**Buuza-vite проект**

**СОДЕРЖАНИЕ:**

**Обмен данными между дочерним и родительским компонентом. (emit) Time: 00:43:26**

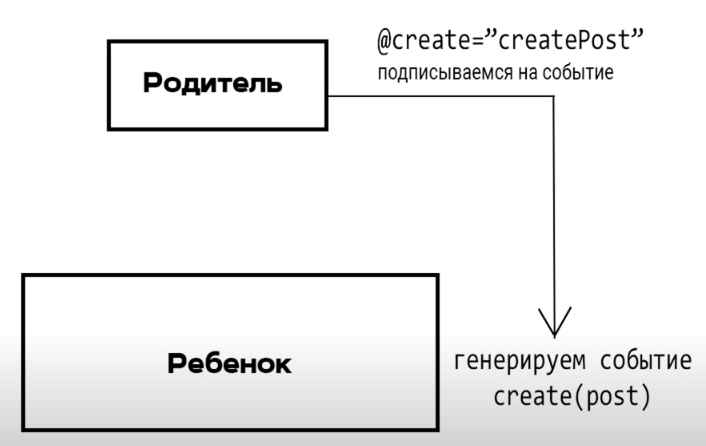
**Тимур Ульби -** [**https://www.youtube.com/watch?v=XzLuMtDelGk&t=2303s**](https://www.youtube.com/watch?v=XzLuMtDelGk&t=2303s)

**Обмен данными между дочерним и родительским компонентом. (emit) Time: 00:43:26**

**[СОДЕРЖАНИЕ:](#СОДЕРЖАНИЕ)**

**Тимур Ульби -** [**https://www.youtube.com/watch?v=XzLuMtDelGk&t=2303s**](https://www.youtube.com/watch?v=XzLuMtDelGk&t=2303s)

Для передачи данных от дочернего к-та к родительскому , мы генерируем событие в дочернем к-те (например: событие **create(post)** с аргументом **post**, post создается внутри дочернего к-та) , а в родительском к-те подписываемся на это событие (например: **@create=”createPost”**)



Например события **инпут** или **клик -** аргументом принимают **event.**

На 27.02.2024 у себя в проекте имеем:

**App.vue**

<script setup>  
import {ref**,** provide**,** watch**, *computed*,** onMounted**,** reactive} from 'vue'**;**import ***axios*** from 'axios'**;**import ***AppFigure*** from '@/components/AppFigure.vue'**;**import AppHeader from '@/components/AppHeader.vue'**;**import ***AppNav*** from '@/components/AppNav.vue'**;**import ***AppFooter*** from '@/components/AppFooter.vue'**;**import ***AppRowMenu*** from '@/components/AppRowMenu.vue'**;**import Drawer from '@/components/Drawer.vue'**;**import AppAside from '@/components/AppAside.vue'**;**import CardList from '@/components/CardList.vue'**;**const items = ref([])**;**const cart = ref([])**;**// const favorites = ref([]);  
const filters = reactive({  
 sortBy: 'category'**,**})**;**// Флаг корзины Активна-Не активна (полная-пустая) true-false.  
const drawerOpen = ref(false)**;**  
// Ф-ия отключает-закрывает корзину  
const closeDrawer = () => {  
 drawerOpen.value = false**;**}**;**// Ф-ия включает-открывает корзину  
const openDrawer = () => {  
 drawerOpen.value = true**;**}**;**const onChangeSelect = event => {  
 filters.sortBy = event.target.value**;**}**;**const totalPrice = ***computed***(  
 () => cart.value.reduce((acc**,** item) => {  
 return acc + item.price**;** }**,** 0)  
)**;**const taxPrice = ***computed***(  
 // () => totalPrice.value \* 0.05  
 () => ***Math***.round((totalPrice.value \* 5) / 100)  
)**;**const addToCart = (item) => {  
 cart.value.push(item)**;** item.isAdded = true**;**}**;**const removeFromCart = (item) => {  
 cart.value.splice(cart.value.indexOf(item)**,** 1)**;** item.isAdded = false**;**}**;**// Функция addToFavorite() добавляет на сервер в базу данных "Избрано", избранный товар.  
const addToFavorite = async (item) => {  
 try {  
 if (!item.isFavorite) {  
 const obj = {  
 parentId: item.id**,** item**,** }**;** item.isFavorite = true**;** const {data} = await axios.post('https://1102df40d9a2f61e.mokky.dev/favorites'**,** obj)**;** item.favoriteId = data.id**;** } else {  
 item.isFavorite = false**;** await axios.delete(`https://1102df40d9a2f61e.mokky.dev/favorites/${item.favoriteId}`)**;** item.favoriteId = null**;** }  
 } catch (err) {  
 ***console***.log(err)**;** }  
}**;**// Функция fetchItems() возвращает нам отсортированные списки по Поиску и по Слову  
const fetchItems = async () => {  
 try {  
 const params = {  
 sortBy: filters.sortBy  
 }**;** const {data} = await axios.get('https://1102df40d9a2f61e.mokky.dev/items'**,** {params})**;** items.value = data.map((obj) => ({  
 ...obj**,** isFavorite: false**,** favoriteId: null**,** isAdded: false  
 }))**;** } catch (err) {  
 ***console***.log(err)**;** }  
}**;**// Функция fetchFavorites() возвращает нам список Избранных товаров  
const fetchFavorites = async () => {  
 try {  
 const {data: favorites} = await axios.get('https://1102df40d9a2f61e.mokky.dev/favorites')**;** items.value = items.value.map((item) => {  
 const favorite = favorites.find((favorite) => favorite.parentId === item.id)**;** if (!favorite) {  
 return item**;** }  
  
