Управление состоянием Vue-компонентов. Директивы условной компиляции и отрисовки списка.

Изучите директивы условной компиляции [1] и директивы отрисовки списка [2]. Обратите внимание на роль атрибута «key» при отрисовке списков посредством директивы v-for. Здесь представлена хорошая статья для понимания этого [3].

Изучите способы регистрации и переиспользования SFC-компонентов [4], а также передачу реактивных состояний в эти компоненты с помощью пропсов [5].

В заданиях Вам понадобится эмиттировать события из SFC-компонента, для этого рекомендуется ознакомиться с разделом [6].

Вопросы и задания

1. Назовите минимум два отличия работы директив v-if и v-show. Предложите для каждого из следующих случаев выбор необходимой директивы и обоснуйте свой выбор:

    - карточки в игре на поиск соответствий, где при клике на карточку показывается изображение и исчезает через секунду;
    - переключение вида отображения элементов с сетки на список и наоборот;
    - показать кнопку панели администратора в зависимости от роли пользователя, выбранной при авторизации;
    - показать или скрыть Vue-компонент, хранящий некоторое заданное (отличное от дефолтного) состояние, сохранив это состояние.

2. Предположим Вам необходимо отрисовать список текстов, находящихся в массиве texts = [“text1”, “text2”, …], при помощи тэгов <p>. Однако, допустим, нам требуется вывести не каждый текст, а только те, что имеют четное число букв. Предложите два способа реализации этой задачи на Vue. Какой способ более оптимальный для очень большого неизменяемого списка и почему? Тот же вопрос для небольшого списка. Тот же вопрос для большого и небольшого списка в случае его частых мутаций.

3. Для чего требуется атрибут «key» при работе со списком? Предоставьте фрагмент кода на Vue в качестве примера (придумайте сами), демонстрирующий проблему обновления состояния вложенных компонентов для элементов некоторого списка.

4. Продолжите работу над приложением «notes-app». Добавьте состояния и обработчики событий для строки поиска (например, searchQuery = ref(“”)), для переключателей вида списка и т. д.

5. Создайте SFC-компонент заметки согласно дизайну. Поместите его в директорию /src/views/notes/components/note. Добавьте все необходимые состояния, пропсы и обработчики событий. Для кнопок меню заметки (редактирование, удаление) пока можете сделать только эмиты событий из компонента. Для реализации закрытия меню взаимодействия с заметкой при клике на область мимо самого меню, может оказаться полезной статья про click outside [7].

6. Выведите список заметок на основную страницу (компонент «notes»). Для получения данных списка сделайте временно реф на массив объектов следующего вида (должен быть валидный JSON формат):

```
{
   id:      Number,          (уникальный в массиве)
   title:   String,          (не может быть пустой строкой)
   text:    String,          (может быть пустой строкой)
   image:   String | null,   (ссылка или null)
   date:    String           (ISO - формат)
}
```

Придумайте порядка 20-30 объектов заметок с различным наполнением. Пример:

```
[
   {
         id: 0,
         title: “моя первая заметка”,
         text: “не забыть сделать еще заметки”,
         image: null,
         date: “2025-07-09T15:50:36+03:00”
   },
```

{
id: 0,
title: “моя вторая заметка”,
text: “не забыть сделать еще заметки”,
image: “https://pw.artfile.me/wallpaper/26-06-2015/650x435/zhivotnye-krysy--myshi-kolosok-myshka-943726.jpg”,
date: “2025-07-09T15:50:36+03:00”
},
…

].

Не забудьте обработать состояние для вывода общего кол-ва заметок в футере.

Ссылки

[1]. https://vuejs.org/guide/essentials/conditional.html

[2]. https://vuejs.org/guide/essentials/list.html

[3]. https://vueschool.io/articles/vuejs-tutorials/tips-and-gotchas-for-using-key-with-v-for-in-vue-js-3/

[4]. https://vuejs.org/guide/components/registration.html

[5]. https://vuejs.org/guide/components/props.html

[6]. https://vuejs.org/guide/components/events.html

[7]. https://discuss.hotwired.dev/t/best-practices-for-handling-clicks-outside-element/1266/2