**UI Code Snippets**

**HTML & CSS**

**Custom checkbox CSS**

**<!DOCTYPE html>**

**<html>**

**<head>**

**<style>**

***/\* Custom check box css start \*/***

**.checkbox-wrapper {**

**display: inline-block;**

**position: relative;**

**padding-left: 30px;**

**margin-right: 10px;**

**margin-bottom: 12px;**

**cursor: pointer;**

**font-size: 16px;**

**-webkit-user-select: none;**

**-moz-user-select: none;**

**-ms-user-select: none;**

**user-select: none;**

**}**

**.checkbox-wrapper input {display: none}**

**.checkbox-wrapper input .checkmark {position: relative;}**

**.checkbox-wrapper input ~ .checkmark::before,**

**.checkbox-wrapper input:checked ~ .checkmark:after {content: "";position: absolute;}**

**.checkbox-wrapper input~.checkmark::before{left:0px;top:0px;width:22px;height:22px;background:#2196F3}**

**.checkbox-wrapper input:checked ~ .checkmark:after {**

**left: 8px;**

**top: 4px;**

**width: 5px;**

**height: 10px;**

**border: solid white;**

**border-width: 0 3px 3px 0;**

**-webkit-transform: rotate(45deg);**

**-moz-transform: rotate(45deg);**

**-ms-transform: rotate(45deg);**

**transform: rotate(45deg);**

**}**

***/\* Custom check box css End \*/***

**</style>**

**</head>**

**<body>**

**<h1>Custom Checkboxes</h1>**

**<label class="checkbox-wrapper">**

**<input type="checkbox" checked="checked">**

**<span class="checkmark">One One</span>**

**</label>**

**</body>**

**</html>**

**Custom radio CSS**

**<!DOCTYPE html>**

**<html>**

**<head>**

**<style>**

***/\* Custom check box css start \*/***

**.radio-wrapper {**

**display: inline-block;**

**position: relative;**

**padding-left: 30px;**

**margin-right: 10px;**

**margin-bottom: 12px;**

**cursor: pointer;**

**font-size: 16px;**

**-webkit-user-select: none;**

**-moz-user-select: none;**

**-ms-user-select: none;**

**user-select: none;**

**}**

***/\* Hide the browser's default checkbox \*/***

**.radio-wrapper input {display: none}**

**.radio-wrapper input .checkmark {position: relative;}**

**.radio-wrapper input ~ .checkmark::before,**

**.radio-wrapper input:checked ~ .checkmark:after {content: "";position: absolute;border-radius:50%;}**

**.radio-wrapper input~.checkmark::before{left:0px;top:0px;width:22px;height:22px;background:#2196F3;}**

**.radio-wrapper input:checked ~ .checkmark:after {**

**left: 6px;**

**top: 6px;**

**width: 10px;**

**height: 10px;**

**background-color: #fff;**

**}**

***/\* Custom check box css End \*/***

**</style>**

**</head>**

**<body>**

**<h1>Custom Checkboxes</h1>**

**<label class="radio-wrapper">**

**<input type="checkbox" checked="checked">**

**<span class="checkmark">One One</span>**

**</label>**

**</body>**

**</html>**

**Arrow CSS**

*/\* Arrow css\*/*

  .arrow{border:solid black;border-width:0 3px 3px 0;display:inline-block;padding:6px;background-color: transparent;}

  .arrow-left {-webkit-transform: rotate(135deg);-moz-transform: rotate(135deg);transform: rotate(135deg);}

Usage in your html page.

<span class="arrow arrow-left"></span>

**Example Result :**

****

**Adding fonts to your application and URL path should be provide based on your folder structure.**

**@font-face {**

**font-family: 'font name';**

**src: url('../fonts/name.ttf');**

**}**

**Loading icon CSS**

***/\* loader css start\*/***

**.loading-spinner {display: inline-block;position: relative;width: 80px;height: 80px;}**

**.loading-spinner div {**

**-webkit-transform-origin: 40px 40px;**

**-moz-transform-origin: 40px 40px;**

**transform-origin: 40px 40px;**

**-webkit-animation: loading-spinner 1.2s linear infinite;**

**-moz-animation: loading-spinner 1.2s linear infinite;**

**animation: loading-spinner 1.2s linear infinite;**

**}**

**.loading-spinner div:after {**

**content: " ";**

**display: block;**

**position: absolute;**

**top: 3px;**

**left: 37px;**

**width: 6px;**

**height: 18px;**

**border-radius: 20%;**

**background: #909090;**

**}**

**.loading-spinner div:nth-child(1) {-webkit-transform: rotate(0deg);-moz-transform: rotate(0deg);transform: rotate(0deg);animation-delay: -1.1s;}**

**.loading-spinner div:nth-child(2) {**

**-webkit-transform: rotate(30deg);**

**-moz-transform: rotate(30deg);**

**transform: rotate(30deg);**

**-webkit-animation-delay: -1s;**

**-moz-animation-delay: -1s;**

**animation-delay: -1s;**

**}**

**.loading-spinner div:nth-child(3) {**

**-webkit-transform: rotate(60deg);**

**-moz-transform: rotate(60deg);**

**transform: rotate(60deg);**

**-webkit-animation-delay: -0.9s;**

**-moz-animation-delay: -0.9s;**

**animation-delay: -0.9s;**

**}**

**.loading-spinner div:nth-child(4) {**

**-webkit-transform: rotate(90deg);**

**-moz-transform: rotate(90deg);**

**transform: rotate(90deg);**

**animation-delay: -0.8s;**

**}**

**.loading-spinner div:nth-child(5) {-webkit-transform: rotate(120deg);-moz-transform: rotate(120deg);**

**transform: rotate(120deg);**

**animation-delay: -0.7s;**

**}**

**.loading-spinner div:nth-child(6) {**

**-webkit-transform: rotate(150deg);**

**-moz-transform: rotate(150deg);**

**transform: rotate(150deg);**

**animation-delay: -0.6s;**

**}**

**.loading-spinner div:nth-child(7) {**

**-webkit-transform: rotate(180deg);**

**-moz-transform: rotate(180deg);**

**transform: rotate(180deg);**

**animation-delay: -0.5s;**

**}**

**.loading-spinner div:nth-child(8) {**

**-webkit-transform: rotate(210deg);**

**-moz-transform: rotate(210deg);**

**transform: rotate(210deg);**

**animation-delay: -0.4s;**

**}**

**.loading-spinner div:nth-child(9) {**

**-webkit-transform: rotate(240deg);**

**-moz-transform: rotate(240deg);**

**transform: rotate(240deg);**

**animation-delay: -0.3s;**

**}**

**.loading-spinner div:nth-child(10) {**

**-webkit-transform: rotate(270deg);**

**-moz-transform: rotate(270deg);**

**transform: rotate(270deg);**

**animation-delay: -0.2s;**

**}**

**.loading-spinner div:nth-child(11) {**

**-webkit-transform: rotate(300deg);**

**-moz-transform: rotate(300deg);**

**transform: rotate(300deg);**

**-webkit-animation-delay: -0.1s;**

**-moz-animation-delay: -0.1s;**

**animation-delay: -0.1s;**

**}**

**.loading-spinner div:nth-child(12) {**

**-webkit-transform: rotate(330deg);**

**-moz-transform: rotate(330deg);**

**transform: rotate(330deg);**

**-webkit-animation-delay: 0s;**

**-moz-animation-delay: 0s;**

**animation-delay: 0s;**

**}**

**@-webkit-keyframes loading-spinner {0% {opacity: 1;} 100% {opacity: 0;}}**

**@-moz-keyframes loading-spinner {0% {opacity: 1;}100% {opacity: 0;}}**

**@keyframes loading-spinner {0% {opacity: 1;}100% {opacity: 0;}}**

Usage in your html page

<div class="loading-spinner"><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div></div>

**Example Result:**

****

**Spinner or loader type 2**

**<style>**

**.dot-loading-spinner {**

**text-align: center;**

**position: fixed;**

**z-index: 9999;**

**background-color: rgba(255,255,255,.7);**

**top: 55px;**

**left: 0;**

**right: 0;**

**bottom: 0;**

**}**

**.dot-loading-spinner::after {**

**content: '';**

**display: block;**

**font-size: 10px;**

**width: 1em;**

**height: 1em;**

**-webkit-animation: spinner 1500ms infinite linear;**

**-moz-animation: spinner 1500ms infinite linear;**

**-ms-animation: spinner 1500ms infinite linear;**

**-o-animation: spinner 1500ms infinite linear;**

**animation: spinner 1500ms infinite linear;**

**border-radius: 0.5em;**

**-webkit-box-shadow: rgba(0, 0, 0, 0.75) 1.5em 0 0 0, rgba(0, 0, 0, 0.75) 1.1em 1.1em 0 0, rgba(0, 0, 0, 0.75) 0 1.5em 0 0, rgba(0, 0, 0, 0.75) -1.1em 1.1em 0 0, rgba(0, 0, 0, 0.5) -1.5em 0 0 0, rgba(0, 0, 0, 0.5) -1.1em -1.1em 0 0, rgba(0, 0, 0, 0.75) 0 -1.5em 0 0, rgba(0, 0, 0, 0.75) 1.1em -1.1em 0 0;**

**box-shadow: rgba(0, 0, 0, 0.75) 1.5em 0 0 0, rgba(0, 0, 0, 0.75) 1.1em 1.1em 0 0, rgba(0, 0, 0, 0.75) 0 1.5em 0 0, rgba(0, 0, 0, 0.75) -1.1em 1.1em 0 0, rgba(0, 0, 0, 0.75) -1.5em 0 0 0, rgba(0, 0, 0, 0.75) -1.1em -1.1em 0 0, rgba(0, 0, 0, 0.75) 0 -1.5em 0 0, rgba(0, 0, 0, 0.75) 1.1em -1.1em 0 0;**

