'use strict';

const modal = document.querySelector('.modal');

const overlay = document.querySelector('.overlay');

const btnCloseModal = document.querySelector('.btn--close-modal');

const btnsOpenModal = document.querySelectorAll('.btn--show-modal');

const btnScrollTo = document.querySelector('.btn--scroll-to');

const section1 = document.querySelector('#section--1');

const nav = document.querySelector('.nav');

const tabs = document.querySelectorAll('.operations\_\_tab');

const tabsContainer = document.querySelector('.operations\_\_tab-container');

const tabsContent = document.querySelectorAll('.operations\_\_content');

///////////////////////////////////////

// Modal window

const openModal = function (e) {

e.preventDefault();

modal.classList.remove('hidden');

overlay.classList.remove('hidden');

};

const closeModal = function () {

modal.classList.add('hidden');

overlay.classList.add('hidden');

};

btnsOpenModal.forEach(btn => btn.addEventListener('click', openModal));

btnCloseModal.addEventListener('click', closeModal);

overlay.addEventListener('click', closeModal);

document.addEventListener('keydown', function (e) {

if (e.key === 'Escape' && !modal.classList.contains('hidden')) {

closeModal();

}

});

///////////////////////////////////////

// Button scrolling

btnScrollTo.addEventListener('click', function (e) {

const s1coords = section1.getBoundingClientRect();

console.log(s1coords);

console.log(e.target.getBoundingClientRect());

console.log('Current scroll (X/Y)', window.pageXOffset, window.pageYOffset);

console.log(

'height/width viewport',

document.documentElement.clientHeight,

document.documentElement.clientWidth

);

// Scrolling

// window.scrollTo(

// s1coords.left + window.pageXOffset,

// s1coords.top + window.pageYOffset

// );

// window.scrollTo({

// left: s1coords.left + window.pageXOffset,

// top: s1coords.top + window.pageYOffset,

// behavior: 'smooth',

// });

section1.scrollIntoView({ behavior: 'smooth' });

});

///////////////////////////////////////

// Page navigation

// document.querySelectorAll('.nav\_\_link').forEach(function (el) {

// el.addEventListener('click', function (e) {

// e.preventDefault();

// const id = this.getAttribute('href');

// console.log(id);

// document.querySelector(id).scrollIntoView({ behavior: 'smooth' });

// });

// });

// 1. Add event listener to common parent element

// 2. Determine what element originated the event

document.querySelector('.nav\_\_links').addEventListener('click', function (e) {

e.preventDefault();

// Matching strategy

if (e.target.classList.contains('nav\_\_link')) {

const id = e.target.getAttribute('href');

document.querySelector(id).scrollIntoView({ behavior: 'smooth' });

}

});

///////////////////////////////////////

// Tabbed component

tabsContainer.addEventListener('click', function (e) {

const clicked = e.target.closest('.operations\_\_tab');

// Guard clause

if (!clicked) return;

// Remove active classes

tabs.forEach(t => t.classList.remove('operations\_\_tab--active'));

tabsContent.forEach(c => c.classList.remove('operations\_\_content--active'));

// Activate tab

clicked.classList.add('operations\_\_tab--active');

// Activate content area

document

.querySelector(`.operations\_\_content--${clicked.dataset.tab}`)

.classList.add('operations\_\_content--active');

});

///////////////////////////////////////

// Menu fade animation

const handleHover = function (e) {

if (e.target.classList.contains('nav\_\_link')) {

const link = e.target;

const siblings = link.closest('.nav').querySelectorAll('.nav\_\_link');

const logo = link.closest('.nav').querySelector('img');

siblings.forEach(el => {

if (el !== link) el.style.opacity = this;

});

logo.style.opacity = this;

}

};

// Passing "argument" into handler

nav.addEventListener('mouseover', handleHover.bind(0.5));

nav.addEventListener('mouseout', handleHover.bind(1));

///////////////////////////////////////

// Sticky navigation: Intersection Observer API

const header = document.querySelector('.header');

const navHeight = nav.getBoundingClientRect().height;

const stickyNav = function (entries) {

const [entry] = entries;

// console.log(entry);

if (!entry.isIntersecting) nav.classList.add('sticky');

else nav.classList.remove('sticky');

};

const headerObserver = new IntersectionObserver(stickyNav, {

root: null,

threshold: 0,

rootMargin: `-${navHeight}px`,

});

headerObserver.observe(header);

