

Интеграция Яндекс.Карт в веб-приложении на React/TypeScript

Андриенко Иван Сергеевич

Приамурский государственный университет имени Шолом-Алейхема

Студент

Пащенко Екатерина Александровна

Приамурский государственный университет имени Шолом-Алейхема

Студент

Аннотация

В статье рассматривается реализация интеграции сервиса Яндекс.Карты в одностраничное веб-приложение, разработанное на основе React и TypeScript. Описывается подход к динамической загрузке JavaScript API Яндекс.Карт, инициализации карты в компоненте Admission, а также обработке типовых ошибок, связанных с загрузкой скрипта и готовностью DOM-элементов. Особое внимание уделяется использованию глобального объекта window.ymaps в контексте типизации TypeScript, управлению состояниями загрузки карты и ошибок через хуки useState и useEffect, а также организации пользовательского интерфейса с индикацией процесса загрузки и сообщений о недоступности карты. Реализация выполнена с использованием демо-доступа к API без явного использования ключа, что позволяет обеспечить базовую функциональность для отображения местоположения организации.

Ключевые слова: Яндекс.Карты, React, TypeScript, веб-приложение, JavaScript API, динамическая загрузка скрипта, карта, интеграция сервисов.

Yandex integration.Maps in the React/TypeScript web application

Andrienko Ivan Sergeevich

Sholom-Aleichem Priamursky State University

Student

Paschenko Ekaterina Alexandrovna

Sholom-Aleichem Priamursky State University

Student

Abstract

The article discusses the implementation of the integration of the Yandex.Maps service into a single-page web application based on React and TypeScript. The approach to dynamic loading of the Yandex JavaScript API is described. Maps, initialization of the map in the Admission component, as well as handling typical errors related to script loading and the availability of DOM elements. Special attention is paid to using the global window.ymaps object in the context of

TypeScript typing, managing map loading and error states through the `useState` and `useEffect` hooks, as well as organizing a user interface with an indication of the loading process and messages about the unavailability of the map. The implementation is performed using demo API access without explicitly using a key, which allows you to provide basic functionality for displaying the organization's location.

Keywords: Yandex.Maps, React, TypeScript, web application, JavaScript API, dynamic script loading, map, service integration.

1 Введение

1.1 Актуальность

Интеграция картографических сервисов в веб-приложения является важной задачей для образовательных, коммерческих и информационных ресурсов, поскольку наличие интерактивной карты повышает удобство навигации и позволяет пользователю быстро определить расположение организации. Для учебных заведений это особенно актуально, так как будущие абитуриенты и их родители часто обращаются к официальным сайтам за информацией о местоположении, транспортной доступности и инфраструктуре. Использование сторонних API, таких как Яндекс.Карты, требует от разработчика не только знания клиентских веб-технологий, но и понимания особенностей работы с внешними скриптами, асинхронной инициализацией и обработкой ошибок. В условиях применения современных фронтенд-фреймворков, например React, задача усложняется необходимостью гармонично встроить картографический сервис в компонентную архитектуру и обеспечить корректное взаимодействие с системой состояний и жизненным циклом компонентов. Это делает исследование практических подходов к интеграции Яндекс.Карт в React-приложения актуальным для сферы веб-разработки.

1.2 Обзор исследований

Интеграция картографических сервисов в программные приложения на основе React и TypeScript находит отражение в современной научной литературе. В работе А.А. Севера рассматривается использование API карт Google Maps, Yandex.Maps и 2GIS в приложении на JavaFX, где анализируются особенности подключения различных картографических сервисов, их программные интерфейсы и практические аспекты реализации отображения карт и маркеров в десктопной среде [1]. Н.И. Филиппов и А.В. Лошаков выполняют сравнительный анализ сервисов Google Maps и Yandex.Maps, оценивая их функциональные возможности, точность картографических данных, удобство интеграции и особенности использования в прикладных задачах [2]. В статье Д.А. Хайруллина и М.Р. Богданова рассматриваются особенности применения статической типизации в React-приложениях, включая методики типизации компонентов, хуков и состояний, а также влияние строгой типизации на надёжность пользовательских интерфейсов [3]. К.Р. Удальцов исследует вопросы обеспечения качества кода

в командной разработке, предлагая формализацию шаблонов безопасных pull request-ов для проектов на React и TypeScript в рамках Git Flow [4]. Особое внимание созданию библиотек компонентов на основе TypeScript уделяется в работе Т.М. Воронко, где описываются принципы модульности, повторного использования кода и проектирования архитектуры компонентов для масштабируемых React-решений [5].

