**Документація до веб-додатку “Календар”**

Код написаний для виконання функціонала календаря.

Версії мов програмування – HTML5, CSS3, php7.4.32

Використовується бібліотека CURL

Процес початкової ініціалізації загрузки веб-сайту:

Цей процес виконується функцією main\_init():

**0)** Для роботи коду необхідно створити константи підключення до кожної з трьох позицій календаря, спільного підключення до всіх тегів <div>, що мають в собі клітини для відображення днів місяця, та підключення до кожного тегу <select> у другій позицій календаря;

Також треба створити константу асоціативного масиву, де ключі - це значення днів тижня від 0 до 6, де 0 - це неділя, а 6 - це субота, а значення за ключами - це класи кожного з <div>, що мають в собі дні місяця.

**0.1)** Також треба зробити Promise до файлу api.php, який отримує дані про свята, оброблює ці дані та відправляє до файлу JS.

**0.1.1)** Робота файлу api.php:

1) Ініціалізація cURL з URL: 'https://holidayapi.com/v1/holidays?pretty&key=d36069ee-3d7a-45ea-8e32-4828cbf0a784&country=UA&year=2022'

, отримання даних формата JSON, збереження цих даних у змінну, закриття cURL.

2) Декодування даних JSON в асоціативний масив

3) Обробка даних: Для кожного свята, масив яких знаходиться за ключем „holidays’, перевіряємо значення ключа „public” – якщо true: збираємо дані з дати свята (ключ „date”), тип даних String; далі забираємо з дати значення місяця та дня, значення місяця знижуємо на один та зберігаємо дані за схемою: ключ — значення місяця, значення в ключах — значення днів всіх свят в цьому місяці, таким чином при збиранні даних значення місяця має тип даних Number, а значення днів — тип даних String.

4) Отриманий масив даних кодуємо в JSON та відправляємо до файлу JS.

**1)** Спочатку отримується поточна дата за допомогою створення екземпляра класу Date

**2)** Далі створюється екземпляр класу Created\_date(), який отримує необхідний атрибут date зі значенням поточної дати:

**2.1)** У цьому класі на основі отриманої дати утворюються початкові дані про місяць, рік та день поточної дати, які зберігаються як властивості класу:

**2.1.1)** Рік - це екземпляр класу Year(), який отримує атрибут зі значенням року. Цей клас має дві властивості, в одній він зберігає це значення року, в іншій визначає на основі отриманого значення чи високосний цей рік, чи ні. Перша властивість ‘year.value’ має тип даних Number, друга властивість ‘great\_year’ - тип даних Boolean, якщо true - то рік високосний.

**2.1.2)** Місяць - це екземпляр класу Month(), який отримує два атрибути - значення місяця від 0 до 11 та Boolean значення високосного року з екземпляру класу Year(). Цей клас має п’ять властивостей:

1. month\_value - значення місяця, тип даних Number
2. month\_ru - Назва місяця російською мовою, тип даних String
3. month\_ua - Назва місяця українською мовою, тип даних String
4. month\_en - Назва місяця англійською мовою, тип даних String
5. count\_day - Кількість днів в місяці, тип даних Number

Властивості від 2 до 5 залежать від отриманого значення місяця, враховуємо, що значення 0 відповідає місяцю Січень, а 11 - місяцю Грудень. Також при отриманні значення 1, який відповідає за Лютий, враховуємо значення високосності року, якщо true - властивість count\_day має значення 29, а не 28.

**2.1.3)** Значення дня зберігається як змінна, тип даних Number

**2.2)** Далі вираховуються значення місяця та року для першої та третьої позиції календаря, враховуючи деякі умови щодо зміни року в залежності від поточного місяця

**2.3)** Також створюються дати перших днів місяця для кожної позиції, як екземпляри класу Date(), зі значенням отриманих років, місяців та першого дня

**3)** Наступний етап - це додавання тегів <option> до вказаних тегів <select> з атрибутом value

**3.1)** Діапазон створення років від 1970 до 2035 року, атрибут value має значення кожного року, тип даних String, textContent - має таке ж значення та такий ж тип даних, що й value. Також виставляємо атрибут selected до того <option>, значення якого дорівнює значенню поточного року, значення поточного року отримується з властивості екземпляру класу Created\_date()