///////////////////////////////////////

// Reveal sections

const allSections = document.querySelectorAll('.section');

const revealSection = function (entries, observer) {

const [entry] = entries;

if (!entry.isIntersecting) return;

entry.target.classList.remove('section--hidden');

observer.unobserve(entry.target);

};

const sectionObserver = new IntersectionObserver(revealSection, {

root: null,

threshold: 0.15,

});

allSections.forEach(function (section) {

sectionObserver.observe(section);

section.classList.add('section--hidden');

});

// Lazy loading images

const imgTargets = document.querySelectorAll('img[data-src]');

const loadImg = function (entries, observer) {

const [entry] = entries;

if (!entry.isIntersecting) return;

// Replace src with data-src

entry.target.src = entry.target.dataset.src;

entry.target.addEventListener('load', function () {

entry.target.classList.remove('lazy-img');

});

observer.unobserve(entry.target);

};

const imgObserver = new IntersectionObserver(loadImg, {

root: null,

threshold: 0,

rootMargin: '200px',

});

imgTargets.forEach(img => imgObserver.observe(img));

///////////////////////////////////////

// Slider

const slider = function () {

const slides = document.querySelectorAll('.slide');

const btnLeft = document.querySelector('.slider\_\_btn--left');

const btnRight = document.querySelector('.slider\_\_btn--right');

const dotContainer = document.querySelector('.dots');

let curSlide = 0;

const maxSlide = slides.length;

// Functions

const createDots = function () {

slides.forEach(function (\_, i) {

dotContainer.insertAdjacentHTML(

'beforeend',

`<button class="dots\_\_dot" data-slide="${i}"></button>`

);

});

};

const activateDot = function (slide) {

document

.querySelectorAll('.dots\_\_dot')

.forEach(dot => dot.classList.remove('dots\_\_dot--active'));

document

.querySelector(`.dots\_\_dot[data-slide="${slide}"]`)

.classList.add('dots\_\_dot--active');

};

const goToSlide = function (slide) {

slides.forEach(

(s, i) => (s.style.transform = `translateX(${100 \* (i - slide)}%)`)

);

};

// Next slide

const nextSlide = function () {

if (curSlide === maxSlide - 1) {

curSlide = 0;

} else {

curSlide++;

}

goToSlide(curSlide);

activateDot(curSlide);

};

const prevSlide = function () {

if (curSlide === 0) {

curSlide = maxSlide - 1;

} else {

curSlide--;

}

goToSlide(curSlide);

activateDot(curSlide);

};

const init = function () {

goToSlide(0);

createDots();

activateDot(0);

};

init();

// Event handlers

btnRight.addEventListener('click', nextSlide);

btnLeft.addEventListener('click', prevSlide);

document.addEventListener('keydown', function (e) {

if (e.key === 'ArrowLeft') prevSlide();

e.key === 'ArrowRight' && nextSlide();

});

dotContainer.addEventListener('click', function (e) {

if (e.target.classList.contains('dots\_\_dot')) {

const { slide } = e.target.dataset;

goToSlide(slide);

activateDot(slide);

}

});

};

slider();

///////////////////////////////////////

///////////////////////////////////////

///////////////////////////////////////

/\*

///////////////////////////////////////

// Selecting, Creating, and Deleting Elements

// Selecting elements

console.log(document.documentElement);

console.log(document.head);

console.log(document.body);

const header = document.querySelector('.header');

const allSections = document.querySelectorAll('.section');

console.log(allSections);

document.getElementById('section--1');

const allButtons = document.getElementsByTagName('button');

console.log(allButtons);

console.log(document.getElementsByClassName('btn'));

// Creating and inserting elements

const message = document.createElement('div');

message.classList.add('cookie-message');

// message.textContent = 'We use cookied for improved functionality and analytics.';

message.innerHTML =

'We use cookied for improved functionality and analytics. <button class="btn btn--close-cookie">Got it!</button>';

// header.prepend(message);

header.append(message);

// header.append(message.cloneNode(true));

// header.before(message);

// header.after(message);

// Delete elements

document

.querySelector('.btn--close-cookie')

.addEventListener('click', function () {

// message.remove();

message.parentElement.removeChild(message);

});