 return {  
 ...item**,** isFavorite: true**,** favoriteId: favorite.id**,** }**;** })**;** } catch (err) {  
 ***console***.log(err)**;** }  
}**;**// Этот watch следит, что если корзина меняется, то все item сохраняются в localStorage.  
watch(cart**,** () => {  
 ***localStorage***.setItem('cart'**, *JSON***.stringify(cart.value))**;** }**,** {deep: true}  
)**;**onMounted(async () => {  
 const localCart = ***localStorage***.getItem('cart')**;** cart.value = localCart ? ***JSON***.parse(localCart) : []**;** await fetchItems()**;** // await fetchFavorites();  
  
 /\*try {  
 const {data} = await axios.get('https://1102df40d9a2f61e.mokky.dev/favorites');  
  
 favorites.value = data.map(obj => obj.item);  
 console.log(data);  
 } catch (err) {  
 console.log(err);  
 }\*/  
  
 items.value = items.value.map((item) => ({  
 ...item**,** isAdded: cart.value.some((cartItem) => cartItem.id === item.id)  
 }))**;**})**;**// Этот watch следит, что если корзина меняется, то все item убираются из добавленных.  
watch(cart**,** () => {  
 items.value = items.value.map((item) => ({  
 ...item**,** isAdded: false  
 }))**;**})**;**watch(filters**,** fetchItems)**;**// Предоставляем данные для дочерних компонентов  
provide('cartActions'**,** {  
 cart**,** closeDrawer**,** openDrawer**,** addToCart**,** removeFromCart**,** totalPrice**,** taxPrice**,**})**;**</script>  
  
<template>  
 <Drawer  
 v-if="drawerOpen"  
 :total-price="totalPrice"  
 :tax-price="taxPrice"  
 />  
 <div>  
 <AppFigure/>  
 <AppHeader  
 :total-price="totalPrice"  
 @open-drawer="openDrawer"  
 />  
 <AppNav/>  
 <AppRowMenu/>  
 <AppFooter/>  
 <section class="dish-menu">  
 <AppAside  
 @on-change-select="onChangeSelect"  
 />  
  