**position: absolute;**

**top: 50%;**

**left: 50%;**

**}**

**@keyframes spinner {**

**0% {**

**-webkit-transform: rotate(0deg);**

**-moz-transform: rotate(0deg);**

**-ms-transform: rotate(0deg);**

**-o-transform: rotate(0deg);**

**transform: rotate(0deg);**

**}**

**100% {**

**-webkit-transform: rotate(360deg);**

**-moz-transform: rotate(360deg);**

**-ms-transform: rotate(360deg);**

**-o-transform: rotate(360deg);**

**transform: rotate(360deg);**

**}**

**}**

**</style>**

**<div class="dot-loading-spinner"></div>**

**Example Result:**

****

**Vertical custom scroll bar or webkit scrollbar**

**body::-webkit-scrollbar{width: 5px;}**

**body::-webkit-scrollbar-track,{**

**border-radius: 0px;**

**-webkit-box-shadow: inset 0 0 5px grey;**

**-moz-box-shadow: inset 0 0 5px grey;**

**box-shadow: inset 0 0 5px grey;**

**}**

**body::-webkit-scrollbar-thumb{**

**border-radius: 5px;**

**background-color: #c0c0c0;**

**}**

**body:hover::-webkit-scrollbar-thumb{**

**border-radius: 5px;**

**background-color: #545d66;**

**}**

**Example Result:**

****

**Triangle shadow or carrot dropdown arrow shadow**

**.triangle {**

**position: relative;**

**box-shadow: 0px 3px 3px 0 rgba(0, 0, 0, 0.4);**

**}**

**.triangle::after{**

**content: "";**

**position: absolute;**

**width: 0;**

**height: 0;**

**margin-left: -0.5em;**

**bottom: -2em;**

**left: 50%;**

**box-sizing: border-box;**

**border: 1em solid black;**

**border-color: transparent transparent #bada55 #bada55;**

**transform-origin: 0 0;**

**transform: rotate(-45deg);**

**box-shadow: -3px 3px 3px 0 rgba(0, 0, 0, 0.4);**

**}**

**//Usage in html**

**<div class="triangle">triangle with a proper box-shadow!</div>**

**Example Result:**

****

**Success check mark animation**

**<style>**

**.checkmark {**

**width: 56px;**

**height: 56px;**

**border-radius: 50%;**

**display: block;**

**stroke-width: 2;**

**stroke: #fff;**

**stroke-miterlimit: 10;**

**box-shadow: inset 0px 0px 0px #7ac142;**

**animation: fill .4s ease-in-out .4s forwards, scale .3s ease-in-out .9s both;**

**}**

**.checkmark\_\_circle {**

**stroke-dasharray: 166;**

**stroke-dashoffset: 166;**

**stroke-width: 2;**

**stroke-miterlimit: 10;**

**stroke: #7ac142;**

**fill: none;**

**animation: stroke 0.6s cubic-bezier(0.65, 0, 0.45, 1) forwards;**

**}**

**.checkmark\_\_check {**

**transform-origin: 50% 50%;**

**stroke-dasharray: 48;**

**stroke-dashoffset: 48;**

**animation: stroke 0.3s cubic-bezier(0.65, 0, 0.45, 1) 0.8s forwards;**

**}**

**@keyframes stroke {**

**100% {**

**stroke-dashoffset: 0;**

**}**

**}**

**@keyframes scale {**

**0%, 100% {**

**transform: none;**

**}**

**50% {**

**transform: scale3d(1.1, 1.1, 1);**

**}**

**}**

**@keyframes fill {**

**100% {**

**box-shadow: inset 0px 0px 0px 30px #7ac142;**

**}**

**}**

**</style>**

**<svg class="checkmark" xmlns="http://www.w3.org/2000/svg" viewBox="0 0 52 52">**

**<circle class="checkmark\_\_circle" cx="26" cy="26" r="25" fill="none"></circle>**

**<path class="checkmark\_\_check" fill="none" d="M14.1 27.2l7.1 7.2 16.7-16.8"/></svg>**

**Output:** 

**Circular Progress bar CSS javascript**

**<!DOCTYPE html>**

**<html>**

**<head>**

**<style>**

**.svg-circular-progressbar{max-width:100px;}**

**.progress-circle-mask {**

**-webkit-transform: rotate(-90deg);**

**-moz-transform: rotate(-90deg);**

**-ms-transform: rotate(-90deg);**

**transform: rotate(-90deg);**

**}**

**.progress-fill-value {**

**stroke: #4288fd; stroke-linecap: round; stroke-dasharray: 339.292;**

**-webkit-transition: stroke-dashoffset .2s ease; -moz-transition: stroke-dashoffset .2s ease;**

**-ms-transition: stroke-dashoffset .2s ease; transition: stroke-dashoffset .2s ease;**

**}**

**.progress-fill-value[stroke-dashoffset^="339.292"]{stroke-linecap: butt;}**

**</style>**

**</head>**

**<body>**

**<div class="svg-circular-progressbar">**

**<svg class="progress-circle-mask" viewBox="0 0 120 120">**

**<circle cx="60" cy="60" r="54" fill="none" stroke="#ecebf2" stroke-width="8"></circle>**

**<circle class="progress-fill-value" cx="60" cy="60" r="54" fill="none" s**

**troke-width="8"**

**stroke-dashoffset="339.292"></circle>**

**</svg>**

**</div>**

**<input id="control" type="range" value="12" />**

**<script>**

**var control = document.getElementById('control');**

**var progressValue = document.querySelector('.progress-fill-value');**

**var RADIUS = 54;**

**var CIRCUMFERENCE = 2 \* Math.PI \* RADIUS;**

**function progress(value) {**

**var progress = value / 100;**

**var dashoffset = CIRCUMFERENCE \* (1 - progress);**

**console.log('progress:', value + '%', '|', 'offset:', dashoffset);**

**progressValue.setAttribute("stroke-dashoffset",dashoffset);**

**}**

**control.addEventListener('input', function (event) {**

**progress(event.target.valueAsNumber);**

**});**

**progressValue.style.strokeDasharray = CIRCUMFERENCE;**

***//here 12 is percentage value Note we can convert percent (100\*discount/actual)***

**progress(12);**

**</script>**

**</body>**

**</html>**

**Example Result:**

****

**Pie chart using CSS**

**<!DOCTYPE html>**

**<html class=''>**

**<head>**

**<style>**

**.svg-pie-contaner {max-width: 200px;}**

**.svg-progress-pie {**

**-webkit-transform: rotate(-90deg);**

**-moz-transform: rotate(-90deg);**

**-ms-transform: rotate(-90deg);**

**transform: rotate(-90deg);**

**display: block;**

**}**

**.progress-pie-fill-value {**

**fill: red;**

**stroke: #0074d9;**

**stroke-width: 60;**

**stroke-dasharray: 0, 189;**

**-webkit-transition: stroke-dasharray .7s ease;**

**transition: stroke-dasharray .7s ease;**

**}**

**</style>**

**</head>**

**<body>**

**<figure class="svg-pie-contaner">**

**<svg class="svg-progress-pie" viewBox="0 0 120 120">**

**<circle cx="60" cy="60" r="60" fill="#ecebf2"></circle>**

**<circle class="progress-pie-fill-value" r="30" cx="60" cy="60" class="pie" style="stroke-dasharray: 10, 189;" />**

**</svg>**

**</figure>**

**</body>**

**</html>**

**Example Result:**

****

**Back button arrow css**

**.back{width:28px;height:28px;background-color: transparent;border:none;outline:none;position:relative}**

**.back::before {**

**content: "";**

**border: solid #333333;**

**border-width: 0 1px 1px 0;**

**display: inline-block;**

**padding: 5px;**

**-webkit-transform: rotate(135deg);**

**-moz-transform: rotate(135deg);**

**transform: rotate(135deg);**

**position: absolute;**

**top: 9px;**

**left: 8px;**

**}**

**.back::after{content:"";width:20px;height:1px;background-color:#333333;position:absolute;top:14px;**

**left:7px;**

**}**

**Example Result:**

****

**Top header with left center right alignment**

***/\*------top header css start -----\*/***

**header {**

**padding: 5px 42px;**

**box-shadow: 0 -1px 5px -2px rgba(0,0,0,0.14), 0 0px 10px 0px rgba(0,0,0,0.12), 0 1px 0 -2px rgba(0,0,0,0.2);**

**background-color: #fff;**

**}**

**.navbar-container ul li{list-style-type: none;position: relative;}**

**.navbar-container ul> li>.nav-item {position: relative;display: block;padding: 10px 0px;}**

**.navbar-container {display: table;width: 100%;}**

**.navbar-cell-item {display: table-cell;vertical-align: middle;}**

**.navbar-item-left {width: 100px;text-align:left;}**

**.navbar-item-left + .navbar-item-center {text-align: center;}**

**.navbar-item-right {text-align: right;width: 100px;}**

**Example Result:**

****

**Animate circle**

<style>

.animate-circle {

font-size: 23px;

margin: 20px;

position: relative;

padding: 0;

width: 5em;

height: 5em;

background-color: #74b23c;

border-radius: 50%;

line-height: 5em;

}

.animate-circle .inner-circle {

position: absolute;

top: 0;

background-color: #fff;

border-radius: 50%;

width: 5em;

height: 5em;

animation: mymove .3s forwards;

}

@keyframes mymove {

from {transform: scale(1)}

to {transform: scale(0)}

}

</style>

<div class="animate-circle">

<div class="inner-circle"></div>

</div>

**Animate text right to left or scrolling up, down, left, right using marquee tag**

**<marquee behavior="scroll" direction="up" scrollamount="2" loop="3">Slow Scrolling</marquee>**

**<marquee behavior="scroll" direction="right" scrollamount="5" vspace="20">Little Fast Scrolling</marquee>**

**<marquee behavior="scroll" direction="left" scrollamount="5" vspace="10%">Fast Scrolling</marquee>**

**<marquee behavior="scroll" direction="right" scrollamount="5">Very Fast Scrolling</marquee>**

<marquee behavior="scroll" direction="right" onmouseover="this.stop();" onmouseout="this.start();">

<a href="#">link 1</a> <a href="#">link 2</a> <a href="#">link 3</a>

</marquee>

**Animate text right to left using CSS animation**

**<style style="text/css">**

**.marquee {**

**height: 50px;**

**overflow: hidden;**

**position: relative;**

**background: #fefefe;**

**color: #333;**

**border: 1px solid #4a4a4a;**

**}**

**.marquee p {**

**position: absolute;**

**width: 100%;**

**height: 100%;**

**margin: 0;**

**line-height: 50px;**

**text-align: center;**

**transform: translateX(100%);**

**animation: scroll-left 20s linear infinite;**

**}**

**@keyframes scroll-left {**

**0% {transform: translateX(100%);}**

**100% {transform: translateX(-100%);}**

**}**

**</style>**

**<div class="marquee">**

**<p> Marquee in CSS </p>**

**</div>**

**Alignment flex vertical center, horizontal center, left right**

.flex-container {

display: -webkit-box; */\*OLD - iOS 6-, Safari 3.1-6 \*/*

display: -moz-box; */\*OLD - Firefox 19 \*/*

display: -ms-flexbox; */\*TWEENER - IE 10 \*/*

display: -webkit-flex; */\*NEW - Chrome \*/*

display: flex; */\* NEW, Spec - Opera 12.1, Firefox 20+ \*/*

}

.flex-item {

-webkit-box-flex: 1; */\* OLD - iOS 6-, Safari 3.1-6 \*/*

-moz-box-flex: 1; */\* OLD - Firefox 19- \*/*

-webkit-flex: 1; */\* Chrome \*/*

-ms-flex: 1; */\* IE 10 \*/*

flex: 1; */\* NEW, Spec - Opera 12.1, Firefox 20+ \*/*

}

.align-items-center {

-ms-flex-wrap: wrap;

flex-wrap: wrap;

-webkit-align-items: center;

-ms-flex-align: center;

align-items: center;

-webkit-justify-content: center;

-moz-justify-content: center;

justify-content: center;

}

.navbar-left {

-webkit-justify-content: flex-start;

-moz-justify-content: flex-start;

justify-content: flex-start;

}

.navbar-right {

-webkit-justify-content: flex-end;

-moz-justify-content: flex-end;

justify-content: flex-end;

}

.space-between{

-webkit-justify-content: space-between;

-moz-justify-content: space-between;

justify-content: space-between;

}

**Masking text on bottom as a layer**

**.mask-bottom{position:relative}**

**.mask-bottom::before {**

**content: "";**

**position: absolute;**

**z-index: 1;**

**left: 0px;**

**bottom: 0px;**

**width: 100%;**

**height: 90px;**

**background-image: -webkit-linear-gradient(rgba(255, 255, 255, 0.2), rgba(255, 255, 255, 0.95));**

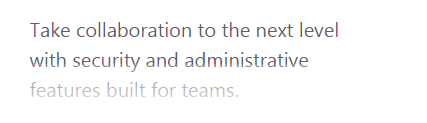
**background-image: -moz-linear-gradient(rgba(255, 255, 255, 0.2), rgba(255, 255, 255, 0.95));**

**background-image: -ms-linear-gradient(rgba(255, 255, 255, 0.2), rgba(255, 255, 255, 0.95));**

**background-image: linear-gradient(rgba(255, 255, 255, 0.2), rgba(255, 255, 255, 0.95));**

**}**

**Example:**

****

**Bootstrap 4 modal vertically and horizontally center css**

**.modal .modal-dialog {**

**-webkit-transform: translate(0,-50%);**

**-o-transform: translate(0,-50%);**

**transform: translate(0,-50%);**

**top: 50%;**

**margin: 0 auto;**

**}**

**Bootstrap 3 modal vertically and horizontally center**

**.modal {**

**text-align: center;**

**padding: 0!important;**

**}**

**.modal:before {**

**content: '';**

**display: inline-block;**

**height: 100%;**

**vertical-align: middle;**

**margin-right: -4px;**

**}**

**.modal-dialog {**

**display: inline-block;**

**text-align: left;**

**vertical-align: middle;**

**}**

**Bootstrap modal**

**<button class="btn btn-primary" data-toggle="modal" data-target="#myModal"></button>**

**<div class="modal fade" id="myModal3" tabindex="-1" role="dialog" aria-labelledby="myModal3Label" aria-hidden="true">**

**<div class="modal-dialog">**

**<div class="modal-content">**

**<div class="modal-header">**

**<button type="button" class="close" data-dismiss="modal" aria-hidden="true">&times;</button>**

**</div>**

**<div class="modal-body"></div>**

**</div>**

**</div>**

**</div>**

**Switch toggle No yes**

<!DOCTYPE html>

<html>

<head>

  <style>

    .can-toggle {

     position: relative;

    }

    .can-toggle \*, .can-toggle \*:before, .can-toggle \*:after {

     box-sizing: border-box;

    }

    .can-toggle input[type=checkbox] {

     opacity: 0;

     position: absolute;

     top: 0;

     left: 0;

    }

    .can-toggle input[type=checkbox][disabled] ~ label {

     pointer-events: none;

    }

    .can-toggle input[type=checkbox][disabled] ~ label .can-toggle\_\_switch {

     opacity: 0.4;

    }

    .can-toggle input[type=checkbox]:checked ~ label .can-toggle\_\_switch:before {

     content: attr(data-unchecked);

     left: 0;

    }

    .can-toggle input[type=checkbox]:checked ~ label .can-toggle\_\_switch:after {

     content: attr(data-checked);

    }

    .can-toggle label {

     -webkit-user-select: none;

     -moz-user-select: none;

     -ms-user-select: none;

     user-select: none;

     position: relative;

     display: flex;

     align-items: center;

    }

    .can-toggle label .can-toggle\_\_label-text {

     flex: 1;

     padding-left: 32px;

    }

    .can-toggle label .can-toggle\_\_switch {

     position: relative;

    }

    .can-toggle label .can-toggle\_\_switch:before {

     content: attr(data-checked);

     position: absolute;

     top: 0;

     text-transform: uppercase;

     text-align: center;

    }

    .can-toggle label .can-toggle\_\_switch:after {

     content: attr(data-unchecked);

     position: absolute;

     z-index: 5;

     text-transform: uppercase;

     text-align: center;

     background: #1b365d;

     transform: translate3d(0, 0, 0);

    }

    .can-toggle input[type=checkbox][disabled] ~ label {

     color: rgba(119, 119, 119, 0.5);

    }

    .can-toggle input[type=checkbox]:checked ~ label .can-toggle\_\_switch {

     background-color: #cccccc;

    }

    .can-toggle input[type=checkbox]:checked ~ label .can-toggle\_\_switch:after {

     color: #fff;

    }

    .can-toggle label .can-toggle\_\_switch {

     transition: background-color 0.3s cubic-bezier(0, 1, 0.5, 1);

     background: #cccccc;

    }

    .can-toggle label .can-toggle\_\_switch:before {

     color: #979191;

    }

    .can-toggle label .can-toggle\_\_switch:after {

     transition: transform 0.3s cubic-bezier(0, 1, 0.5, 1);

     color: #fff;

    }

    .can-toggle input[type=checkbox]:checked ~ label .can-toggle\_\_switch:after {

     transform: translate3d(80%, 0, 0);

    }

    .can-toggle label {

     font-size: 13px;

     margin-bottom: 0;

    }

    .can-toggle label .can-toggle\_\_switch {

     height: 26px;

     flex: 0 0 100px;

     border-radius: 20px;

    }

    .can-toggle label .can-toggle\_\_switch:before,

    .can-toggle label .can-toggle\_\_switch::after {

     display: flex;

     justify-content: center;

     align-items: center;

    }

    .can-toggle label .can-toggle\_\_switch:before {

     left: 50%;

     height: 100%;

     width: 50%;

     padding: 0px 9px 0px 5px;

    }

    .can-toggle label .can-toggle\_\_switch::after {

     top: 0px;

     left: 0px;

     border-radius: 20px;

     width: 55%;

     height: 100%;

    }

  </style>

</head>

<body>

  <div class="can-toggle">

    <input id="b" type="checkbox">

    <label for="b">

      <div class="can-toggle\_\_switch" data-checked="Yes" data-unchecked="No"></div>

    </label>

  </div>

</body>

</html>

**Out put:** 

**JavaScript**

**Difference between two dates**

**function daysBetween(startDate, endDate) {**

**var millisecondsPerDay = 1000 \* 60 \* 60 \* 24;**

**var startDateUTC = Date.UTC(startDate.getFullYear(), startDate.getMonth(), startDate.getDate());**

**var endDateUTC = Date.UTC(endDate.getFullYear(), endDate.getMonth(), endDate.getDate());**

**return Math.floor((endDateUTC - startDateUTC) / millisecondsPerDay);**

**}**

**Usage Example:**

**daysBetween(new Date("2019-06-08T13:56:31.828Z"), new Date("2019-06-12T13:56:31.828Z"))**

**OR**

**daysBetween(new Date("2019-06-08"), new Date("2019-06-12"))**

**Difference between two date (it works well)**

function \_getDaysDiff(start, end) {

  /\* \_getDaysDiff(new Date("2021-07-09"), new Date("2021-07-12")) \*/

  if(arguments.length < 2 || Object.prototype.toString.call(start) !== "[object Date]" || Object.prototype.toString.call(end) !== "[object Date]") return "Invalid dates provided";

  // return Math.abs(Math.floor(new Date(end).getTime() / (3600 \* 24 \* 1000)) - Math.floor(new Date(start).getTime() / (3600 \* 24 \* 1000)));

  const MILLISECONDS\_IN\_DAY = 86400000;

  const MILLISECONDS\_IN\_MINUTE = 60000;

  function getTimezoneOffsetInMilliseconds (dirtyDate) {

    var date = new Date(dirtyDate.getTime())

    var baseTimezoneOffset = date.getTimezoneOffset()

    date.setSeconds(0, 0)

    var millisecondsPartOfTimezoneOffset = date.getTime() % MILLISECONDS\_IN\_MINUTE

    return baseTimezoneOffset \* MILLISECONDS\_IN\_MINUTE + millisecondsPartOfTimezoneOffset

