**Documentation start configuration**

**JS (optimization)**

**Laziloading(**lazyLS.js**)**

**img**

Для картинок додємо **data-src** **з шляхом реальної картинки**, в саме ж **src закидуємо шлях для іконки в 1пк**. Для фону можна задати анімацію загрузки при бажанні\потреби.

Якщо в нас містить обгортка для img в вигляді **picture source** – створюємо замість **srcset** – **data-srcset** **із шляхо до справжньої картинки,** сам **srcset** **завантажуємо картинку 1пк**

**map**

В обгортку для **iframe** додаємо **data-map** із значення **src iframe-а** – сам **iframe** видаляємо. Обгортці задаємо клас **\_load-map**

**content**

Підгрузка контенту лише при скролі до певного моменту сторінки. Обгортці із контентом, що має підгружатись додаємо клас **\_load-more.** При підгрузці елементів блока-обгортки в ньому появиться блок із класом **\_loading-icon,** a самому блоку-обгортці тимчасовий клас **\_loading.** Завдяки яким можна стилізувати візуалізацію загрузки.

Сам контент підгружається із окремого html фала завдяки ajax

**JS (animation)**

1. **Animation (anim-scroll.js) – визначає чи еелмент з яким нам потрібно працювати знаходиться в полі зору:**

Для активації даного модуля існує діжурний клас - **\_anim-items.** Вішається він на елемент, елементи якого ми хочемо анімувати.

Для задання **анімації елементам, які мусять відображатись статично (параграфи)**, ми огортаємо їх в div, кожній такій обгортці задаємо діжурний клас **\_anim-items.** Виносимо стартову анімацію в окремий клас і в css прописуємо структуру

Для відображення анімації одноразово при скролі задаємо клас **\_anim-no-hide**

**.animated-class.\_active –** схема прописування стилів анімації елементів відносно сторінки

**.\_active .animated-class** {} – схема прописування стилів анімації для елементів відносно блоку обгортки (блоку-обгортці додається клас **\_anim-item**)

1. **Img-parallax (mouse-parallax.js) – добавляє картинкам паралакс ефект при руху мишкою чи скролі:**

Для активації скріпта в сторінка в своїй структурі повинна мати клас **.parallax.**

Якщо елемент працює зсунутий завдяки властивостям **transform,** то при налаштуваннях стартових позицій елемента, що буде паралакситись при руху мишки потрібно буде прописати їх занчення для того **transform(x or y),** який ми використовуємо в стилях

**JS (responsive)**

1. **Dynamic Adaptive (dynamic\_adaptive.js) – переміщення елементів в HTML структурі на вказаному діапазоні**

В html елементі, який ми хочемо переміщувати вказуємо **data - атребут – [data-da]** з значеннями (**клас елемента** в який ми хочемо перенести елемент, **проміжок (числом)** з якого застосовуватимуться зміни, **позиція елемента** відносно інших елементів).

Приклад **data-da=`.menu\_\_body, 768, last / first / (1/2/3../10..)`**

Для **зміни параметрів відліку при адаптив**і **з десктоптої (max-width)** на **мобільну верстку (min-width)** в файлі скрита, при його ініціалізації замінити параметр **max** на **min**

1. **CheckDeviceRange (checkRange.js) – перевіряє ширину вікна і контейнера для навішування каласів прив’язаних до конкретних діапазонів.**

Для коректної роботи даного скріпта необхідно в стилях для **:root** прописати змінні по яким скріпт зможе визначати коректно ширину девайса і контейнера і навішати правильні класи

:root{

            --msw: 1440px; - ширина контейнера

            --containerPadding: 30px; - паддінги контейнера

            --mainPadding: 20px; - основні падінги контенту

        }

**JS (checker)**

1. **Check device (checkMobile.js) – перевірка на використання сторінки мобільними девайсами**

**JS (helpers)**

1. **slowMoveToAnchorLink (slowAnchorMove.js) – плавне переміщення до потрібного об’єкту за кілком на якірне посилання**

**slowAnchorMove(**link, upSelector, topScrolling**)** додає плавний скрол для переміщення по усім якірним посиланням. І візуалізує елемент сторінки для переміщення вверх сторінки при прокрутці на певну відстань згори

**link =** передається селектор даного типу ( **[href^=’#’]** )

**upSelector =** селектор елемента для підйому вверх сторінки

**topScrolling =** к-сть пікселів, яку потрібно проскролити дляпояви елемента підйому вверх сторінки

1. **scrolling (scrolling.js)** – метод призначений для підйому на верх сторінки при кліку на елемент для прокрутки сторінки вгору

**scrolling(**selector**) –** в **selector** передаєм елемент, при кліку на який ми підіймаємось вверх сторінки

1. **Modals**

**bindModals(**triggerSelector, modalSelector, closeSelector, destroy, closeClickOverlay**) –** метод навішання логіки на модільні вікна

-**triggerSelector** = трігери на модільне вікно

-**modalSelector** = трігери на підлошку модільного вікна

-**closeSelector** = трігери на закриття вікна

-**closeClickOverlay** = true/false (true) на закриття модільного вікна по кліку на підлошку

- **destroy** = true/false (false) параметр, що вказує чи потрібно видаляти трігер на який викликається модальне вікно

Для візуальні ефекти відображення модального вікна задати стилі класу **\_visible**

**[data-modal] –** додаємо дата атребут для кожного модального вікна (overlay) щоб закрити усі вікна, при відкритті нового