<!-- :items="favorites"-->  
<!-- is-favorites-->  
  
 <CardList  
 :items="items"  
 @add-to-cart="addToCart"  
 @remove-from-cart="removeFromCart"  
 @add-to-favorite="addToFavorite"  
 />  
 </section>  
 </div>  
</template>

**AppAside.vue**

<script setup>  
  
***defineProps***({  
 items: ***Array*,** category: String**,** isFavorites: Boolean**,** onChangeSelect: Function**,**})  
  
const emit = ***defineEmits***(['onChangeSelect'])  
</script>  
  
<template>  
 <aside>  
 <div class="items">  
 <button  
 type="button"  
 class="btn item"  
 @click="emit('onChangeSelect'**,** 'Гриль')"  
 >  
 <span>Гриль</span>  
 </button>  
  
  
<!-- <button type="button" class="btn item"><span>Гриль</span></button>-->  
 <button type="button" class="btn item"><span>Наборы</span></button>  
 <button type="button" class="btn item"><span>Буузы</span></button>  
 <button type="button" class="btn item"><span>Пельмени</span></button>  
 <button type="button" class="btn item"><span>Сковороды</span></button>  
 <button type="button" class="btn item"><span>На компанию</span></button>  
 <button type="button" class="btn item"><span>К Пиву</span></button>  
 <button type="button" class="btn item"><span>Нагетсы</span></button>  
 <button type="button" class="btn item"><span>Супы</span></button>  
 <button type="button" class="btn item"><span>Лапша</span></button>  
 <button type="button" class="btn item"><span>Салаты</span></button>  
 <button type="button" class="btn item"><span>Гарниры</span></button>  
 <button type="button" class="btn item"><span>Соусы</span></button>  
 <button type="button" class="btn item"><span>Закуски</span></button>  
 <button type="button" class="btn item"><span>Для детей</span></button>  
 <button type="button" class="btn item"><span>Блины</span></button>  
 <button type="button" class="btn item"><span>Десерты</span></button>  
 <button type="button" class="btn item"><span>Напитки</span></button>  
 </div>  
 </aside>  
</template>

**CardList.vue**

<script setup>  
import CardItem from '@/components/CardItem.vue'**;  
  
*defineProps***({  
 items:***Array*,** isAdded: Boolean**,** isFavorites: Boolean**,**})  
  
const emit = ***defineEmits***(['addToFavorite'**,** 'addToCart'**,** 'removeFromCart'])**;**</script>  
  
<template>  
 <main>  
 <div class="title-menu">Популярные блюда</div>  
 <div class="items">  
 <CardItem  
 v-for="item in items"  
 :key="item.id"  
 :id="item.id"  
 :title="item.title"  
 :image-url="item.imageUrl"  
 :category="item.category"  
 :ingredients="item.ingredients"  
 :weight="item.weight"  
 :price="item.price"  
 :is-added="item.isAdded"  
 :is-favorite="item.isFavorite"  
 :on-click-add="!isAdded ? ()=>emit('addToCart'**,** item) : ()=>emit('addToCart'**,** item.count)"  
 :on-click-remove="isFavorites ? null : ()=>emit('removeFromCart'**,** item)"  
 :on-click-favorite="isFavorites ? null : ()=>emit('addToFavorite'**,** item)"  
 />  
 </div>  
 </main>  
</template>

**CardItem.vue**

<script setup>  
import {ref} from 'vue'**;**const count = ref(0)**;  
  
*defineProps***({  
 id: Number**,** category: String**,** imageUrl: String**,** title: String**,** ingredients: String**,** weight: Number**,** price: Number**,** isAdded: Boolean**,** isFavorite: Boolean**,** onClickAdd: Function**,** onClickRemove: Function**,** onClickFavorite: Function**,**})**;**const inc = () => {  
 count.value++**;**}**;**const dec = () => {  
 if (count.value > 0) {  
 count.value--**;** }  
}**;**</script>  
  
<template>  
  
 <div class="item">  
  
 <div class="dish-img">  
 <img :src="imageUrl" alt="Dish"/>  
 <div class="dish-amount" v-if="count>0">{{ count }}</div>  
 </div>  
  