  }

  function toDate(argument) {

    if (arguments.length < 1) {

      throw new TypeError(

        '1 argument required, but only ' + arguments.length + ' present'

      )

    }

    const argStr = Object.prototype.toString.call(argument)

    // Clone the date

    if (

      argument instanceof Date ||

      (typeof argument === 'object' && argStr === '[object Date]')

    ) {

      // Prevent the date to lose the milliseconds when passed to new Date() in IE10

      return new Date(argument.getTime())

    } else if (typeof argument === 'number' || argStr === '[object Number]') {

      return new Date(argument)

    } else {

      if (

        (typeof argument === 'string' || argStr === '[object String]') &&

        typeof console !== 'undefined'

      ) {

        // eslint-disable-next-line no-console

        console.warn(

          "Starting with v2.0.0-beta.1 date-fns doesn't accept strings as arguments. Please use `parseISO` to parse strings. See: https://git.io/fjule"

        )

        // eslint-disable-next-line no-console

        console.warn(new Error().stack)

      }

      return new Date(NaN)

    }

  }

  function startOfDay(dirtyDate) {

    if (arguments.length < 1) {

      throw new TypeError(

        '1 argument required, but only ' + arguments.length + ' present'

      )

    }

    var date = toDate(dirtyDate)

    date.setHours(0, 0, 0, 0)

    return date

  }

  function compareAsc(dirtyDateLeft, dirtyDateRight) {

    if (arguments.length < 2) {

      throw new TypeError(

        '2 arguments required, but only ' + arguments.length + ' present'

      )

    }

    var dateLeft = toDate(dirtyDateLeft)

    var dateRight = toDate(dirtyDateRight)

    var diff = dateLeft.getTime() - dateRight.getTime()

    if (diff < 0) {

      return -1

    } else if (diff > 0) {

      return 1

      // Return 0 if diff is 0; return NaN if diff is NaN

    } else {

      return diff

    }

  }

  function differenceInCalendarDays(

    dirtyDateLeft,

    dirtyDateRight

  ) {

    if (arguments.length < 2) {

      throw new TypeError(

        '2 arguments required, but only ' + arguments.length + ' present'

      )

    }

    var startOfDayLeft = startOfDay(dirtyDateLeft)

    var startOfDayRight = startOfDay(dirtyDateRight)

    var timestampLeft =

      startOfDayLeft.getTime() - getTimezoneOffsetInMilliseconds(startOfDayLeft)

    var timestampRight =

      startOfDayRight.getTime() - getTimezoneOffsetInMilliseconds(startOfDayRight)

    // Round the number of days to the nearest integer

    // because the number of milliseconds in a day is not constant

    // (e.g. it's different in the day of the daylight saving time clock shift)

    return Math.round((timestampLeft - timestampRight) / MILLISECONDS\_IN\_DAY)

  }

  var dateLeft = toDate(end)

  var dateRight = toDate(start)

  var sign = compareAsc(dateLeft, dateRight)

  var difference = Math.abs(differenceInCalendarDays(dateLeft, dateRight))

  dateLeft.setDate(dateLeft.getDate() - sign \* difference)

  // Math.abs(diff in full days - diff in calendar days) === 1 if last calendar day is not full

  // If so, result must be decreased by 1 in absolute value

  var isLastDayNotFull = compareAsc(dateLeft, dateRight) === -sign

  var result = sign \* (difference - isLastDayNotFull)

  // Prevent negative zero

  return result === 0 ? 0 : result

}

**Date limit based on date select. OR Enable between dates days only based on selected date**

**var tomorrrow = new Date(newDate);**

**tomorrrow.setDate(tomorrrow.getDate() + 5);**

**var maxDate2=new Date(tomorrrow);**

**var date2=new Date(newDate);**

**Date formats**

**//ISO dates can be written with added hours, minutes, and seconds (YYYY-MM-DDTHH:MM:SSZ):**

**function setDateFormat(date, format) {**

**format = format || "";**

**var monthStr = ['Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun', 'Jul', 'Aug', 'Sep', 'Oct', 'Nov', 'Dec'];**

**var dateObj = new Date(date);**

**var d = dateObj.getDate();**

**var m = dateObj.getMonth();**

**var y = dateObj.getFullYear();**

**var hr = dateObj.getHours();**

**var ampm = hr >= 12 ? 'pm' : 'am';**

**var ampmhrs = ( hr % 12 );**

**ampmhrs = ampmhrs ? ampmhrs : 12; *// the hour '0' should be '12'***

**var min = dateObj.getMinutes();**

**var sec = dateObj.getSeconds();**

**function pad(number) {return ("0" + number).slice(-2);}**

**if(format === "YYYY-MM-DD") return ( y + '-' + (m <= 9 ? '0' + (m+1) : (m+1)) + "-" + pad(d) );**

**if(format === "MMM DD, MM:SS") return ( monthStr[m] + ' ' + pad(d) + ", " + pad(min) + ":" + pad(sec) );**

**if(format === "MMM DD, HH:MM-12ampm") return ( monthStr[m] + ' ' + pad(d) + ", " + ampmhrs + ':' + pad(min) + ' ' + ampm );**

**if(format === "MMM DD, HH:MM-24ampm") return ( monthStr[m] + ' ' + pad(d) + ", " + pad(hr) + ":" + pad(min) + ' ' + ampm );**

**if(format === "MMM DD, HH:MM") return ( monthStr[m] + ' ' + pad(d) + ", " + pad(hr) + ":" + pad(min) );**

**return ( monthStr[m] + " " + pad(d) + " " + y );**

**}**

**//setDateFormat(dateString) || setDateFormat(new Date(), 'YYYY-MM-DD')**

**//setDateFormat(new Date(), 'MMM DD, MM:SS') || setDateFormat(date, 'MMM DD, HH:MM-12ampm')**

**//Example Result: Jan 01 2019 || 2019-01-01 || Dec 15, 23:55 || Jan 11, 1:03 pm || Jan 11, 13:03**

**Timer countdown months days hours minutes seconds format**

**function timerInExpiring(secondsCount, timerCallback, expireCallback) {**

**function padDecimal(number) {return ("0" + number).slice(-2);}**

**var \_remainingTimeSeconds = secondsCount;**

**var calculateTimer = function() {**

**var days = Math.floor(\_remainingTimeSeconds / 24 / 60 / 60);**

**var months = Math.floor(\_remainingTimeSeconds / 2628002.88);**

**var dLeft = Math.floor(days % 30);**

**var hoursLeft = Math.floor((\_remainingTimeSeconds) - (days \* 86400));**

**var hours = Math.floor(hoursLeft / 3600);**

**var minutesLeft = Math.floor((hoursLeft) - (hours \* 3600));**

**var minutes = Math.floor(minutesLeft / 60);**

**var remainingSeconds = \_remainingTimeSeconds % 60;**

**return {**

**'difference': \_remainingTimeSeconds,**

**'months': (months > 0 ? months : 0),**

**'dLeft': (dLeft > 0 ? dLeft : 0),**

**'days': (days > 0 ? days : 0),**

**'hours': padDecimal((hours > 0 ? hours : 0)),**

**'minutes': padDecimal(minutes > 0 ? minutes : 0),**

**'seconds': padDecimal(remainingSeconds > 0 ? remainingSeconds : 0),**

**'h': (hours > 0 ? hours : 0),**

**'m': minutes > 0 ? minutes : 0,**

**'s': (remainingSeconds > 0 ? remainingSeconds : 0)**

**};**

**}**

**var startCountdown = function() {**

**function updateClock() {**

**var \_timerDetails = calculateTimer();**

**if (timerCallback) {**

**timerCallback(\_timerDetails)**

**}**

**if (\_timerDetails.difference <= 0) {**

**if (expireCallback) {**

**expireCallback('expired')**

**}**

**clearInterval(\_timeinterval);**

**} else {**

**\_remainingTimeSeconds--;**

**}**

**}**

**updateClock();**

**var \_timeinterval = setInterval(updateClock, 1000);**

**};**

**startCountdown();**

**}**

**timerInExpiring(6825600, function(\_t) {**

**document.getElementById("demo").innerHTML = \_t.months + ":Months " + \_t.dLeft + ":daysLeft " + \_t.hours + ":h " + \_t.minutes + ":m " + \_t.seconds + ":s ";**

**}, function(expiredRes) {**

**console.log("Expired")**

**});**

**Example Result:**



**Timer countdown using expiry date and current data**

**function getTimeRemaining(expiryDate) {**

**var dif = new Date(expiryDate).getTime() - new Date().getTime();**

**var days = Math.floor(dif / (1000 \* 60 \* 60 \* 24));**

**var hours = Math.floor((dif % (1000 \* 60 \* 60 \* 24)) / (1000 \* 60 \* 60));**

**var minutes = Math.floor((dif % (1000 \* 60 \* 60)) / (1000 \* 60));**

**var seconds = Math.floor((dif % (1000 \* 60)) / 1000);**

**function pad(number) {return ("0" + number).slice(-2);}**

**return {**

**'difference': dif,**

**'days': (days > 0 ? days : 0),**

**'hours': (hours > 0 ? hours:0),**

**'minutes': pad(minutes),**

**'seconds': pad(seconds),**

**'fullview': function () {**

**if(days > 0) return (((this.days <10) ? this.days + "days, " : this.days + "day, ") + ((this.hours <10) ? this.hours + "hour" : this.hours + "hours") );**

**if(hours > 0 ) return (this.hours + "h:" + this.minutes + "m:" + this.seconds + "s");**

**if(minutes > 0 ) return (this.minutes + "m:" + this.seconds + "s");**

**return ((seconds > 0) ? ( this.seconds + "s") : 0);**

**}**

**};**

**}**

**function initializeTimer(endtime) {**

**function updateClock() {**

**var timeobject = getTimeRemaining(endtime);**

**document.getElementById('demo').innerHTML = timeobject.fullview();**

**if (timeobject.difference <= 0) {**

**clearInterval(timeinterval);**

**}**

**}**

**updateClock();**

**timeinterval = setInterval(updateClock, 1000);**

**}**

**var deadline = "2020-01-01T03:00:00.000Z";**

**initializeTimer(deadline);**

**Example Result:**

 **OR** 

**Find Index in array object based on id**

**function findIndexInJsonObj(array, prop, value) {**

***//Data.map(e => e.name).indexOf('Nick')***

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

**if(array[i][prop] === value) {**

**return i;**

**}**

**}**

**return -1;**

**}**

**Distance between two places based on latitude and longitude**

**function distance(lat1, lon1, lat2, lon2, unit) {**

**var radlat1 = Math.PI \* lat1/180**

**var radlat2 = Math.PI \* lat2/180**

**var radlon1 = Math.PI \* lon1/180**

**var radlon2 = Math.PI \* lon2/180**

**var theta = lon1-lon2**

**var radtheta = Math.PI \* theta/180**

**var dist = Math.sin(radlat1) \* Math.sin(radlat2) + Math.cos(radlat1) \* Math.cos(radlat2) \* Math.cos(radtheta);**

**dist = (Math.acos(dist) \* 180/Math.PI) \* 60 \* 1.1515;**

**if (unit=="K") { dist = dist \* 1.609344 }**

**if (unit=="N") { dist = dist \* 0.8684 }**

**return dist**

**}**

**Query selector for loop (iteration over nodelist ) all browser support**

**var divs = document.querySelectorAll('div');**

**[].forEach.call(divs, function(element) {**

**element.style.color = "red";**

**});**

**Accessng the current URL in parts. Example:**

**http://css-tricks.com/example/index.html?s=flexbox**

**window.location.protocol => "http:"**

**window.location.host => "css-tricks.com"**

**window.location.pathname => "example/index.html"**

**window.location.search => "?s=flexbox"**

**window.location.origin => "http://css-tricks.com"**

**JSON formatter or Print Object as json serialiser like angular json filter on html page**

**function print\_jsonView(o) {**

**return JSON.stringify(o,null,'\t').replace(/\n/g,'<br>').replace(/\t/g,'&nbsp;&nbsp;&nbsp;');**

**}**

**//Example**

**var d= [{"lunch": "sandwich","dinner": "stirfry"},{"lunch": "sandwich","dinner": "stirfry"}];**

**document.getElementById("#put-here").innerHTML= print\_jsonView(myObject);;**

**Select Random item in Array**

**var myArray = [**

**"Apples",**

**"Bananas",**

**"Pears"**

**];**

**var randomItem = myArray[Math.floor(Math.random()\*myArray.length)];**

**Horizontal Scroll progress bar**

**<html>**

**<body>**

**<style>**

**body {height: 1500px;}**

**.scroll-progress {**

**background: #72ba5e;**

**position: fixed;**

**width: 0%;**

**height: 4px;**

**z-index: 3;**

**top: 0;**

**left: 0;**

**}**

**</style>**

**<div class="scroll-progress"></div>**

**<script>**

**var h = document.documentElement,**

**b = document.body,**

**st = 'scrollTop',**

**sh = 'scrollHeight',**

**progress = document.querySelector('.scroll-progress'),**

**scroll;**

**document.addEventListener('scroll', function() {**

**scroll = (h[st] || b[st]) / ((h[sh] || b[sh]) - h.clientHeight) \* 100;**

**progress.style.width = scroll + '%';**

**}, {**

**passive: true**

**});**

**</script>**

**</body>**

**</html>**

**html5 and custom tag browser support**

**// In header tag add below code**

**<script>**

**(function() {**

**var e = "abbr,article,aside,audio,mytag".split(','),**

**i = e.length;**

**while (i--) {**

**document.createElement(e[i])**

**}**

**})();**

**</script>**

**//OR**

**<!--[if lt IE 9]>**

**<script> document.createElement("mytag"); </script>**

**<![endif]-->**

**Log Array Data for debug with console.log() Example**

Instead of calling console.log(), we'll use console.table() now

**<script>**

**var languages = [**

**{ name: "JavaScript", fileExtension: ".js" },**

**{ name: "TypeScript", fileExtension: ".ts" },**

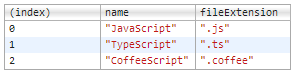
**{ name: "CoffeeScript", fileExtension: ".coffee" }**

**];**

**console.table(languages)**

**</script>**

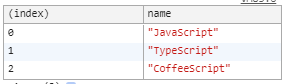
**Example**: By using method the view should be like below and check in your developer tool after render.