**3.2)** Діапазон створення років від 0 до 11, атрибут value має значення від 0 до 11, тип даних String, атрибут textContent - значення властивості month\_ru (Параграф 2.1.2), атрибут selected отримує <option> з атрибутом value, який дорівнює атрибуту month\_value (Параграф 2.1.2). Значення цих атрибутів береться зі значень для другої позиції. Той же процес повторюється для української мови, тобто атрибуту month\_ua (Параграф 2.1.2) та для англійської мови - атрибут month\_en (Параграф 2.1.2)

**4)** Функція add\_dates(), яка відповідає за розташування днів місяця та номерів неділі. Вона приймає 4 атрибути:

1. Дата першого дня місяця для кожної позиції, тип даних Object Date()
2. Місяць кожної позиції, тип даних Object Month()
3. Рік кожної позиції, тип даних Object Year()
4. Змінна, що має в собі підключення до кожної позиції

Атрибути дати, місяця та року отримуються від властивостей класу Created\_date(), для кожної позиції вказуються свої дані, тож функцію треба викликати тричі.

**4.1)** Спочатку проходимо по всім полям для днів місяця, змінюючи його атрибут textContent на “ ”

**4.2)** Далі створюємо чотири змінні:

1. Перша змінна - це Boolean значення перевірки місяця на кількість всіх днів в місяці = 31 та дня тижня для першого дня місяця = 6. Тобто якщо в місяці 31 день, а день тижня для першого дня місця - Субота, то значення true
2. Друга змінна - це Boolean значення перевірки місяця на кількість всіх днів в місяці = (31 або 30) та дня тижня для першого дня місяця = 0. Тобто якщо в місяці 31 або 30 днів, а день тижня для першого дня місця - Неділя, то значення true
3. Також створюємо змінну номера неділі, яка є в місяці - стартове значення = 0
4. Та четверта змінна - для відтворення початкової дати для відображення всіх днів місяця, стартове значення - undefined

**4.3)** Далі проходить перевірка на перші дві змінні:

**4.3.1)** Якщо перша змінна має значення true => четверта змінна - початкова дата стає екземпляром класу Date() зі значеннями року та місяця, та значенням дня = 3

**4.3.1.1)** Далі йде цикл від 3 до кількості днів у місяці; в кожній ітерації циклу створюється екземпляр класу Date(); отримується його день тижня та створюється підключення до блоку за ключем, що має значення дня тижня. Якщо значення циклу дорівнює 8 або 9, реалізуємо візуалізацію дробу.

**4.3.2)** Якщо друга змінна має значення true => 4 змінна стає таким самим екземпляром класу Date(), тільки значення дня дорівнює 2, а реалізація дробу проходить лише при значенні циклу = 8

**4.3.3)** Інакше реалізація відображення починається з першого дня місяця

**4.4)** Після розташування всіх днів виконується асинхронна функція holidays(), вона приймає три атрибути:

1) значення місяця першої позиції — тип даних Number

2) значення місяця другої позиції — тип даних Number

3) значення місяця третьої позиції — тип даних Number

Функція виконує перевірку всіх днів на кожній позиції на приналежність цієї дати до статусу свята.

**4.4.1)** Перша дія — це підключення до файлу „api.php”, тип даних object Response

**4.4.2)** Друга дія — розпакування даних в object Response в форматі JSON, тип даних Promise

**4.4.3)** Третя дія - розпакування та робота з даними в Promise:

1) Утворюємо масив , що містить дані, передані з Promise

2) Проходимо по кожному дню кожного місяця та чистимо наявний клас CSS, який відповідає за свята

3) Проходимо по кожному дню кожного місяця та перевіряємо приналежність дати до святкової дати

**4.5)** Далі виконується нова функція getWeek() - вона приймає 4 атрибути:

1. Рік - тип даних Object Year()
2. Підключення до позиції
3. Місяць - тип даних Object Month()
4. Дата, від якої починається реалізація відображення днів місяця

Рік, підключення та Місяць отримуються з атрибутів функції add\_dates().

