/\*

\* Copyright (c) 2015, 2019, Oracle and/or its affiliates. All rights reserved.

\* ORACLE PROPRIETARY/CONFIDENTIAL. Use is subject to license terms.

\*

\*

\*

\*

\*

\*

\*

\*

\*

\*

\*

\*

\*

\*

\*

\*

\*

\*

\*

\*

\*/

var noResult = {l: "No results found"};

var catModules = "Modules";

var catPackages = "Packages";

var catTypes = "Types";

var catMembers = "Members";

var catSearchTags = "SearchTags";

var highlight = "<span class=\"resultHighlight\">$&</span>";

var searchPattern = "";

var RANKING\_THRESHOLD = 2;

var NO\_MATCH = 0xffff;

var MAX\_RESULTS\_PER\_CATEGORY = 500;

function escapeHtml(str) {

return str.replace(/</g, "&lt;").replace(/>/g, "&gt;");

}

function getHighlightedText(item, matcher) {

var escapedItem = escapeHtml(item);

return escapedItem.replace(matcher, highlight);

}

function getURLPrefix(ui) {

var urlPrefix="";

var slash = "/";

if (ui.item.category === catModules) {

return ui.item.l + slash;

} else if (ui.item.category === catPackages && ui.item.m) {

return ui.item.m + slash;

} else if ((ui.item.category === catTypes && ui.item.p) || ui.item.category === catMembers) {

$.each(packageSearchIndex, function(index, item) {

if (item.m && ui.item.p == item.l) {

urlPrefix = item.m + slash;

}

});

return urlPrefix;

} else {

return urlPrefix;

}

return urlPrefix;

}

function makeCamelCaseRegex(term) {

var pattern = "";

var isWordToken = false;

term.replace(/,\s\*/g, ", ").trim().split(/\s+/).forEach(function(w, index) {

if (index > 0) {

// whitespace between identifiers is significant

pattern += (isWordToken && /^\w/.test(w)) ? "\\s+" : "\\s\*";

}

var tokens = w.split(/(?=[A-Z,.()<>[\/])/);

for (var i = 0; i < tokens.length; i++) {

var s = tokens[i];

if (s === "") {

continue;

}

pattern += $.ui.autocomplete.escapeRegex(s);

isWordToken = /\w$/.test(s);

if (isWordToken) {

pattern += "([a-z0-9\_$<>\\[\\]]\*?)";

}

}

});

return pattern;

}

function createMatcher(pattern, flags) {

var isCamelCase = /[A-Z]/.test(pattern);

return new RegExp(pattern, flags + (isCamelCase ? "" : "i"));

}

var watermark = 'Search';

$(function() {

$("#search").val('');

$("#search").prop("disabled", false);

$("#reset").prop("disabled", false);

$("#search").val(watermark).addClass('watermark');

$("#search").blur(function() {

if ($(this).val().length == 0) {

$(this).val(watermark).addClass('watermark');

}

});

$("#search").on('click keydown paste', function() {

if ($(this).val() == watermark) {

$(this).val('').removeClass('watermark');

}

});

$("#reset").click(function() {

$("#search").val('');

$("#search").focus();

});

$("#search").focus();

$("#search")[0].setSelectionRange(0, 0);

});

$.widget("custom.catcomplete", $.ui.autocomplete, {

\_create: function() {

this.\_super();

this.widget().menu("option", "items", "> :not(.ui-autocomplete-category)");

},

\_renderMenu: function(ul, items) {

var rMenu = this;

var currentCategory = "";

rMenu.menu.bindings = $();

$.each(items, function(index, item) {

var li;

if (item.l !== noResult.l && item.category !== currentCategory) {

ul.append("<li class=\"ui-autocomplete-category\">" + item.category + "</li>");

currentCategory = item.category;

}

li = rMenu.\_renderItemData(ul, item);

if (item.category) {

li.attr("aria-label", item.category + " : " + item.l);

li.attr("class", "resultItem");

} else {

li.attr("aria-label", item.l);

li.attr("class", "resultItem");

}

});

},

\_renderItem: function(ul, item) {

var label = "";

var matcher = createMatcher(escapeHtml(searchPattern), "g");

if (item.category === catModules) {

label = getHighlightedText(item.l, matcher);

} else if (item.category === catPackages) {

label = (item.m)

? getHighlightedText(item.m + "/" + item.l, matcher)

: getHighlightedText(item.l, matcher);

} else if (item.category === catTypes) {

label = (item.p)

? getHighlightedText(item.p + "." + item.l, matcher)

: getHighlightedText(item.l, matcher);

} else if (item.category === catMembers) {

label = getHighlightedText(item.p + "." + (item.c + "." + item.l), matcher);

} else if (item.category === catSearchTags) {

label = getHighlightedText(item.l, matcher);

} else {

label = item.l;

}

var li = $("<li/>").appendTo(ul);

var div = $("<div/>").appendTo(li);

if (item.category === catSearchTags) {

if (item.d) {

div.html(label + "<span class=\"searchTagHolderResult\"> (" + item.h + ")</span><br><span class=\"searchTagDescResult\">"

+ item.d + "</span><br>");

} else {

div.html(label + "<span class=\"searchTagHolderResult\"> (" + item.h + ")</span>");

}

} else {

div.html(label);

}

return li;

}

});

function rankMatch(match, category) {

if (!match) {

return NO\_MATCH;

}

var index = match.index;

var input = match.input;

var leftBoundaryMatch = 2;

var periferalMatch = 0;

var delta = 0;

// make sure match is anchored on a left word boundary

if (index === 0 || /\W/.test(input[index - 1]) || "\_" === input[index - 1] || "\_" === input[index]) {

leftBoundaryMatch = 0;