// keys wise printing

**console.table(languages, ["name"]);**

**Result**:



**Internet Explorer detection (IE detect)**

**<script>**

**if (/MSIE (\d+\.\d+);/.test(navigator.userAgent)) { *//test for MSIE x.x;***

**var ieversion=new Number(RegExp.$1) *// capture x.x portion and store as a number***

**console.table(languages)**

**}**

**</script>**

**Deep copy object or array**

**var name = JSON.parse(JSON.sringify(data));**

**IE Support add classes to html tag**

**<!doctype html>**

**<!--[if lt IE 7]><html class="lt-ie6"><![endif]-->**

**<!--[if IE 7]><html class="lt-ie7"><![endif]-->**

**<!--[if IE 8]><html class="lt-ie8"><![endif]-->**

**<!--[if gt IE 8]> <html class="lt-ie9"><![endif]-->**

**Counter Animate or Animation number increment**

**function animateNumber(elem, from, to, duration) {**

**var interval = setInterval(function() {**

**if (from >= to) clearInterval(interval);**

**elem.innerText = from++;**

**}, duration);**

**}**

**animateNumber(document.getElementById("demo"), 0, 100, 6);**

**Get percentage of a value**

**function getPercentOfValue(percent, totalValue, decimal) {**

**return decimal ? ( (percent/100) \* totalValue ).toFixed(decimal) : ( (percent/100) \* totalValue )**

**}**

**getPercentOfValue(20, 200)**

***//Example Result: 40***

**Get percentage from value**

**function getPercent(consumedValue, totalValue, decimal) {**

**return decimal ? ( (100\*consumedValue) / totalValue).toFixed(decimal) : ( (100\*consumedValue) / totalValue)**

**}**

**getPercent(20, 80, 2)**

***//Example Result: 25.00***

**Get remaining value**

**function getRemaingValue(consumedValue, totalValue){**

**consumedValue = consumedValue ? (consumedValue.toString()).replace(/[^0-9.]/g,'') : 0;**

**totalValue = totalValue ? (totalValue.toString()).replace(/[^0-9.]/g,'') : 0;**

**return ( totalValue - consumedValue );**

**}**

**getRemaingValue(50, 170);**

***//Example Result: 120***

**Creating new array object with existing array object**

**var languages = [{ name: "JavaScript", fruit: "orange" },{ name: "CoffeeScript", fruit: "banana" }];**

**var newObj = languages.map((item) => {**

**return ({"keyname":item.name});**

**});**

***//Example Result:***

**newObj = [{"keyname":"JavaScript"},{"keyname":"CoffeeScript"}]**

**OR || Get Array with specific properties from Array or select \* from Array format**

**function queryProps(paramData, ...params) {**

**if(!paramData.length) return paramData;**

**return paramData.map((item)=>{**

**return params.reduce(function(result, key) {**

**result[key] = item[key];**

**return result;**

**}, {});**

**})**

**}**

**var arr=[{"name":"Bhargav","gender":"male"},{"name":"guna","gender":"male"}];**

**queryProps(arr, "name");**

***//Example Result: [{"name":"Bhargav"},{"name":"guna"}***

**Multi filter array**

**var productArray = [**

**{ name: "A", color: "Blue red", size: 50 },**

**{ name: "B", color: "Blue", size: 60 },**

**{ name: "C", color: "Black", size: 70 },**

**{ name: "D", color:"Green", size: 50 },**

**];**

**Combination key filter from json**

**var filter = {**

**address: 'England',**

**name: 'Mark'**

**};**

**var users = [{**

**name: 'John',**

**age: 25,**

**address: 'USA'**

**},**

**{**

**name: 'Tom',**

**age: 35,**

**address: 'England'**

**},**

**{**

**name: 'Mark',**

**age: 28,**

**address: 'England'**

**}**

**];**

**users= users.filter(function(item) {**

**for (var key in filter) {**

**if (item[key] === undefined || item[key] != filter[key])**

**return false;**

**}**

**return true;**

**});**

**Canvas to Image toDataURL conversion**

**<!DOCTYPE html>**

**<html>**

**<body>**

**<p>Image to canvas to DataURL image</p>**

**<img width="1200" height="800" id="scream" src="pic\_the\_scream.jpg" alt="The Scream">**

**<p>Canvas:</p>**

**<canvas id="myCanvas" width="1200" height="800" style="border:1px solid #d3d3d3;">**

**Your browser does not support the HTML5 canvas tag.**

**</canvas>**

**<button onclick="send()">Generate toDataURL</button>**

**<script>**

**window.onload = function() {**

**var canvas = document.getElementById("myCanvas");**

**var ctx = canvas.getContext("2d");**

**var img = document.getElementById("scream");**

**ctx.drawImage(img, 10, 10);**

**};**

**function generateDataUrl() {**

**var canvas = document.getElementById("myCanvas");**

**console.log(canvas.toDataURL())**

**}**

**</script>**

**</body>**

**</html>**

**Image to Dataurl**

<!DOCTYPE html>

<html>

<body>

<style>body{background:orange}</style>

<img id="ddd" src="data:image/png;base64,iVBORw0KGgoAAAANSUhEUgAAAAwAAAANCAYAAACdKY9CAAAAyklEQVQoU62SPQ5BURCFv1PqLMECJNRCQqFnBdZgBWIFloDWJigkWmqNVqcT1ZF5ee/l+m/c5OYmM3NyfuaKH8d2HThKusWoinnbVWAiaZzUFsAI2AM9SZcU0AXWeWNjuxgu8AFqvQUANWAOLHOG4u18Y2gCITNjBU7A+SMgdNh+kPls+qX5P4DtMBm31Cvp9JbBdh9YAVNgliQzBC5p1JkH2xVgBzSSpR+A8BRJlbspTedb3uSgbDi2mtQHIfE5pWBqA1tJ109f7A7fI3RTNqVpFAAAAABJRU5ErkJggg==" />

<img id="toDataUrl" />

<button onclick="imageToDataUrl()">click</button>

<script>

function imageToDataUrl() {

var canvas = document.createElement("canvas");;

var ctx = canvas.getContext('2d');

var img = document.getElementById("ddd");

canvas.width = img.width;

canvas.height = img.height;

*// Draw the image*

ctx.drawImage(img, 0, 0);

document.body.appendChild(canvas);

document.getElementById("toDataUrl").src=canvas.toDataURL('image/png');

console.log("url", canvas.toDataURL('image/png'));

canvas.remove();

}

</script>

</body>

</html>

**Converting file size as bytes to KB/MB/GB ------- YB** ***// this function for converting file size to kb or mb***

**function formatFileSize(bytes,decimalPoint) {**

**if(bytes == 0) return '0 Bytes';**

**const k = 1024;**

**const dm = decimalPoint || 2;**

**const sizes = ['Bytes', 'KB', 'MB', 'GB', 'TB', 'PB', 'EB', 'ZB', 'YB'];**

**const i = Math.floor(Math.log(bytes) / Math.log(k));**

**return parseFloat((bytes / Math.pow(k, i)).toFixed(dm)) + ' ' + sizes[i];**

**}**

**Convert number to string**

**function toString(value){**

**value = value || "";**

**return value.toString();**

**}**

**toString(50);**

**//Example Result: "50"**

**Print Random item from array**

**var myArray = [**

**"Apples",**

**"Bananas",**

**"Pears"**

**];**

**var randomItem = myArray[Math.floor(Math.random()\*myArray.length)];**

**Converting upload file as base64 img/pdf OR Upload file**

**<!DOCTYPE html>**

**<html>**

**<head>**

**<title>File upload and data url</title>**

**</head>**

**<body>**

**<input name="uploadifle" type="file" class="upload" accept="\*" onchange="changeListener(event)">**

**<script>**

**function changeListener($event) {**

**var file = $event.target.files[0];**

**var myReader = new FileReader();**

**myReader.readAsDataURL(file);**

**myReader.onloadend = (e) => {**

**console.log('myReader.result: ', myReader.result)**

**}**

**}**

**</script>**

**</body>**

**</html>**

**Extract only number from string**

***/\* --------- this method for extracting only number from string -------- \*/***

**function extractNumber(value) {**

**value = value || "";**

**return value.replace(/[^0-9]/g,"")**

**}**

**Close window**

function closeWindow(){

var win=window.open("","\_self");

win.close();

}

Usage in HTML

<button onclick="closeWindow()">Close window</button>

NOTE: If want to close the window then we can close only the windows that were opened by it. Example any

Page that which is loaded on through navigation that page only can close other wise wont close it returns

Warning. Example: Scripts may close only the windows that were opened by it.

**Pure javascript xmlhttprequest or javascript ajax call**

***/\* --------- RestAPI or http post response call -------- \*/***

**function handlePostRequest(options, successCallback, errorCallback) {**

**options = options || {};**

**var client;**

**if (window.XMLHttpRequest) {**

***// code for IE7+, Firefox, Chrome, Opera, Safari***

**client=new XMLHttpRequest();**

**} else { *// code for IE6, IE5, 'ActiveXObject' in window***

**client=new ActiveXObject("Microsoft.XMLHTTP");**

**}**

**client.onreadystatechange = function() {**

**if (this.readyState == 4 && this.status == 200) {**

**if(successCallback){**

**successCallback(client.response);**

**}**

**}**

**if(this.readyState == 4 && this.status !== 200){**

**if(errorCallback){errorCallback(this.status);}**

**}**

**};**

**client.onerror = function () {**

**if(errorCallback){errorCallback(this.status);}**

**}**

**client.open("GET",options.url,true);**

**client.send();**

***//client.open("POST", options.url,true);***

***//client.setRequestHeader("Content-Type", "text/plain;charset=UTF-8");***

***//client.send(options.data);***

**}**

**Get request handling with API using jQuery Ajax**

**function handleGetRequest(options, successCallback, errorCallback) {**

**options = options || {};**

**$.ajax({**

**type: 'GET',**

**url: options.url,**

**}).then(function(response) {**

**if(successCallback){**

**successCallback(response);**

**}**

**}, function(xhr, errorType, exception) {**

**if(errorCallback){errorCallback(xhr);}**

**console.log("section responseText: ", xhr);**

**});**

**}**

**//Usage in your file**

**handleGetRequest({"url":(environment.api+"/getusg")},function(res){**

**console.log("data: ",JSON.stringify(res));**

**}, function(error){**

**console.log("error: ",error);**

**});**

**Post form data or multipart form data using Ajax**

***/\* --------- RestAPI or http post response call -------- \*/***

**function handlePostRequest(options, successCallback, errorCallback) {**

**options = options || {};**

**$.ajax({**

**type: 'POST',**

**url: options.url,**

**data:options.data**

**}).then(function(response) {**

**if(successCallback){**

**successCallback(response);**

**}**

**}, function(xhr, errorType, exception) {**

**if(errorCallback){errorCallback(xhr);}**

**console.log("section responseText: ", xhr);**

**});**

**}**

**function sendFormData(data){**

**const formData = new FormData();**

**formData.append('file', data[0].file.files[0]);**

**formData.append('firstName', data[0].firstName);**

**formData.append('lastName', data[0].lastName);**

**handleGetRequest({"url":(environment.api,"data":formData}, function(res){**

**console.log("data: ",JSON.stringify(res));**

**}, function(error){**

**console.log("error: ",error);**

**});**

**}**

Bootstrap modal body dynamic height using jquery

**function setModalMaxHeight(element) {**

**this.$element = $(element);**

**this.$content = this.$element.find('.modal-content');**

**var borderWidth = this.$content.outerHeight() - this.$content.innerHeight();**

**var dialogMargin = $(window).width() < 768 ? 20 : 60;**

**var contentHeight = $(window).height() - (dialogMargin + borderWidth);**

**var headerHeight = this.$element.find('.modal-header').outerHeight() || 0;**

**var footerHeight = this.$element.find('.modal-footer').outerHeight() || 0;**

**var maxHeight = contentHeight - (headerHeight + footerHeight);**

**this.$content.css({**

**'overflow': 'hidden'**

**});**

**this.$element**

**.find('.modal-body').css({**

**'max-height': maxHeight,**

**'overflow-y': 'auto'**

**});**

**}**

**$('.modal').on('show.bs.modal', function() {**

**$(this).show();**

**setModalMaxHeight(this);**

**});**

**$(window).resize(function() {**

**if ($('.modal.in').length != 0) {**

**setModalMaxHeight($('.modal.in'));**

**}**

**});**

**Page views count or number of times page visits**

**<!DOCTYPE html>**

**<html>**

**<body>**

**<div id="counter"></div>**

**<button id="reset">Reset</button>**

**<script>**

**var visitedCount = window.sessionStorage.getItem('on\_load\_counter');**

**if (visitedCount === null) {**

**visitedCount = 0;**

**}**

**visitedCount++;**

**window.sessionStorage.setItem("on\_load\_counter", visitedCount);**

**document.getElementById('counter').innerHTML = visitedCount;**

**function reset\_counter() {**

**window.sessionStorage.removeItem('on\_load\_counter');**

**}**

**document.getElementById('reset').addEventListener('click', reset\_counter);**

**</script>**

**</body>**

**</html>**

**User inactivity or idle detect on window**

***/\*--- This method for to detect idle (in-activity) on window ---\*/***

**function onIdleDetect(timeOutSeconds,callback){**

***//Idle detection***

**var \_idleTimeout;**

**var \_userActivity=function() {**

**window.clearTimeout(\_idleTimeout);**

**if(callback){\_idleTimeout = window.setTimeout(callback, timeOutSeconds);}**

**}**

**$(document).on('mousedown mousemove keypress ontouchstart touchmove click scroll',\_userActivity);**

**\_userActivity();**

**}**

**var expireSeconds = 60 \* 1000;*/\* ---- 1 min ---- \*/***

**onIdleDetect(expireSeconds, function(){**

**console.log("Session Timedout: ");**

**});**

**Calculate consumed percentage**

var total = 500;

var partial = 100

function consumedPercentage(oldNumber, newNumber) {

return ((newNumber \* 100) / oldNumber) + "%"

}

//Example:

consumedPercentage(total, partial)

