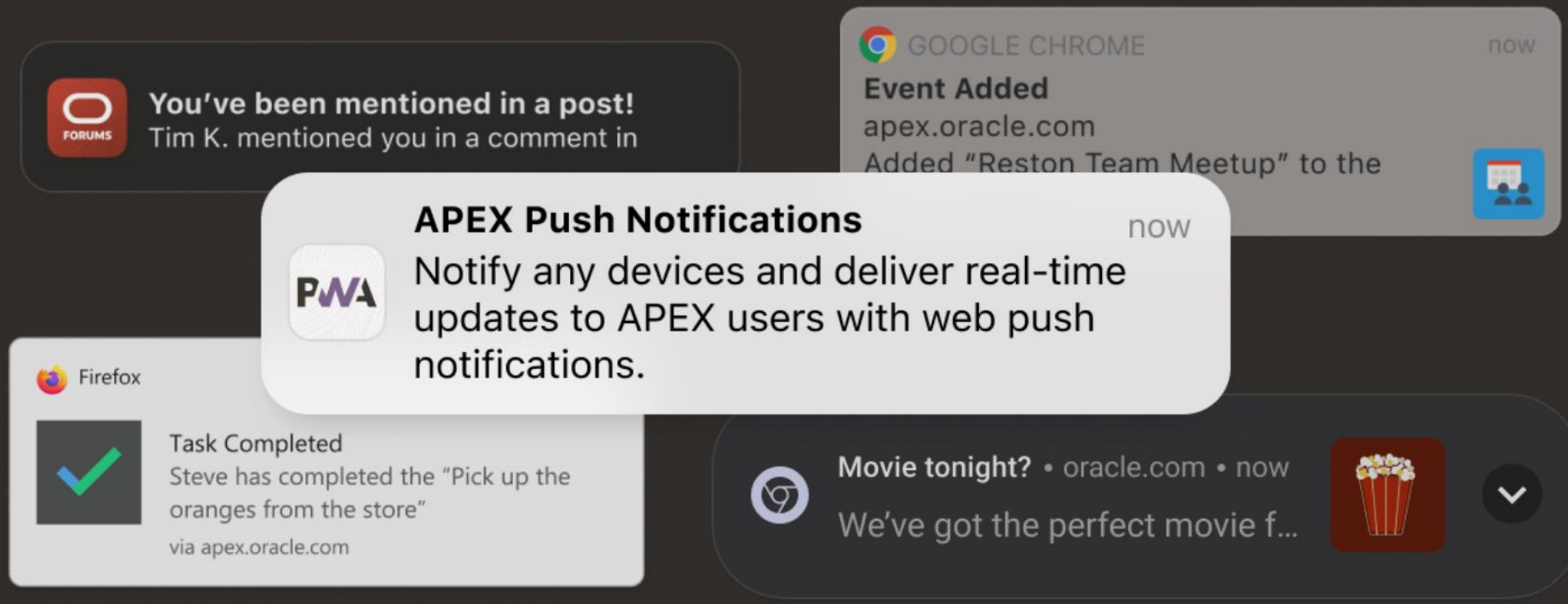
Simple NO config using new APEX APIs

- 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.



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=_lzurjl5Vm0
- 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/