} else if (input[index] === input[index].toUpperCase() && !/^[A-Z0-9\_$]+$/.test(input)) {

leftBoundaryMatch = 1;

}

var matchEnd = index + match[0].length;

var leftParen = input.indexOf("(");

// exclude peripheral matches

if (category !== catModules && category !== catSearchTags) {

var endOfName = leftParen > -1 ? leftParen : input.length;

var delim = category === catPackages ? "/" : ".";

if (leftParen > -1 && leftParen < index) {

periferalMatch += 2;

} else if (input.lastIndexOf(delim, endOfName) >= matchEnd) {

periferalMatch += 2;

}

}

for (var i = 1; i < match.length; i++) {

// lower ranking if parts of the name are missing

if (match[i])

delta += match[i].length;

}

if (category === catTypes) {

// lower ranking if a type name contains unmatched camel-case parts

if (/[A-Z]/.test(input.substring(matchEnd)))

delta += 5;

if (/[A-Z]/.test(input.substring(0, index)))

delta += 5;

}

return leftBoundaryMatch + periferalMatch + (delta / 200);

}

$(function() {

$("#search").catcomplete({

minLength: 1,

delay: 300,

source: function(request, response) {

var result = [];

var newResults = [];

searchPattern = makeCamelCaseRegex(request.term);

if (searchPattern === "") {

return this.close();

}

var camelCaseMatcher = createMatcher(searchPattern, "");

var boundaryMatcher = createMatcher("\\b" + searchPattern, "");

function concatResults(a1, a2) {

a2.sort(function(e1, e2) {

return e1.ranking - e2.ranking;

});

a1 = a1.concat(a2.map(function(e) { return e.item; }));

a2.length = 0;

return a1;

}

if (moduleSearchIndex) {

$.each(moduleSearchIndex, function(index, item) {

item.category = catModules;

var ranking = rankMatch(boundaryMatcher.exec(item.l), catModules);

if (ranking < RANKING\_THRESHOLD) {

newResults.push({ ranking: ranking, item: item });

}

return newResults.length < MAX\_RESULTS\_PER\_CATEGORY;

});

result = concatResults(result, newResults);

}

if (packageSearchIndex) {

$.each(packageSearchIndex, function(index, item) {

item.category = catPackages;

var name = (item.m && request.term.indexOf("/") > -1)

? (item.m + "/" + item.l)

: item.l;

var ranking = rankMatch(boundaryMatcher.exec(name), catPackages);

if (ranking < RANKING\_THRESHOLD) {

newResults.push({ ranking: ranking, item: item });

}

return newResults.length < MAX\_RESULTS\_PER\_CATEGORY;

});

result = concatResults(result, newResults);

}

if (typeSearchIndex) {

$.each(typeSearchIndex, function(index, item) {

item.category = catTypes;

var name = request.term.indexOf(".") > -1

? item.p + "." + item.l

: item.l;

var ranking = rankMatch(camelCaseMatcher.exec(name), catTypes);

if (ranking < RANKING\_THRESHOLD) {

newResults.push({ ranking: ranking, item: item });

}

return newResults.length < MAX\_RESULTS\_PER\_CATEGORY;

});

result = concatResults(result, newResults);

}

if (memberSearchIndex) {

$.each(memberSearchIndex, function(index, item) {

item.category = catMembers;

var name = request.term.indexOf(".") > -1

? item.p + "." + item.c + "." + item.l

: item.l;

var ranking = rankMatch(camelCaseMatcher.exec(name), catMembers);

if (ranking < RANKING\_THRESHOLD) {

newResults.push({ ranking: ranking, item: item });

}

return newResults.length < MAX\_RESULTS\_PER\_CATEGORY;

});

result = concatResults(result, newResults);

}

if (tagSearchIndex) {

$.each(tagSearchIndex, function(index, item) {

item.category = catSearchTags;

var ranking = rankMatch(boundaryMatcher.exec(item.l), catSearchTags);

if (ranking < RANKING\_THRESHOLD) {

newResults.push({ ranking: ranking, item: item });

}

return newResults.length < MAX\_RESULTS\_PER\_CATEGORY;

});

result = concatResults(result, newResults);

}

response(result);

},

response: function(event, ui) {

if (!ui.content.length) {

ui.content.push(noResult);

} else {

$("#search").empty();

}

},

autoFocus: true,

position: {

collision: "flip"

},

select: function(event, ui) {

if (ui.item.l !== noResult.l) {

var url = getURLPrefix(ui);

if (ui.item.category === catModules) {

url += "module-summary.html";

} else if (ui.item.category === catPackages) {

if (ui.item.url) {

url = ui.item.url;

} else {

url += ui.item.l.replace(/\./g, '/') + "/package-summary.html";

}

} else if (ui.item.category === catTypes) {

if (ui.item.url) {

url = ui.item.url;

} else if (ui.item.p === "<Unnamed>") {

url += ui.item.l + ".html";

} else {

url += ui.item.p.replace(/\./g, '/') + "/" + ui.item.l + ".html";

}

} else if (ui.item.category === catMembers) {

if (ui.item.p === "<Unnamed>") {

url += ui.item.c + ".html" + "#";

} else {

url += ui.item.p.replace(/\./g, '/') + "/" + ui.item.c + ".html" + "#";

}

if (ui.item.url) {

url += ui.item.url;

} else {

url += ui.item.l;

}

} else if (ui.item.category === catSearchTags) {

url += ui.item.u;

}

if (top !== window) {

parent.classFrame.location = pathtoroot + url;

} else {

window.location.href = pathtoroot + url;

}

$("#search").focus();

}

}

});

});