//Result: 20%

**Calculate remaining percentage**

**var total = 500;**

**var partial = 100**

**function getRemainingPercentage(oldNumber, newNumber){**

**return ( ( (oldNumber - newNumber) / oldNumber) \* 100 ) + "%";**

**}**

**//Example:**

**getRemainingPercentage(total, partial)**

**//Result: 80%**

**Make data immutable like cannot modified by others or deep copy**

**// this useful when object are not having functions**

**function getImmutableData(data) {**

**if(!data) return null;**

**return JSON.parse(JSON.stringify(data))**

**}**

**//Example:**

**var data1 = ["0","1","2"];**

**var data2 = getImmutableData(data1);**

**//Deep copy Array of objects**

**function \_deepCopyInstance(d) {**

**if(Array.isArray(d)) {**

**return d.map(x=>{**

**if(typeof x !== 'object') return x;**

**let \_obj = {};**

**for(let k in x) {**

**\_obj[k] = x[k];**

**}**

**return \_obj**

**});**

**}else if(typeof d === 'object') {**

**let \_obj = {};**

**for(let k in d) {**

**\_obj[k] = d[k];**

**}**

**return \_obj**

**} else**

**return d;**

**};**

**var latest = \_deepCopyInstance (data);**

**Is value exist in array of objects**

**function \_isNotInArrayObject(d, k, v) {**

**if(!d || !k || !v) return false;**

**if(!Array.isArray(d) || typeof k !== "string" || typeof v !== "string") return false;**

**for( let j = 0; j < d.length; j++ ){**

**if(d[j][k] === v){**

**return false**

**}**

**}**

**return true;**

**};**

**\_isNotInArrayObject(yourdata, "keyname", yourvalue)**

**Debounce or live search**

**const \_debounce = (fn, delay) => {**

**let timeoutID;**

**return function(...args) {**

**if(timeoutID) {**

**clearTimeout(timeoutID);**

**}**

**timeoutID = setTimeout(()=>{**

**fn(...args)**

**}, delay)**

**}**

**};**

**onChange={utils.\_debounce(e =>{**

**// your logic**

**},450)}**

**// invoke below logic when ever you get data**

**time=1;**

**clearTime=null;**

**isExpire(1,function(){**

**console.log("Lost connect");**

**});**

**Below logic for to set a timer to indicate there is no data after some specified**

**let time=1;**

**let clearTime;**

**function isExpire(min, callback) {**

**clearTime = setTimeout(function(){**

**time++;**

**if(((time % 3600) / 60) > min){**

**if(callback) callback();**

**clearTimeout(clearTime);//Lost connection**

**}**

**else{**

**isExpire(min);// connection is good**

**}**

**},1000);**

**}**

**const onHandleSearch = (event) => {**

**let result = allData.filter((data) => data.title.toLowerCase().search(event.target.value.toLowerCase()) !== -1);**

**setFilteredData(result);**

**}**

**<input type="text" onChange={(event) =>handleSearch(event)} />**

**scroll**

**function handleScroll(event) {**

**var node = event.target;**

**const bottom = node.scrollHeight - node.scrollTop === node.clientHeight;**

**if (bottom) {**

**console.log('BOTTOM REACHED:', bottom);**

**}**

**}**

**const paneDidMount = (node) => {**

**if (node) {**

**node.addEventListener('scroll', handleScroll);**

**setReferenceNode(node);**

**}**

**};**

**ref={paneDidMount}**

**useEffect(() => {**

**return () => referenceNode.removeEventListener('scroll', handleScroll);**

**}, []);**

**onScroll={() => handleScroll()}**

**Find at least one match in data**

**var res=[{"name":"mano", "id":""},{"name": "kumar", "id":"cvbhghj7"}];**

**function findAtleaseOnematch(data,prop) {**

**if(!data && prop) return false;**

**var arr = [];**

**data.forEach(function(item){**

**if(item[prop]) arr.push(true);**

**else arr.push(false)**

**})**

**return ( arr.indexOf(true) !==-1 )**

**}**

**//Example:**

**findAtleaseOnematch(res,'id');**

**//Result: true;**

**Comparing two objects or json data equal or not**

**var res1 = [{"name": "mano","id": "sdfsdf"},{"name": "kumar","id": "cvbhghj7"}];**

**var res2 = [{"name": "mano","id": ""},{"name": "kumar","id": "cvbhghj7"}];**

**function isMatch(data1, data2) {**

**if(!data1 && !data2) {return false};**

**return ( JSON.stringify(data1) === JSON.stringify(data2) )**

**}**

**//Example:**

**isMatch(res1, res2);**

**//Result: false**

**Perticularly specific word is available or not in a string checking**

**user?.userPreferences?.role.match(/manager|admin/gi)**

**-> updated object key value in existing object**

**var obj = {isFilter: true, name:"Bhargav", designation:"Developer"};**

**var action = {key:"isFilter", value: false};**

**{...obj, [action.key]:action.value}**

**OutPut: {isFilter: false, name: "Bhargav", designation: "Developer"}**

**Find matching value in json object**

**var res1 = [{"name": "mano","id": "sdfsdf"},{"name": "kumar","id": "cvbhghj7"}];**

**function findValue(data,prop,value) {**

**if(typeof data == undefined || typeof prop == undefined || typeof value == undefined) return {};**

**var result = data.filter(function(item){return item[prop] == value});**

**return ( result.length ? result[0] : {} );**

**}**

**//Example:**

**findValue(res1,"name","kumar");**

**//Result: {name: "kumar", id: "cvbhghj7"};**

**Validation pattern types**

**//Email validation pattern**

**pattern="[a-zA-Z0-9.-\_]{1,}@[a-zA-Z.-]{2,}[.]{1}[a-zA-Z]{2,}"**

***//Accepts only number and text***

**pattern="^[A-Za-z0-9 ]+$"**

**//Accepts only number**

**pattern="\\d+"**

**//Accepts only text**

**pattern="[a-zA-Z]+"**

**Usage in html**

**<input type="text" name="input" pattern="^[A-Za-z0-9 ]+$" />**

**Change regular expression validation dynamically on dropdown change**

**dropItem;**

**inputModel;**

**isInvalid = true;**

**dropdown = [**

**{value:"text 1",label:"text 1",type:"number", regExp:"[^0-9]"},**

**{value:"text 2",label:"text 2",type:"string", regExp:"[^a-zA-Z ]"},**

**{value:"text 3",label:"text 3",type:"email", regExp:""}**

**];**

**onDropChange(item){**

***//this.dropItem=item;***

**this.inputModel="";**

**this.isInvalid=false;**

**}**

**onInputEnter(event){**

**if(this.dropItem['type'] !== "email"){**

**let regExp = new RegExp(this.dropItem['regExp'],'gi');**

**event.target.value = event.target.value.replace(regExp, '');**

**this.isInvalid=false;**

**}else{**

**let pattern=/[a-zA-Z0-9.-\_]{1,}@[a-zA-Z.-]{2,}[.]{1}[a-zA-Z]{2,}/**

**if(!pattern.test(event.target.value)){**

**this.isInvalid=true;**

**}else{**

**this.isInvalid=false;**

**}**

**}**

**}**

**//Usage in html**

**<select [(ngModel)]="dropItem" (change)="onDropChange(dropItem)">**

**<option \*ngFor="let item of dropdown" [ngValue]="item">{{item.label}}</option>**

**</select>**

**<input type="text" (input)="onInputEnter($event)" [disabled]="!dropItem" [(ngModel)]="inputModel" />**

**<div \*ngIf="inputModel && isInvalid">Invalid email</div>**

**URL validation**

**var reguE = /(https|http|www\.)/**

**Results: validation accepts only below url formats**

**reguE.test("https://stackoverflow.com")**

**reguE.test("http://stackoverflow.com")**

**reguE.test("www.stackoverflow.com")**

**remove empty space first and last**

**function trimSpace(val) {**

**return ( val ? val.replace(/^\s+|\s+$/g,'') : "" )**

**}**

**Price format or adding commas to input number on change**

**function addCommas(x) {**

**let commasaparate = x.toString().split(".");**

**commasaparate[0] = commasaparate[0].replace(/\B(?=(\d{3})+(?!\d))/g, ",");**

**return commasaparate.join(".");**

**}**

**function onchnage(event){**

**console.log(value);**

**event.target.value = "$"+ addCommas(event.target.value.replace(/[^0-9.]/g,''))**

**}**

**Usage in HTML**

**<input type="text" name="name" onchange="onchnage(event)">**

**Prevent first space on text field**

**function preventFirstSpaceOnKeyPress(e){**

**var key = e.keyCode;**

**if(!e.target.value && key === 32){**

**e.preventDefault();**

**}**

**}**

**Prevent special characters on keypress**

***/\* prevent special characters on enter in textarea\*/***

**onKeypressChange(e){**

**let regE = /[~!@#$%\^&\*()+=\-\[\]\\';/{}|\\":<>\?]/;**

**if(regE.test(e.key) || e.key== '`' ){**

**e.preventDefault();**

**}**

**}**

**Usage in html**

**<input type="text" name="input" onkeypress="onKeypressChange(event)" />**

**base64 pdf to blob pdf**

**function converterPDFBase64ParaBlob(base64) {**

**var base64ToArrayBuffer = function(e) {**

**for (var t = window.atob(e), n = t.length, r = new Uint8Array(n), o = 0; o < n; o++) {**

**var i = t.charCodeAt(o);**

**r[o] = i**

**}**

**return r**

**};**

**function base64pdfUrlConverter(){**

**var e = base64ToArrayBuffer(base64)**

**, t = new Blob([e],{**

**type: "application/pdf"**

**});**

**if (!window.navigator || !window.navigator.msSaveOrOpenBlob)**

**return window.URL.createObjectURL(t);**

**window.navigator.msSaveOrOpenBlob(t)**

**}**

**return base64pdfUrlConverter();**

**}**

**converterPDFBase64ParaBlob(base64str)**

Deet copy array object, find is value exist or not in array of object?

export const \_deepCopyArrayObject = (d) => {

  if (!d) return d;

  else if (Array.isArray(d)) {

    return d.map(x => {

      if (typeof x !== 'object') return x;

      let \_obj = {};

      for (let k in x) {

        \_obj[k] = x[k];

      }

      return \_obj

    });

  } else if (typeof d === 'object') {

    let \_obj = {};

    for (let k in d) {

      \_obj[k] = d[k];

    }

    return \_obj

  } else

    return d;

};

export const \_isValueInArrayObject = (d, k, v) => {

  for (let j = 0; j < d.length; j++) {

    if (d[j][k] === v) {

      return false

    }

  }

  return true;

};

export const \_debounce = (fn, delay) => {

  let timeoutID;

  return function (...args) {

    if (timeoutID) {

      clearTimeout(timeoutID);

    }

    timeoutID = setTimeout(() => {

      fn(...args)

    }, delay)

  }

};

**Angular 2+**

<https://atom.io/packages/angular-2-typescript-snippets>

**Creating model for data types using class (Both class and interface for type checking)**

// Business.ts

export default class Business {

firstName?: String;

lastName?: String;

id?: Number;

}

**Creating model using interface**

**// Business.ts**

**export interface Business {**

**firstName?: String;**

**lastName?: String;**

**id?: Number;**

**}**

**Date formats using DatePipe**

**import { DatePipe } from '@angular/common';**

**var datePipe = new DatePipe("en-US");**

**var value = datePipe.transform(dateValue, 'MMM-dd-yyyy');**

**Creating custom date pipes for Global date format**

**Constants.ts file**

export class Constants {

static readonly DATE\_FMT = 'dd/MMM/yyyy';

static readonly DATE\_TIME\_FMT = `${Constants.DATE\_FMT} hh:mm:ss`;

}

**import { Constants } from './constants';**

**import { Pipe, PipeTransform } from '@angular/core';**

**import { DatePipe } from '@angular/common';**

**@Pipe({**

**name: 'dateFormat'**

**})**

**export class DateFormatPipe extends DatePipe implements PipeTransform {**

**transform(value: any, args?: any): any {**

**return super.transform(value, Constants.DATE\_FMT);**

**}**

**}**

**Usage in HTML files**

**<div>{{ today | dateFormat}}</div>**

**Post method in anagular**

**function sendDocuments(data){**

**return this.http.post<any[]>(this.url+'/documents/',formData);**

**}**

**Resizing element or set dynamic height based on window**

viewSize;

ngOnInit(){

this.viewSize = (window.innerHeight - 200);

}

onResize(){

this.viewSize = (window.innerHeight - 200)

}

**//Usage in html**

**<div class="scrollbox" (window:resize)="onResize()">**

**<embed src="https://gahp.net/wp-content/uploads/2017/09/sample.pdf" [style.height.px]="viewSize">**

**</div>**

**Show loader until resource url loaded or iframe context load**

loading=true;

dismissLoading(){

this.loading =false;

}

Usage in html

<iframe src="https://sample.pdf" (load)="dismissLoading()"></iframe>

**Textarea characters remaining count and max length restrictions**

**remainChars(value, num){**

**return (value ? (num - value.length) : 20)**

**}**

**//Usage in html**

**<textarea maxLength="20" [(ngModel)]="textareaModel"></textarea>**

**<div>{{remainChars(textareaModel,20)}}</div>**

**Template driven validation**

**<h3>Template-Driven Form Validation</h3>**

**<div class="card">**

**<div class="card-body">**

**<form name="form" (ngSubmit)="f.form.valid && onSubmit()" #f="ngForm" [mustMatch]="['password', 'confirmPassword']" novalidate>**

**<div class="form-row">**

**<div class="form-group">**

**<label>Title</label>**

**<select name="title" class="form-control" [(ngModel)]="model.title" #title="ngModel" [ngClass]="{ 'is-invalid': f.submitted && title.invalid }" required>**

**<option value="">Please select</option>**

**<option value="Mr">Mr</option>**

**<option value="Mrs">Mrs</option>**

**</select>**

**<div \*ngIf="f.submitted && title.invalid" class="invalid-feedback">**

**<div \*ngIf="title.errors.required">Title is required</div>**

**</div>**

**</div>**

**<div class="form-group">**

**<label>First Name</label>**

**<input type="text" name="firstName" class="form-control" [(ngModel)]="model.firstName" #firstName="ngModel" [ngClass]="{ 'is-invalid': f.submitted && firstName.invalid }" required>**

**<div \*ngIf="f.submitted && firstName.invalid" class="invalid-feedback">**

**<div \*ngIf="firstName.errors.required">First Name is required</div>**

**</div>**

**</div>**

**<div class="form-group">**

**<label>Email</label>**

**<input type="text" name="email" class="form-control" [(ngModel)]="model.email" #email="ngModel" pattern="[a-zA-Z0-9.-\_]{1,}@[a-zA-Z.-]{2,}[.]{1}[a-zA-Z]{2,}" [ngClass]="{ 'is-invalid': f.submitted && email.invalid }" required email>**

**<div \*ngIf="f.submitted && email.invalid" class="invalid-feedback">**

**<div \*ngIf="email.errors.required">Email is required</div>**

**<div \*ngIf="email.errors.pattern">Email must be a valid email address</div>**

***<!-- <div \*ngIf="email.errors.email">Email must be a valid email address</div> -->***

**</div>**

**</div>**

**<div class="form-group">**

**<label>Password</label>**

**<input type="password" name="password" class="form-control" [(ngModel)]="model.password" #password="ngModel" [ngClass]="{ 'is-invalid': f.submitted && password.invalid }" required minlength="6">**

**<div \*ngIf="f.submitted && password.invalid" class="invalid-feedback">**

**<div \*ngIf="password.errors.required">Password is required</div>**

**<div \*ngIf="password.errors.minlength">Password must be at least 6 characters</div>**

**</div>**

**</div>**

**<div class="form-group">**

**<label>Confirm Password</label>**

**<input type="password" name="confirmPassword" class="form-control" [(ngModel)]="model.confirmPassword" #confirmPassword="ngModel" [ngClass]="{ 'is-invalid': f.submitted && confirmPassword.invalid }" required>**