**showModalByTime(**selector, time**) –** відкриття модільного вікна по таймінгу

**openByScroll(**selector**) –** відкриття вікна при умові, що ми доскролили кінця сторінки і не натиснули на жоден трігер

1. **Tabs**

**tabs(**headerSelector, tabSelector, contentSelector, activeClass, display = ‘block’**); -** метод для задання логіки табам

**headerSelector** = контейнер табів

**tabSelector** = конкрений таб

**contentSelector** = контент відповідний конкретному табу

**activeClass** = клас активності

**display** = тип display властивості який має задаватись елементам при їх відображені

По замовчуванню задати контенту стилі для приховування

Окремо задати стилі для класу активності поточного таба (**activeClass**) і класу активності конкретного контенту **\_content** і задамо їх першому табу і контенту

1. **ChangeModalState**

**changeModalState(**state**) –** метод для збору данних із полів різного типу для подільшої відправки на сервер

**state = {}** – пустий об’єкт створений окремо, який передається в метод для заповнення данними із полів

* В метолі отримуємо поля дані з яких нам потрібно отримати

**bindActionToElems(**event, elem, prop**)** – внутрішній метод метода changeModalState, завдяки якому по вканій події event, в вказаного елемента elem, ми братимемо значення і задаватимемо його в **state** під ключем prop. В самому методі прописано логіку отримання данних з елементів, які передані в метод в залежності від їх **nodeNam**

1. **Form**

**forms(**state**) –** форма для відправки данних із форми із кнопкою submit на сервер.

**State** – не обов’язковий атребут, що являє собою об’єкт із даними зібраними із усіх модільних вікон призначених для збору даних

Якщо в нас є потреба використовувати state, то в останньому модальному вікні призначеному для відправки даних з форми, задаємо дата-атребут **data-calc = end**

1. **formUpload**

**[name=’upload’] –** атребут вішається на поле, що буде приймати фого при його перетягуванні в це поле

1. **Timer**

**timer(**id, deadline**)** – метод для оживлення таймера

**id** = селектор блоку таймера

**deadline** = кінцевий час, до якого працюватиме таймер формату (2020-02-01)

**getTimeRemaining(**deadline**)** – метод повертає час переданий йому в **deadline** на **дні, години, хвилини, секунди** (внутрішній метод)

**setClock(**id, deadline) – внутрішній метод для timer, обгортка для логіки оновлення часу в функціональній частині і верстці

**updateClock()** – внутрішній метод для оновлення часу в верстці і зупинки таймера при закінчені часу

1. **Images**

**images(**workfield, popup**) –** метод для створення функціоналу по відкритті бальших варіантів фото в модальному вікні

**workfield** = робоча область із фото

**popup** = клас із стилями для overlay

**preview** = клас, який потрібно задати img тегам, в яких розміщені потрібні нам картинки. Кожну картинку обгорнути в посилання, в атребуті **href** якого є шлях, для картинки більше

1. **Slider**

**slider(**slides, dir, prev, next**) –** модуль для активації кастомного слайдера

slides = передаємо клас слайдів

dir = вказуємо спосіб гортання слайдерів (vertical/horizontal)

prev = селектор кнопки prev

next = селектор кнопки next

1. **Mask (**mask.js**)**

**mask(**selector**) –** фукнціонал по створенню маски для поля введення номера телефону

**selector =** поле для введення номера телефону

1. **ShowMoreCard (**showMoreSyles.js**) –** модуль для підгрузки прихованих елементів сторінки при кліку на трігер

**showMoreStyles(**trigger, wrapper**)**

**trigger** = трігер для візуалізації прихованих елеметів

**wrapper** = обгортка для елементів, які потрібно підгрузити чи візуалізувати

1. **Calc (**calc.js**) –** метод для обчислення вартості товару по параметрам введеним користувачем

**calc(**size, material, options, promocode, result**) –** в параметри передаємо поля чи селектори, значення яких ми будемо використовувати для розрахунків

1. **Accordion (**accordion.js**)** – модуль для активації акордеона

**accordion(**selector**) –** в селектор ми передаємо трігер на який користувач повинен клікнути при для відкриття акордеона

Стилі для контентної частини акордеона, яка приховується і візуалізовується (.active-style – стилі активності для трігера, .active-content – стилі активності для контенту)

.often-questions .accordion-block {

  background-color: #f7e7e6;

  max-height: 0;

  opacity: 0;

  transition: all 0.3s ease-out;

  overflow: hidden; /\*\*? фіксимо баг із елементом before/

}

.often-questions .accordion-block.active-content {

  overflow: visible;

  opacity: 1;

}

1. **Drag and Drop (**drop.js**) –** модуль для навішування на поле вводу функціонал перетягування фото із папки на поле

**drop(**selector**)**

**selector = передаємо усі поля із name=’upload’**

Additional params

Якщо ми хочемо, щоб поле приймало декілька картинок задаємо атрибут **multiple**,для передачі картинок будь-якого типу – прописуємо атрибут **accept='image/\*'**

**Mailer(**send.php**)**

Кожному тегу form задаємо атребут **action** – що має містити в собі шлях до файла мейлера (send.php).   
Для даного мейлера підходять файли php в деректорії **mailer.mg**

Тегу **body** в атребут class додаємо php код, який при успішному відправлені форми додасть клас

<?php echo isset($\_REQUEST['success']) ? 'success-' . htmlspecialchars($\_REQUEST['success']) : '' ?>