Дата утворюється під час перевірки перших двох утворених змін (Параграф 4.2 - 4.3)

Ця функція виконує підрахування та відображення номерів тижнів в місяці кожної позиції.

**4.5.1)** Спочатку утворюється змінна - екземпляр класу Date() зі значенням року взятого з властивості атрибуту Рік, значенням місяця = 0, та значенням дня = 1. Таким чином утворюється дата першого дня вказаного року. Далі дізнається індекс цього дня, що у випадку неділі дорівнює 7, а в інших випадках дорівнює номеру дня неділі, взятого за допомогою методу getDay(). Знаходимо номер неділі для дати відліку, бо вона першою виступає у відображенні. Видаляємо контент, який був раніше (ця функція необхідна при подальших змінах) та перевіряємо місяць на Лютий - якщо перше лютого випадає на понеділок, а днів у лютому 28, то тижнів буде всього 4, тож треба змінити порядок відображення номерів неділі.

Інакше створюємо змінну зі значеннями останнього дня вказаного року, тобто 31 грудня. Проходить цикл від 0 до 5, при якому перевіряється місяць і день тижня, якщо місяць - Грудень або якщо місяць - Січень, інакше - виставляємо номера неділі, отримані за формулою.

На цьому етапі дія функції add\_dates() закінчується повністю.

**5)** Далі виконується функція add\_data(), що відображає місяць і рік для першої та третьої позиції календарю. Вона приймає чотири атрибути, два перших з яких є об’єктом класу Рік та Місяць для першої позиції, а два інші - такі ж об’єкти, тільки для третьої позиції

**6)** Наступний етап - це отримання значення вибраного дня. Перевірка враховує наявність у місяці відображення дробу та в залежності від отриманого значення дня від поточної дати виділяє поточний день.

На цьому початкова загрузка сайту завершена. Далі сайт очікує змін від користувача. На сайті існує п’ять варіантів змін - зміна року, зміна місяця за кожною мовою (окремі три зміни) та виділення іншого дня.

**7)**  Перша зміна - зміна року. Перші дії - це отримання зміненного значення року та поточного значення місяця від російського варіанта місяця. Ці дані мають тип даних Number. Далі створюється дата - екземпляр класу Date(), зі значеннями року і місяця, отриманих від користувача, а день підраховується функцією checked\_date(), вона приймає в собі два атрибути - значення місяця та року/

**7.1)** Функція checked\_date() виконує такі дії:

1. Утворює дві змінні - екземпляри класів Year() та Month().
2. Далі проходять перевірки на вибраний день, що має в собі відображення дробу, такий день далі віддає день, що є меншим у дробі
3. Враховує, що в місяці може бути менше днів, ніж значення дня, що обрав користувач
4. Та повертає значення дня від вибраного дня, тип даних - Number

**8)** Дії при зміні місяці майже ідентичні - також береться значення року та місяця, тільки в залежності від мови обирається значення саме того <select> мови. Наступною дією для кожної зміни року або місяця - це виконання функції change\_init() - про неї буде пізніше (Параграф 8.1). А ось після виконання цієї функції - проводиться зміна для атрибуту selected для <option> кожного <select> - значення місяця повинні співпадати один з одним - місяць, що був обраний, повинен бути однаковим для трьох варіантів мови.

**8.1)**  Функція change\_init() - приймає один атрибут - дату, що була створена після збирання даних про місяць, день та рік, тип даних - Object Date(). перша її дія - створена екземпляру класу Created\_date() ( Параграф 2)

**8.2)** Наступні дії - виконання функції add\_dates() тричі та add\_data() (Параграфи 4 та 5).

**8.3)** Після цих функції здійснюється очищення відображення всіх обраних колись днів та в залежності від отриманої дати, відображення обраного користувачем дня, враховуючи деякі умови відображення дробу.

На цьому виконання загрузки сайту за зміненими користувачем даними завершена.

**9)** Остання можлива зміна - це зміна обраного дня. Зміна проходить за кліком по обраному дню. Дії після кліку:

1. Очищуються всі обрані колись дні
2. Відображається обраний зараз користувачем день