**<div \*ngIf="f.submitted && confirmPassword.invalid" class="invalid-feedback">**

**<div \*ngIf="confirmPassword.errors.required">Confirm Password is required</div>**

**<div \*ngIf="confirmPassword.errors.mustMatch">Passwords must match</div>**

**</div>**

**</div>**

**<div class="form-group form-check">**

**<input type="checkbox" name="acceptTerms" id="acceptTerms" class="form-check-input" [(ngModel)]="model.acceptTerms" #acceptTerms="ngModel" [ngClass]="{ 'is-invalid': f.submitted && acceptTerms.invalid }" required>**

**<label for="acceptTerms" class="form-check-label">Accept Terms & Conditions</label>**

**<div \*ngIf="f.submitted && acceptTerms.invalid" class="invalid-feedback">Accept Ts & Cs is required</div>**

**</div>**

**</div>**

**<div class="text-center">**

**<button class="btn btn-primary mr-1">Register</button>**

**<button class="btn btn-secondary" type="reset">Cancel</button>**

**</div>**

**</form>**

**</div>**

**</div>**

**Confirm Password must match directive ==> must-match.directive file**

**import { Directive, Input } from '@angular/core';**

**import { NG\_VALIDATORS, Validator, ValidationErrors, FormGroup } from '@angular/forms';**

**export function MustMatch(controlName: string, matchingControlName: string) {**

**return (formGroup: FormGroup) => {**

**const control = formGroup.controls[controlName];**

**const matchingControl = formGroup.controls[matchingControlName];**

***// return null if controls haven't initialised yet***

**if (!control || !matchingControl) {**

**return null;**

**}**

***// return null if another validator has already found an error on the matchingControl***

**if (matchingControl.errors && !matchingControl.errors.mustMatch) {**

**return null;**

**}**

***// set error on matchingControl if validation fails***

**if (control.value !== matchingControl.value) {**

**matchingControl.setErrors({ mustMatch: true });**

**} else {**

**matchingControl.setErrors(null);**

**}**

**}**

**}**

**@Directive({**

**selector: '[mustMatch]',**

**providers: [{ provide: NG\_VALIDATORS, useExisting: MustMatchDirective, multi: true }]**

**})**

**export class MustMatchDirective implements Validator {**

**@Input('mustMatch') mustMatch: string[] = [];**

**validate(formGroup: FormGroup): ValidationErrors {**

**return MustMatch(this.mustMatch[0], this.mustMatch[1])(formGroup);**

**}**

**}**

**//In app.madule file import directive with respective path and inject in declarations like below**

import { MustMatchDirective } from './must-match.directive';

declarations: [

**MustMatchDirective**

]

**Reactive form validation**

**<form [formGroup]="registerForm" (ngSubmit)="onSubmit()">**

**<div class="form-row">**

**<div class="form-group col">**

**<label>Title</label>**

**<select formControlName="title" class="form-control" [ngClass]="{ 'is-invalid': submitted && f.title.errors }">**

**<option value=""></option>**

**<option value="Mr">Mr</option>**

**<option value="Mrs">Mrs</option>**

**</select>**

**<div \*ngIf="submitted && f.title.errors" class="invalid-feedback">**

**<div \*ngIf="f.title.errors.required">Title is required</div>**

**</div>**

**</div>**

**<div class="form-group col-5">**

**<label>First Name</label>**

**<input type="text" formControlName="firstName" class="form-control" [ngClass]="{ 'is-invalid': submitted && f.firstName.errors }" />**

**<div \*ngIf="submitted && f.firstName.errors" class="invalid-feedback">**

**<div \*ngIf="f.firstName.errors.required">First Name is required</div>**

**</div>**

**</div>**

**<div class="form-group">**

**<label>Last Name</label>**

**<input type="text" formControlName="lastName" class="form-control" [ngClass]="{ 'is-invalid': submitted && f.lastName.errors }" />**

**<div \*ngIf="submitted && f.lastName.errors" class="invalid-feedback">**

**<div \*ngIf="f.lastName.errors.required">Last Name is required</div>**

**</div>**

**</div>**

**<div class="form-group col">**

**<label>Password</label>**

**<input type="password" formControlName="password" class="form-control" [ngClass]="{ 'is-invalid': submitted && f.password.errors }" />**

**<div \*ngIf="submitted && f.password.errors" class="invalid-feedback">**

**<div \*ngIf="f.password.errors.required">Password is required</div>**

**<div \*ngIf="f.password.errors.minlength">Password must be at least 6 characters</div>**

**</div>**

**</div>**

**<div class="form-group">**

**<label>Confirm Password</label>**

**<input type="password" formControlName="confirmPassword" class="form-control" [ngClass]="{ 'is-invalid': submitted && f.confirmPassword.errors }" />**

**<div \*ngIf="submitted && f.confirmPassword.errors" class="invalid-feedback">**

**<div \*ngIf="f.confirmPassword.errors.required">Confirm Password is required</div>**

**<div \*ngIf="f.confirmPassword.errors.mustMatch">Passwords must match</div>**

**</div>**

**</div>**

**<div class="form-group">**

**<label>Email</label>**

**<input type="text" formControlName="email" class="form-control" [ngClass]="{ 'is-invalid': submitted && f.email.errors }" />**

**<div \*ngIf="submitted && f.email.errors" class="invalid-feedback">**

**<div \*ngIf="f.email.errors.required">Email is required</div>**

**<div \*ngIf="f.email.errors.pattern">Email must be a valid email address</div>**

**</div>**

**</div>**

**<div class="form-group form-check">**

**<input type="checkbox" formControlName="acceptTerms" id="acceptTerms" class="form-check-input" [ngClass]="{ 'is-invalid': submitted && f.acceptTerms.errors }" />**

**<label for="acceptTerms" class="form-check-label">Accept Terms & Conditions</label>**

**<div \*ngIf="submitted && f.acceptTerms.errors" class="invalid-feedback">Accept Ts & Cs is required</div>**

**</div>**

**<div class="text-center">**

**<button class="btn btn-primary mr-1">Register</button>**

**<button class="btn btn-secondary" type="reset" (click)="onReset()">Cancel</button>**

**</div>**

**</div>**

**</form>**

**//Usage in .ts files**

**import { Component, OnInit } from '@angular/core';**

**import { FormBuilder, FormGroup, Validators } from '@angular/forms';**

***// import custom validator to validate that password and confirm password fields match***

**import { MustMatch } from './must-match.validator';**

**@Component({ selector: 'app', templateUrl: 'app.component.html' })**

**export class AppComponent implements OnInit {**

**registerForm: FormGroup;**

**submitted = false;**

**emailRegEx="[a-zA-Z0-9.-\_]{1,}@[a-zA-Z.-]{2,}[.]{1}[a-zA-Z]{2,}";**

**constructor(private formBuilder: FormBuilder) { }**

**ngOnInit() {**

**this.registerForm = this.formBuilder.group({**

**title: ['', Validators.required],**

**firstName: ['', Validators.required],**

**lastName: ['', Validators.required],**

**email: ['', [Validators.required, , Validators.pattern(this.emailRegEx)]],**

**password: ['', [Validators.required, Validators.minLength(6)]],**

**confirmPassword: ['', Validators.required],**

**acceptTerms: [false, Validators.requiredTrue]**

**}, {**

**validator: MustMatch('password', 'confirmPassword')**

**});**

**}**

***// convenience getter for easy access to form fields***

**get f() { return this.registerForm.controls; }**

**onSubmit() {**

**this.submitted = true;**

***// stop here if form is invalid***

**if (this.registerForm.invalid) {**

**return;**

**}**

**}**

**onReset() {**

**this.submitted = false;**

**this.registerForm.reset();**

**}**

**}**

**Multiple API calls at a time, even any one of API fails it will give proper execution with order**

**import { Component, OnInit } from '@angular/core';**

**import { Observable, forkJoin, of } from 'rxjs';**

**import { catchError } from 'rxjs/operators';**

**import { HttpClient } from '@angular/common/http';**

**@Component({**

**selector: 'yourcomponentname',**

**template: `yourtemplatename`,**

**})**

**export class HelloComponent implements OnInit {**

**urls = ['https://jsonplaceholder.typicode.com/comments/1', 'https://jsonplaceholder.typicode.com/comments/1s', 'https://jsonplaceholder.typicode.com/comments/3'];**

**constructor(private http: HttpClient) { }**

**ngOnInit() {**

**this.requestDataFromMultipleSources().subscribe(re => {**

**console.log("data: ",re);**

**});**

**}**

**requestDataFromMultipleSources(): Observable<any[]> {**

**let response1 = this.http.get("https://jsonplaceholder.typicode.com/comments/1").pipe(**

**catchError(err => of(err.status)),**

**);**

**let response2 = this.http.get("https://jsonplaceholder.typicode.com/comments/1s").pipe(**

**catchError(err => of({"error":err.status})),**

**);**

**let response3 = this.http.get("https://jsonplaceholder.typicode.com/comments/3").pipe(**

**catchError(err => of(err.status)),**

**);**

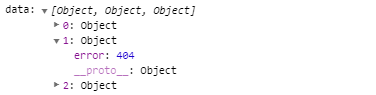
***// Observable.forkJoin (RxJS 5) changes to just forkJoin() in RxJS 6***

**return forkJoin([response1, response2, response3]);**

**}**

**}**

**Result:**

****

**Email field filtering new and updated fields**

**emailPattern = "[a-zA-Z0-9.-\_]{1,}@[a-zA-Z.-]{2,}[.]{1}[a-zA-Z]{2,}";**

**serverResponse={**

**"travelerList":[**

**{**

**"Person":{**

**"EmailAddress": "HIMAVANTH.414@GMAIL.COM"**

**}**

**},**

**{**

**"Person":{**

**"EmailAddress": ""**

**}**

**}**

**]**

**};**

**travelersInfo = [];**

**travelersInfoCopy=[];**

**ngOnInit(){**

***//this.numberOnly = "[a-zA-Z]+";***

**this.serverResponse.travelerList.forEach(item=>{**

**this.travelersInfo.push(item.Person);**

**});**

**this.travelersInfo.forEach((item,index)=>{**

**if(item["EmailAddress"]){**

**item["id"]=("emailId\_"+(index+1))**

**}**

**});**

***/\* holding original data for tracking changes \*/***

**this.travelersInfoCopy = this.getImmutableData(this.travelersInfo);**

**}**

***/\* It converts data into immutable that cannot modfied by others \*/***

**getImmutableData(data){**

**return data ? (JSON.parse(JSON.stringify(data))) : data;**

**}**

***/\*It compares 2object or 2 arrays or any json data whether equal or not if equal then returns true other wise false \*/***

**isMatch(data1,data2){**

**return (JSON.stringify(data1) == JSON.stringify(data2))**

**}**

**onChangeInput(listItem) {**

**this.travelersInfoCopy.forEach(item=>{**

**if(listItem['id']){**

**if(item["id"] == listItem['id']){**

**if(listItem['EmailAddress'].toLowerCase() !== item["EmailAddress"].toLowerCase()){**

**listItem["indicator"]=listItem['EmailAddress'] ? "U" : "";**

**}else{**

**listItem["indicator"]="";**

**}**

**}**

**}**

**});**

**}**

**saveEmails(){**

**let resultDat = this.travelersInfo.filter(x=>x["indicator"] == "U" || !x['id']);**

**console.log("resultDat: ",resultDat);**

**}**

**Data sharing between components (sibling or unrelated or parent child)**

By using new BehaviorSubject () shares data between components. For reusable, create a service (to manage methods globally).

Note: ({ providedIn: 'root' } once you set providedIn prop in Injectable then it will available entire application(globally), so no need to import this service in app.module.

yourService.ts

import { Injectable } from '@angular/core';

import { Observable, BehaviorSubject } from 'rxjs';

@Injectable({ providedIn: 'root' })

export class AuthService {

private actionPayload = new BehaviorSubject<any>("");

// when assign a data to payloadDataInstance then it will maintain through out application.

// (it's just a variable no subscribtion, you can use this is also for ref. With out behaviorsuject it works.)

payloadGlobalStore:any;

setPayload(data: any) {

this.actionPayload.next(data)

}

clearPayload() {

this.actionPayload.next("");

}

getPayload(): Observable<any> {

return this.actionPayload.asObservable();

}

}

Set data in you component like below(sending data to respective page).

exampleComponent1.ts (example name is homecomponent)

import { Component, OnInit } from '@angular/core';

import { AuthService } from '.././auth.service';

import {Router} from '@angular/router';

@Component({

selector: 'app-home',

templateUrl: './home.component.html'

})

export class HomeComponent implements OnInit {

records=[

{"firstName":"Suman","lastName":"Aakula","startDate":"2019-04-09T10:16:42.299Z"},

{"firstName":"Kumar","lastName":"Dictum","startDate":"2019-02-10T10:16:42.299Z"}

];

constructor(private authService:AuthService, private router: Router) {}

ngOnInit() {

//console.log(this.authService.payloadGlobalStore)

}

goToUsers(){

this.authService.payloadGlobalStore =[…this.records];

this.authService.setPayload(this.records);

this.router.navigate(['/users']);

}

}

ExampleComponent2.ts (in this component you will receive data )

import { Component, OnInit, OnDestroy } from '@angular/core';

import { AuthService } from '.././auth.service';

import { Subscription } from 'rxjs';

@Component({

selector: 'app-users',

templateUrl: './users.component.html'

})

export class UsersComponent implements OnInit, OnDestroy {

records:any;

subscription: Subscription;

constructor(private authService:AuthService) {

   this.subscription=authService.getPayload().subscribe((res)=>{

this.records=res ? res : [];

console.log(res);

});

//console.log("Users page with out subscription: ", this.authService.payloadGlobalStore);

}

ngOnInit() {}

ngOnDestroy(){

// must unsubscribe to ensure no memory leaks. When page leave then it will execute.

this.subscription.unsubscribe();

}

}

**Data sharing from Parent to Child using @Input() decorator**

@Input() <peroperty>: <type>;

**Data sharing from child to parent using @Output() decorator**

@Output() <event>: EventEmitter<type> = new EventEmitter();

**Restful API calls (GET/POST/UPDATE/DELETE)**

First import HttpClientModule in app.module.ts file

Step1: App.module.ts

import { HttpClientModule } from '@angular/common/http';

@NgModule({

imports: [HttpClientModule]

})

export class AppModule { }

Step2: import HttpClient your file

youComponent.ts or yourService.ts

import { HttpClient } from '@angular/common/http';

export class yourComponent {

constructor(private http: HttpClient) { }

getMethod {

this.http.get<User[]>(`${config.apiUrl}/users`).subscribe((result)=>{code here});

}

postMethod {

this.http.post<any>(`${config.apiUrl}/users/authenticate`, data).subscribe((result)=>{code here});