///////////////////////////////////////

// Styles, Attributes and Classes

// Styles

message.style.backgroundColor = '#37383d';

message.style.width = '120%';

console.log(message.style.color);

console.log(message.style.backgroundColor);

console.log(getComputedStyle(message).color);

console.log(getComputedStyle(message).height);

message.style.height =

Number.parseFloat(getComputedStyle(message).height, 10) + 30 + 'px';

document.documentElement.style.setProperty('--color-primary', 'orangered');

// Attributes

const logo = document.querySelector('.nav\_\_logo');

console.log(logo.alt);

console.log(logo.className);

logo.alt = 'Beautiful minimalist logo';

// Non-standard

console.log(logo.designer);

console.log(logo.getAttribute('designer'));

logo.setAttribute('company', 'Bankist');

console.log(logo.src);

console.log(logo.getAttribute('src'));

const link = document.querySelector('.nav\_\_link--btn');

console.log(link.href);

console.log(link.getAttribute('href'));

// Data attributes

console.log(logo.dataset.versionNumber);

// Classes

logo.classList.add('c', 'j');

logo.classList.remove('c', 'j');

logo.classList.toggle('c');

logo.classList.contains('c'); // not includes

// Don't use

logo.clasName = 'jonas';

///////////////////////////////////////

// Types of Events and Event Handlers

const h1 = document.querySelector('h1');

const alertH1 = function (e) {

alert('addEventListener: Great! You are reading the heading :D');

};

h1.addEventListener('mouseenter', alertH1);

setTimeout(() => h1.removeEventListener('mouseenter', alertH1), 3000);

// h1.onmouseenter = function (e) {

// alert('onmouseenter: Great! You are reading the heading :D');

// };

///////////////////////////////////////

// Event Propagation in Practice

const randomInt = (min, max) =>

Math.floor(Math.random() \* (max - min + 1) + min);

const randomColor = () =>

`rgb(${randomInt(0, 255)},${randomInt(0, 255)},${randomInt(0, 255)})`;

document.querySelector('.nav\_\_link').addEventListener('click', function (e) {

this.style.backgroundColor = randomColor();

console.log('LINK', e.target, e.currentTarget);

console.log(e.currentTarget === this);

// Stop propagation

// e.stopPropagation();

});

document.querySelector('.nav\_\_links').addEventListener('click', function (e) {

this.style.backgroundColor = randomColor();

console.log('CONTAINER', e.target, e.currentTarget);

});

document.querySelector('.nav').addEventListener('click', function (e) {

this.style.backgroundColor = randomColor();

console.log('NAV', e.target, e.currentTarget);

});

///////////////////////////////////////

// DOM Traversing

const h1 = document.querySelector('h1');

// Going downwards: child

console.log(h1.querySelectorAll('.highlight'));

console.log(h1.childNodes);

console.log(h1.children);

h1.firstElementChild.style.color = 'white';

h1.lastElementChild.style.color = 'orangered';

// Going upwards: parents

console.log(h1.parentNode);

console.log(h1.parentElement);

h1.closest('.header').style.background = 'var(--gradient-secondary)';

h1.closest('h1').style.background = 'var(--gradient-primary)';

// Going sideways: siblings

console.log(h1.previousElementSibling);

console.log(h1.nextElementSibling);

console.log(h1.previousSibling);

console.log(h1.nextSibling);

console.log(h1.parentElement.children);

[...h1.parentElement.children].forEach(function (el) {

if (el !== h1) el.style.transform = 'scale(0.5)';

});

///////////////////////////////////////

// Sticky navigation

const initialCoords = section1.getBoundingClientRect();

console.log(initialCoords);

window.addEventListener('scroll', function () {

console.log(window.scrollY);

if (window.scrollY > initialCoords.top) nav.classList.add('sticky');

else nav.classList.remove('sticky');

});

///////////////////////////////////////

// Sticky navigation: Intersection Observer API

const obsCallback = function (entries, observer) {

entries.forEach(entry => {

console.log(entry);

});

};

const obsOptions = {

root: null,

threshold: [0, 0.2],

};

const observer = new IntersectionObserver(obsCallback, obsOptions);

observer.observe(section1);

///////////////////////////////////////

// Lifecycle DOM Events

document.addEventListener('DOMContentLoaded', function (e) {

console.log('HTML parsed and DOM tree built!', e);

});

window.addEventListener('load', function (e) {

console.log('Page fully loaded', e);

});

window.addEventListener('beforeunload', function (e) {

e.preventDefault();

console.log(e);

e.returnValue = '';

});

\*/