 <div class="subtitle">{{ title }}</div>  
 <!-- <div class="ingredients">  
 {{ ingredients }}  
 </div>-->  
 <div class="weight">Вес блюда {{ weight }} г.</div>  
  
 <div class="price-add">  
 <a v-if="count>0" @click="dec()**;** onClickRemove()">  
 <img  
 class="cursor-pointer"  
 src="/images/minus-50.png"  
 style="width: 30px**;** height: 30px**;** margin: 1px**;**"  
 alt="Minus"  
 />  
 </a>  
  
 <p>{{ price }} &#8381;</p>  
  
 <a @click="inc()**;** onClickAdd()">  
 <img  
 class="cursor-pointer"  
 src="/images/plus-50.png"  
 style="width: 30px**;** height: 30px**;** margin: 1px**;**"  
 alt="Plus"  
 />  
 </a>  
 </div>  
 </div>  
</template>  
  
<style scoped>  
  
.price-add {  
 display: flex**;** justify-content: space-between**;** background: lime**;** color: #0e4199**;** font-size: 18px**;** align-items: center**;** border-radius: 10px**;** padding: 5px**;**}  
  
.dish-img {  
 position: relative**;** /\*z-index: -1;\*/  
}  
  
.dish-amount {  
 font-size: 3.75rem**;** /\*line-height: 1;\*/  
 /\*font-weight: 400;\*/  
 position: absolute**;** top: 0**;** left: 0**;** right: 0**;** bottom: 0**;** /\*display: -webkit-box;\*/  
 /\*display: -ms-flexbox;\*/  
 /\*display-box: content;\*/  
 display: flex**;** /\*-webkit-box-align: center;\*/  
 /\*-ms-flex-align: center;\*/  
 align-items: center**;** /\*-webkit-box-pack: center;\*/  
 /\*-ms-flex-pack: center;\*/  
 justify-content: center**;** border-radius: 10px**;** margin-bottom: 4px**;** color: lime**;** background: rgba(3**,** 0**,** 41**,** .4)**;**}  
</style>

**Стоит задача : при клике на кнопку в боковой панели Aside ( так же в Row) вызвать сортировку массива items по ключу category и передачу на отрисовку rendering отсортированного по категории блюд (Например: «Гриль») массива.**

DefineEmits

**defineEmits** — это макрос в Vue.js Composition API, позволяющий объявить события, которые компонент может передавать своему родителю. Он указывается внутри в компоненте, и помогает сделать API компонента более наглядным и самодокументированным.

Давайте посмотрим на пример использования **defineEmits**:

<!-- MyComponent.vue -->  
<script setup>  
**import** { defineEmits } from 'vue'  
  
**const** emit = defineEmits(['my-event'])  
  
**function** triggerEvent() {  
 emit('my-event', 'payload')  
}  
</script>

В данном примере мы объявляем, что компонент может генерировать событие под названием **my-event.** Затем мы используем функцию **emit**, возвращаемую **defineEmits**, чтобы выдать это событие с полезной нагрузкой (**payload**) при вызове функции **triggerEvent**.

Это очень удобно, поскольку позволяет документировать события компонентов в одном месте в случае, если в одном компоненте происходит несколько событий. Кроме того, теперь мы можем проверять полезную нагрузку.

<script>  
**export** **default** {  
 // определяем как объект для проверки  
 emits:{  
  
 // или предоставляем функцию для проверки  
 // возвращает истинные значения для валидных, ложные - для невалидных  
 'check-name'(payload){  
 **return** !!(**typeof** payload === 'string')  
 }  
 },  
 data() {  
 **return** {  
 name:''  
 }  
 },  
 methods: {  
 sendData() {  
  
 **this**.$emit('check-name', **this**.name)  
 }  
 }  
}  
</script>

**[СОДЕРЖАНИЕ: Time: 00:43:26](#СОДЕРЖАНИЕ)**