}

updateMethod {

this.http.put<User[]>(`${config.apiUrl}/users/${id}`,data).subscribe((result)=>{code here});

}

deleteMethod {

this.http.delete<User[]>(`${config.apiUrl}/users/${id}`).subscribe((result)=>{code here});

}

}

**Detecting browser back and forward click**

**import { PlatformLocation } from '@angular/common'**

**constructor(private location: PlatformLocation){**

**location.onPopState((e) => {**

**console.log('your code here');**

**});**

***/\****

***window.history.pushState(null, "", window.location.href);***

***window.onpopstate = function() {***

***console.log(window.location.href);***

***window.history.pushState(null, "", window.location.href);***

***};***

***\*/***

**}**

**Routing in Angular**

Configure (add) the routing of angular components inside an app.module.ts or separate routing.module.ts file like below

app.module.ts

**import { NgModule } from '@angular/core';**

**import { HomeComponent } from './home.component';**

**import { PageNotFoundComponent } from './notfound.component';**

**import { RouterModule, Routes } from '@angular/router';**

**const routes: Routes = [**

**{ path: '', redirectTo: '/home', pathMatch: 'full' },**

**{ path: 'home', component: HomeComponent },**

**{path:'not-found',component:PageNotFoundComponent},**

**{path:'\*\*', pathMatch:'full', redirectTo:'/not-found' }**

**];**

**@NgModule({**

**imports: [RouterModule.forRoot(routes)],**

**exports: [RouterModule],**

**})**

**export class AppRoutingModule {}**

**Usage: in app.component.html add below tag**

**<router-outlet></router-outlet>**

**<a style="padding:10px;" routerLink="/"**

**routerLinkActive="active">Home</a>**

**<a style="padding:10px;" routerLink="/users"**

**routerLinkActive="active">Users</a>**

routing.module.ts and here one more concept implementing resolve (After dynamic(api call) data loaded then this will be route. )

**import { NgModule } from '@angular/core';**

**import { HttpClientModule } from '@angular/common/http';**

**import { RouterModule, Routes } from '@angular/router';**

**import { AppComponent } from './app.component';**

**import { HomeComponent } from './home.component';**

**import { NewsComponent } from './news.component';**

**import { NewsResolver } from './news-resolver.service';**

**const routes: Routes = [**

**{ path: '', redirectTo: '/home', pathMatch: 'full' },**

**{ path: 'home', component: HomeComponent },**

**{ path: 'news/:id',**

**component: NewsComponent,**

**resolve: {**

**news: NewsResolver**

**}**

**}**

**];**

**@NgModule({**

**imports: [RouterModule.forRoot(routes), HttpClientModule],**

**exports: [RouterModule]**

**})**

**export class AppRoutingModule {}**

service.ts (this file created for resolve -> It performs route data retrieval before route activation.)

**import { HttpClient } from '@angular/common/http';**

**import { Injectable } from '@angular/core';**

**import { Resolve } from '@angular/router';**

**import { Observable } from 'rxjs';**

**import { catchError, map } from 'rxjs/operators';**

**@Injectable({providedIn:'root'})**

**export class NewsResolver implements Resolve<any> {**

**constructor(private http: HttpClient) { }**

**resolve(): Observable<any> {**

**let newsUrl = 'https://httpbin.org/post';**

**let news = 'The sky is blue'; *//Mock data to be returned by test API***

**return this.http.post(newsUrl, news).pipe(**

**map( (dataFromApi) => dataFromApi ),**

**catchError( (err) => Observable.throw(err.json().error) )**

**)**

**}**

**}**

**Detecting Routing change or navigation changing or page changing**

Now, write the following code inside the  **app.component.ts**file.

***// app.component.ts***

**import { Component } from '@angular/core';**

**import { NavigationCancel,**

**Event,**

**NavigationEnd,**

**NavigationError,**

**NavigationStart,**

**Router } from '@angular/router';**

**@Component({**

**selector: 'app-root',**

**templateUrl: './app.component.html'**

**})**

**export class AppComponent {**

**constructor(private \_router: Router) {**

**this.\_router.events.subscribe((event: Event) => {**

**this.navigationInterceptor(event);**

**});**

**}**

**private navigationInterceptor(event: Event): void {**

**if (event instanceof NavigationStart) {**

**}**

**if (event instanceof NavigationEnd) {**

**}**

**if (event instanceof NavigationCancel) {**

**}**

**if (event instanceof NavigationError) {**

**}**

**}**

**}**

**Lazy loading in Angular implementation.**

Lazy loading is a technique in Angular that allows you to load JavaScript components asynchronously when a specific route activated. With lazy loading our application does not need to load everything at once, we can improve initial loading time of an application by loading the components on demand. Implement like below.

App.module.ts (Provide below format in app module) (you can set your component name your own)

**import { NgModule } from '@angular/core';**

**import { RouterModule, Routes } from '@angular/router';**

***// authgaurd is optional***

**const appRoutes: Routes = [**

**{path: 'dashboard', loadChildren: "./dashboard/dashboard.module#DashboardModule" }**

**];**

**@NgModule({**

**declarations: [YourComponent],**

**imports: [RouterModule.forRoot(appRoutes, {useHash: true})],**

**})**

**export class AppModule { }**

dashboard.ts

import { Component, OnInit } from '@angular/core';

@Component({

selector: 'app-dashboard',

templateUrl: './dashboard.component.html',

styleUrls: ['./dashboard.component.css']

})

export class DashboardComponent {}

dashoboard.routes.module.ts

import { NgModule } from '@angular/core';

import { Routes, RouterModule } from '@angular/router';

import { DashboardComponent } from './dashboard.component';

const routes: Routes = [

{path: '', component: DashboardComponent}

];

@NgModule({

imports: [RouterModule.forChild(routes)],

exports: [RouterModule]

})

export class dadhboardRoutingModule { }

dashboard.module.ts

import { NgModule } from "@angular/core";

import { CommonModule } from "@angular/common";

import { dadhboardRoutingModule } from './rote.module';

import { DashboardComponent } from './dashboard.component';

@NgModule({

imports: [CommonModule,dadhboardRoutingModule],

declarations: [DashboardComponent]

})

export class DashboardModule {}

**Providing Access for components using authgaurd**

import { Injectable } from '@angular/core';

import { Router, CanActivate, ActivatedRouteSnapshot, RouterStateSnapshot } from '@angular/router';

@Injectable({ providedIn: 'root' })

export class AuthGuard implements CanActivate {

constructor(private router: Router) { }

canActivate(route: ActivatedRouteSnapshot, state: RouterStateSnapshot) {

return checkPermission(state.url);

}

checkPermission(url){

if (localStorage.getItem('currentUser')) {

// logged in so return true

return true;

}

// not logged in so redirect to login page with the return url

this.router.navigate(['/login']);

return false;

}

}

import { Injectable } from '@angular/core';

import { CanActivate, CanActivateChild, Router, ActivatedRouteSnapshot, RouterStateSnapshot } from '@angular/router';

import { AuthService } from './auth.service';

@Injectable({providedIn: 'root'})

export class AuthGuarder implements CanActivate, CanActivateChild{

constructor(private authService: AuthService, private router: Router) { }

canActivate(route: ActivatedRouteSnapshot, state: RouterStateSnapshot): boolean {

let url: string = state.url;

return this.checkPermission(url);

}

canActivateChild(route: ActivatedRouteSnapshot, state: RouterStateSnapshot): boolean {

return this.canActivate(route, state);

}

checkPermission(url: string): boolean {

if (this.authService.isLoggedIn) {

return true;

}

// Navigate to the login page with extras

this.router.navigate(['/login']);

return false;

}

}

//Implementaion in routes

const myRoots: Routes = [

{ path: '', component: HomeComponent, pathMatch: 'full'

{ path: 'register', component: RegistrationComponent },

{ path: 'login', component: LoginComponent},

{ path: 'home', component: HomeComponent, canActivate: [AuthGuard]}

];

Note: this.authService.isLoggedIn is a Boolean and default is false, when login that time will make true (will update in authService file)

**Child routes in angular 6**

{

path: 'admin',

component: AdminPageComponent,

children: [

{

path: 'users',

component: AdminUserComponent

}

]

}

**Location strategies (hash routing)**

**imports: [**

**RouterModule.forRoot(appRoutes, {useHash: true})**

**]**

***// output URL: example.com/#/page***

**Multiple api calls with forkjoin.**

**import { Observable, forkJoin, of,from } from 'rxjs';**

**import { catchError } from 'rxjs/operators';**

**import { concatMap,concat } from 'rxjs/operators';**

**urls = ['https://jsonplaceholder.typicode.com/comments/1', 'https://jsonplaceholder.typicode.com/comments/1s', 'https://jsonplaceholder.typicode.com/comments/3'];**

**listofOrder() {**

**return from(this.urls).pipe(**

**concatMap(url => {**

**return this.http.get(url).pipe(**

**catchError(err => of(err.status))**

**);**

**}));**

**}**

**ngOnInit() {**

**this.listofOrder().subscribe(re => {**

***// console.log("H: ");***

**console.log("can: ",re);**

**});**

**this.requestDataFromMultipleSources().subscribe(re => {**

**console.log("data: ",re);**

**});**

**}**

**requestDataFromMultipleSources(): Observable<any[]> {**

**let response1 = this.http.get("https://jsonplaceholder.typicode.com/comments/1").pipe(**

**catchError(err => of(err.status)),**

**);**

**let response2 = this.http.get("https://jsonplaceholder.typicode.com/comments/1s").pipe(**

**catchError(err => of({"error":err.status})),**

**);**

**let response3 = this.http.get("https://jsonplaceholder.typicode.com/comments/3").pipe(**

**catchError(err => of(err.status)),**

**);**

***// Observable.forkJoin (RxJS 5) changes to just forkJoin() in RxJS 6***

**return forkJoin([response1, response2, response3]);**

**}**

**Sorting and search filter combination**

**DBDATA=[{"firstName":"Leilani","id":"1550","startDate":"2015/05/27"},{"firstName":"Jemima","id":"7582","startDate":"2015/05/21"},{"firstName":"Hiroko","id":"9368","startDate":"2015/03/13"},{"firstName":"Nathaniel","id":"8331","startDate":"2014/12/05"},{"firstName":"Silas","id":"0746","startDate":"2014/11/13"},{"firstName":"Jermaine","id":"1545","startDate":"2015/03/06"},{"firstName":"Bhargav","id":"4395","startDate":"2015/05/22"},{"firstName":"Rama","id":"2973","startDate":"2014/12/01"}];**

**public records:Array<any> = [];**

**public itemsPerPage:number = 5;**

**public totalItems:number = 0;**

**public columns:Array<any> = [**

**{headerName: 'firstName', title:"FirstName", sort:true, sorting:false },**

**{headerName: 'id', title:"JurnyNumber", sort: true, sorting:false},**

**{headerName: 'startDate', title:"Start Date", sort: false, sorting:false}**

**];**

**public filterBy:any = {**

**paging: true,**

**searchText: "",**

**sortName:"",**

**sorting:false,**

**checkedValues: [],**

**searchColumns:{firstName:"", id:""}**

**};**

**public constructor() {**

**this.totalItems = this.DBDATA.length;**

**}**

**public ngOnInit() {**

**this.onChangeTable();**

**}**

**addtableRows(data:Array<any> = this.DBDATA):Array<any> {**

**let start = 0;**

**return data.slice(start, this.itemsPerPage);**

**}**

**changeSort(data:any):any {**

**const { sortName, sorting } = this.filterBy;**

**if (!sortName) {**

**return data;**

**}**

**const columns = this.columns || [];**

***/\* sorting by assceding and descending \*/***

**return data.sort((previous:any, current:any) => {**

**if (previous[sortName] > current[sortName]) {**

**return !sorting ? -1 : 1;**

**} else if (previous[sortName] < current[sortName]) {**

**return sorting ? -1 : 1;**

**}**

**return 0;**

**});**

**}**

**onSearchFilter(data:any):any {**

**const { searchText } = this.filterBy;**

**let filteredData:Array<any> = [...data];**

**if (!searchText) {**

**return filteredData;**

**}**

***/\* iterate keys and assign searchtext to key \*/***

**for (let key in this.filterBy.searchColumns) {**

**this.filterBy.searchColumns[key] = searchText**

**}**

**filteredData = data.filter(item => {**

**return Object.keys(this.filterBy.searchColumns).some((keyName) => {**

**return new RegExp(this.filterBy.searchColumns[keyName], 'gi').test(item[keyName]) || this.filterBy.searchColumns[keyName] == "";**

**});**

**})**

**return filteredData;**

**};**

**onChangeTable():any {**

**let filteredData = this.onSearchFilter(this.DBDATA);**

**let sortedData = this.changeSort(filteredData);**

**this.records = this.filterBy.paging ? this.addtableRows(sortedData) : sortedData;**

**this.totalItems = sortedData.length;**

**}**

***/\* thir method will fire when click on sortable header column \*/***

**handleSort(headerColumn) {**

**const { headerName } = headerColumn;**

**this.filterBy.sortName=headerColumn.headerName;**

**for (let i = 0; i < this.columns.length; i++) {**

**if (this.columns[i].headerName === headerName) {**

**headerColumn.sorting = !headerColumn.sorting;**

**this.filterBy.sorting = headerColumn.sorting;**

**} else {**

**this.columns[i].sorting = false;**

**}**

**}**

**console.log("HI: ",this.filterBy.sorting);**

**this.onChangeTable();**

**}**

**Usage in html**

**<input type="text" (keyup)="onChangeTable()" [(ngModel)]="filterBy.searchText" placeholder="Serach any fields"/>**

**<table border="1" width="100%">**

**<tr>**

**<th \*ngFor="let column of columns" (click)="column.sort ? handleSort(column) : null">**

**<span>{{column.title}}</span>**

**<span \*ngIf="column.sort" class="sortIcon" [ngClass]="{'asc':column.sorting}"></span>**

**</th>**

**</tr>**

**<tr \*ngFor="let customer of records">**

**<th scope="row">{{customer.firstName}}</th>**

**<td>{{customer.id}}</td>**

**<td>{{customer.startDate}}</td>**

**</tr>**

**</table>**

<https://stackblitz.com/edit/angular-6-template-2e5kjj>

<https://stackblitz.com/edit/angular-ary5np?file=src%2Fapp%2Fapp.component.html>

Reactjs

Interterceptors

**import axios from 'axios';**

***//https://stackoverflow.com/questions/43210516/how-to-redirect-from-axios-interceptor-with-react-router-v4***

**export default {**

**interceptors: (store) => {**

**const {getState} = store;**

**console.log("Store: ",getState());**

***// request interceptors***

**axios.interceptors.request.use(function(config) {**

**const token = 'cookie.get(\_\_TOKEN\_KEY\_\_)';**

**if ( token != null ) {**

**console.log("testing");**

**config.headers.Authorization = `Bearer ${token}`;**

**}**

**return config;**

**}, function(err) {**

**return Promise.reject(err);**

**});**

***//response interceptors***

**axios.interceptors.response.use(response => {**

**return response;**

**}, error => {**

**if (error['response']['status'] === 401) {**

***//store.dispatch(logoutUser());***

**}**

**if (error['response']['status'] === 404) {**

***//history.push('/not-found');***

**}**

**return Promise.reject(error);**

**});**

**},**

**};**

**Usage in index.js file**

**import http from "./utils/httpInterceptors";**

**const store = createStore(rootReducer,applyMiddleware(thunk));**

**http.interceptors(store);**

Autocomplete or suggestions or input search widget

/\*

Read me for usaging autocomplete selector in html

author: Bhargav C.Kumar

---------------------------------------------------------------------------------------------------

<Autocomplete

placeholder="givename"

propName="propertyName"

elementName="inputName"

onSelectCallback={this.yourFuncatioName}

dataList={this.state.yourdata}

onChangeCallback={this.yourFuncatioName}

isRequired={true or false}

liveSearch={true}

provideClass="class 1 class 2"

noRecordsMessage="some thing message"

noRecordsShow={true}

isDesabled={true}

/>

1. placeholder -> this property for giving place holder for search input (optional).

2. propName -> this property for when you have array of object ( [{name: "Bhargav", gender: "male"}] ) then want show name in the list as a visible text(mandatory).

If data response is only simple array like ["name1","name2","name3"] then do not set property of 'propName'(not required).

3. elementName -> this property for providing name for input like (<input name="username") (optional).

4. onSelectCallback -> this property(method) for to retrieve selected item( returns two arguments like item and event) when selecting an item from filtered dropdown (optional or mandatory).

5. dataList -> this a data transaction property(on load data or live data), pass your data on this property for showing search results (mandatory).

6. onChangeCallback -> this property for to set a callback method to retrieve an event on every change from input (optional or if want to get event then this is mandatory).

7. isRequired -> this property for making field required true or false for validation (optional).

8. liveSearch -> use this property when you have live search filter list(payload) (this is mandatory when have live search).

9. provideClass -> this property for adding class to input you can add 'n' number of classes with space separate (optional).

10. noRecordsMessage -> this property for showing message text when records not found.

11. noRecordsShow -> this property mandatory if you want to show no records message.

12. isDesabled -> this property for disabling field (optional)

-------------------------------------------------------------------------------------------------

\*/

**import React from "react";**

**import PropTypes from "prop-types";**

**class Autocomplete extends React.Component {**

**static propTypes = {**

**dataList: PropTypes.instanceOf(Array)**

**};**

**constructor(props) {**

**super(props);**

**this.state = {**

***// The active selection's index***

**activeSuggestion: 0,**

***// The suggestions that match the user's input***

**filteredSuggestions: [],**

***// Whether or not the suggestion list is shown***

**showSuggestions: false,**

***// What the user has entered***

**userInput: ""**

**};**

**this.toggleContainer = React.createRef();**

**this.textInput = React.createRef();**

**this.onSelect = this.onSelect.bind(this);**

**this.onClickOutsideHandler = this.onClickOutsideHandler.bind(this);**

**}**

**componentWillReceiveProps(nextProps) {**

***// You don't have to do this check first, but it can help prevent an unneeded render***

**if (!nextProps['dataList']) {**

**let filteredSuggestions = [];**

**this.setState({**

**activeSuggestion: 0,**

**filteredSuggestions,**

**showSuggestions: false**

**});**

**}**

**if (nextProps['dataList'] && this.props['liveSearch']) {**

**let filteredSuggestions = nextProps['dataList'];**

**this.setState({**

**activeSuggestion: 0,**

**filteredSuggestions**

**});**

**}**

**}**

**componentDidMount() {**

**window.addEventListener("click", this.onClickOutsideHandler);**

**}**

**componentWillUnmount() {**

**window.removeEventListener("click", this.onClickOutsideHandler);**

**}**

**onClickOutsideHandler(event) {**

**if (this.state.showSuggestions && !this.toggleContainer.current.contains(event.target)) {**

**this.setState({**

**showSuggestions: false,**

**activeSuggestion: 0,**

**filteredSuggestions: []**

**});**

**}**

**}**

**onChangeValue = e => {**

**const {**

**dataList,**

**propName**

**} = this.props;**

**const userInput = e.currentTarget.value;**

**let filteredSuggestions = [];**

***// Filter our suggestions that don't contain the user's input***

**if (this.props['onChangeCallback']) {**

**this.props.onChangeCallback(e);**

**}**

**filteredSuggestions = this.props['liveSearch'] ? dataList : dataList.filter(item => (propName ? item[propName] : item).toLowerCase().indexOf(userInput.toLowerCase()) > -1);**

**this.setState({**

**activeSuggestion: 0,**

**filteredSuggestions,**

**showSuggestions: true,**

**userInput: e.currentTarget.value**

**});**

**};**

**onSelect(e, val) {**

**e.persist();**

**e.stopPropagation();**

**this.setState({**

**activeSuggestion: 0,**

**filteredSuggestions: [],**

**showSuggestions: false,**

**userInput: e.currentTarget.innerText**

**});**

**if (this.props['onSelectCallback']) {**

**let event = {};**

**event['target'] = this.textInput.current;**

**setTimeout(() => {**

**this.props.onSelectCallback(val, event);**

**}, 0);**

**}**

**};**

**onKeyDown = (e) => {**

**const {**

**activeSuggestion,**

**filteredSuggestions**

**} = this.state;**

***// User pressed the enter key***

**if (e.keyCode === 13 && filteredSuggestions['length']) {**

**e.preventDefault();**

**let event = {};**

**event['target'] = this.textInput.current;**

**if (this.props['onSelectCallback']) {**

**this.props.onSelectCallback(filteredSuggestions[activeSuggestion], event);**

**}**

**this.setState({**

**activeSuggestion: 0,**

**showSuggestions: false,**

**userInput: this.props['propName'] ? filteredSuggestions[activeSuggestion][this.props['propName']] : filteredSuggestions[activeSuggestion]**

**});**

**}**

***// User pressed the up arrow***

**else if (e.keyCode === 38) {**

**if (activeSuggestion === 0) {**

**return;**

**}**

**this.setState({**

**activeSuggestion: activeSuggestion - 1**

**});**

**this.elementHighlightVisible(e.target.parentNode, (activeSuggestion - 1));**

**}**

***// User pressed the down arrow***

**else if (e.keyCode === 40) {**

**if (activeSuggestion - 1 === filteredSuggestions.length) {**

**return;**

**}**

**if ((activeSuggestion + 1) < filteredSuggestions.length) {**

**this.setState({**

**activeSuggestion: activeSuggestion + 1**

**});**

**this.elementHighlightVisible(e.target.parentNode, (activeSuggestion + 1));**

**}**

**}**

**};**

**elementHighlightVisible($element, index) {**

**var container = $element.querySelectorAll('.suggestions');**

**var choices = container[0].querySelectorAll('.suggestion-item');**

**var highlighted = choices[index];**

**var posY = highlighted.offsetTop + highlighted.clientHeight - container[0].scrollTop;**

**var height = container[0].offsetHeight;**

**if (posY > height) {**

**container[0].scrollTop += posY - height;**

**} else if (posY < highlighted.clientHeight) {**

**if (index === 0) {**

**container[0].scrollTop = 0;**

**} else {**

**container[0].scrollTop -= highlighted.clientHeight - posY;**

**}**

**}**

**};**

**render() {**

**const {**

**onChangeValue,**

**onSelect,**

**onKeyDown,**

**state: {**

**activeSuggestion,**

**filteredSuggestions,**

**showSuggestions,**

**userInput**

**}**

**} = this;**

**let suggestionsListComponent;**

**if (showSuggestions && userInput) {**

**if (filteredSuggestions.length) {**

**suggestionsListComponent = (**

**<ul className = "suggestions" > {**

**filteredSuggestions.map((item, index) => {**

**let className;**

***// Flag the active item with a class***

**if (index === activeSuggestion) {**

**className = "suggestion-item suggestion-active";**

**} else {**

**className = "suggestion-item";**

**}**

**return (**

**<li className = {className} key = {`${item[this.props['propName']]}\_${index}`} onClick = { (e) => onSelect(e, item) } >**

**{this.props['propName'] ? item[this.props['propName']] : item}**

**</li>**

**);**

**})**

**}**

**</ul>**

**);**

**} else {**

**if (this.props['noRecordsShow'] === true) {**

**suggestionsListComponent= (**

**<div className="no-suggestions" >**

**<span>**

**{this.props['noRecordsMessage'] ? this.props['noRecordsMessage'] : "No suggestions"}**

**</span>**

**</div>**

**);**

**}**

**}**

**}**

**return (**

**<div className="autocomplete-box" ref={this.toggleContainer} >**

**<input**

**required={this.props.isRequired}**

**name={this.props.elementName}**

**className={this.props.provideClass}**

**type="text"**

**onChange={onChangeValue}**

**onKeyDown={onKeyDown}**

**value={userInput}**

**ref={this.textInput}**

**disabled={this.props.isDesabled}**

**/>**

**<label> {this.props.placeholder}</label>**

**{suggestionsListComponent}**

**</div>**

**);**

**}**

**}**

**export default Autocomplete;**

progressbar in react

function Progressbar(props) {

const [progressDetails, setProgressDetails] = React.useState({

stroke:"",

strokeDashoffset: 339.292

});

React.useEffect(()=>{

if(props && Object.keys(props).length) {

var RADIUS = 54;

var CIRCUMFERENCE = 2 \* Math.PI \* RADIUS;

var progress = (props.percentage / 100);

setTimeout(()=>{

setProgressDetails({strokeDashoffset: (CIRCUMFERENCE \* (1 - progress)), stroke: props?.color},);

*//vm.percentage(CIRCUMFERENCE \* (1 - progress))*

},1)

}

}, [props]);

return (

<><svg class="progress-circle" viewBox="0 0 120 120">

<circle cx="60" cy="60" r="54" fill="none" stroke="#ecebf2" stroke-width="8"></circle>

<circle {...progressDetails} class="progress\_\_value" cx="60" cy="60" r="54" fill="none" stroke-width="8"

></circle>

</svg>

<pre>{JSON.stringify(progressDetails)}</pre>

</>

);

}

CSS

.progress-circle{

width: 191px;

height: 191px;

transform: rotate(-90deg)

}

.progress\_\_value {

stroke-dasharray: 339.292;

*/\* stroke-dashoffset: 339.292;*

*animation: progress 2s linear alternate;*

*-webkit-animation-fill-mode: forwards;*

*animation-fill-mode: forwards; \*/*

-webkit-transition: stroke-dashoffset 2s ease-in;

-moz-transition: stroke-dashoffset 2s ease-in;

-ms-transition: stroke-dashoffset 2s ease-in;

-o-transition: stroke-dashoffset 2s ease-in;

transition: stroke-dashoffset 2s ease-in;

border-radius: 8px;

-webkit-border-radius: 8px;

-moz-border-radius: 8px;

-ms-border-radius: 8px;

-o-border-radius: 8px;

box-shadow: 0 0 5px 0 rgba(198, 198, 198, 0.5);

-webkit-box-shadow: 0 0 5px 0 rgba(198, 198, 198, 0.5);

-moz-box-shadow: 0 0 5px 0 rgba(198, 198, 198, 0.5);

stroke-linecap: round;

}

Another progressbar exclamatory mark sign

<!DOCTYPE html>

<html>

<body>

<style>

.propgressCircleInner {

width: 120px;

height: 120px;

border-radius: 50%;

stroke-width: 3;

stroke: #fff;

margin:auto;

box-shadow: inset 0px 0px 0px #7ac142;

animation: fill .4s ease-in-out .4s forwards, scale .3s ease-in-out .9s both;

}

.propgressCircleInner .checkmark\_\_circle {

stroke-dasharray: 166;

stroke-dashoffset: 166;

stroke-width: 1;

stroke: #cd0100;

stroke-linecap: round;

fill: none;

animation: stroke 0.6s cubic-bezier(0.65, 0, 0.45, 1) forwards;

}

.propgressCircleInner .checkmark\_\_animate {

transform-origin: 50% 50%;

opacity: 0;

animation: stroke2 0.3s cubic-bezier(0.65, 0, 0.45, 1) 0.8s forwards;

}

@keyframes stroke {

100% {

stroke-dashoffset: 0;

}

}

@keyframes stroke2 {

100% {

opacity:1;

}

}

@keyframes scale {

0%, 100% {

transform: none;

}

50% {

transform: scale3d(1.1, 1.1, 1);

}

}

@keyframes fill {

100% {

box-shadow: inset 0px 0px 0px 80px #cd0100;

}

}

</style>

<svg class="propgressCircleInner" xmlns="http://www.w3.org/2000/svg" viewBox="0 0 52 52">

<circle class="checkmark\_\_circle" cx="26" cy="26" r="25" fill="none"></circle>

<circle class="checkmark\_\_animate" id="exclamatory" fill="none" cx="26.5" cy="40" r="1"></circle>

<path d="M26 10h1v24h-1z" fill="none" class="checkmark\_\_animate"></path>

</svg>

</body>

</html>



Material-table details panel customi show hide

const tableRef = useRef();

  const [data, setData] = useState(rawData);

  /\*useEffect(() => {

    tableRef.current.state.data = tableRef.current.state.data.map(data => {

      console.log(data);

      data.tableData.showDetailPanel = tableRef.current.props.detailPanel;

      return data;

    });

  },[]);\*/

  const handleShowDetailPanel = event => {

    if (!tableRef.current) return;

    const { detailPanel } = tableRef.current.props;

    let handleShowDetailPanel = detailPanel;

    // If props.detailPanel is an array not a func

    if (typeof detailPanel === 'object') {

      // ~ I AM USING INDEX 0 HERE - CHANGE TO WHATEVER YOU NEED ~

      handleShowDetailPanel = detailPanel[0].render;

    }

    console.log('Detailspanel: ', detailPanel);

    const \_data = [...data]; // Make copy of data

    \_data.forEach((item, index) => {

      // If `showDetailPanel` already exists, remove it to close the detailPanel..

      if (\_data[index].tableData.showDetailPanel) {

        \_data[index].tableData.showDetailPanel = '';

      } else {

        // If `showDetailPanel` does not exist, add it to open detailPanel..

        \_data[index].tableData = {

          ...\_data[index].tableData,

          showDetailPanel: handleShowDetailPanel

        };

      }

    });

    setData(\_data);

  };

OR give directly on materialTable selector in inside data prop

data={filteredNotes.map((d, i) => ({

                  ...d,

                  tableData: {

                      showDetailPanel: detailsPanelData,

                  },

              }))}