1.3 Цель исследования

Целью данной работы является разработка и описание подхода к интеграции Яндекс.Карт в React/TypeScript-приложение для отображения местоположения Политехнического техникума в г. Биробиджане. В рамках исследования рассматриваются методы динамической загрузки JavaScript API Яндекс.Карт, инициализации карты с использованием глобального объекта `ymaps`, управление состояниями загрузки и ошибок, а также организация пользовательского интерфейса, включающего индикацию процесса загрузки и обработку ситуаций недоступности картографического сервиса.

2 Материалы и методы

В качестве основы для реализации использовалось одностраничное веб-приложение, разработанное на библиотеке React с применением языка TypeScript, что обеспечивает компонентный подход и статическую типизацию. Интеграция Яндекс.Карт осуществлялась через официальный JavaScript API версии 2.1, подключаемый не статически в HTML, а динамически посредством создания элемента `script` в компоненте.

3 Результаты и обсуждения

Реализация Яндекс.Карты в компоненте Admission была выполнена с использованием JavaScript API версии 2.1. Основной задачей являлась интеграция интерактивной карты в архитектуру React-приложения с сохранением корректной инициализации, управлением асинхронной загрузкой скрипта и обработкой возможных ошибочных сценариев. Для начала была выполнена декларация типов для глобального объекта `window.ymaps`, что необходимо при работе с TypeScript. Без явного описания данного объекта компилятор генерирует ошибки, поскольку API Яндекс.Карт подключается динамически и отсутствует в типовой среде выполнения TypeScript. Фрагмент декларации глобального интерфейса представлен на рисунке (рис. 1).

```
19 // Декларация типов для Яндекс.Карт
20 declare global {
21   interface Window {
22     ymaps: any;
23   }
24 }
```

Рис. 1. Декларация типов для глобального объекта `window.ymaps`

Следующим этапом была организация состояний, связанных с процессом загрузки карты. В компоненте были определены два состояния: `mapLoaded` – отвечающее за успешную инициализацию карты, и `mapError` – фиксирующее ошибки при загрузке или отрисовке. Использование хуков состояния позволяет компоненту реактивно отображать различные варианты интерфейса в зависимости от текущего статуса загрузки карты. Код инициализации состояний приведён на рисунке (рис. 2).

```

3 // Компонент страницы "Абитуриенту" с информацией о поступлении
9 export default function Admission() {
10   // Состояния для данных
11   const [specialties, setSpecialties] = useState<Specialty[]>([]);
12   const [documents, setDocuments] = useState<RequiredDocument[]>([]);
13   const [dates, setDates] = useState<ImportantDate[]>([]);
14   const [contacts, setContacts] = useState<AdmissionContact[]>([]);
15   const [dormitory, setDormitory] = useState<DormitoryData>({ description: null, address: null, images: [] });
16   const [loading, setLoading] = useState(true);
17   const [mapLoaded, setMapLoaded] = useState(false);
18   const [mapError, setMapError] = useState(false);

```

Рис. 2. Состояния загрузки карты в компоненте Admission

Одним из ключевых элементов реализации стала функция динамической загрузки скрипта API Яндекс.Карт. В отличие от статического подключения в `<script>` внутри HTML, динамическая загрузка позволяет контролировать момент загрузки API, предотвращает повторные подключения и обеспечивает обработку ошибок. В рассматриваемой реализации используется публичный демо-доступ Яндекса без указания ключа. Такой режим позволяет выполнять базовую отрисовку карты, ограниченную по функциональности и количеству запросов. Код функции `loadYandexMapsScript` приведён на рисунке (рис. 3).

```

161 // Функция для динамической загрузки скрипта Яндекс.Карт
162 const loadYandexMapsScript = () => {
163   return new Promise<void>((resolve, reject) => {
164     if (window.ymaps) {
165       resolve();
166       return;
167     }
168
169     const script = document.createElement('script');
170     script.src = 'https://api-maps.yandex.ru/2.1/?lang=ru_RU';
171     script.type = 'text/javascript';
172     script.onload = () => {
173       console.log('Yandex Maps API загружен успешно');
174       resolve();
175     };
176     script.onerror = (error) => {
177       console.error('Ошибка загрузки Yandex Maps API:', error);
178       reject(error);
179     };
180     document.head.appendChild(script);
181   });
182 };

```

Рис. 3. Функция для динамической загрузки JavaScript API Яндекс.Карт

Процесс инициализации карты был реализован в хуке `useEffect`. Логика включает последовательную загрузку скрипта, ожидание готовности DOM-элемента карты и обращение к функции `ymaps.ready`. На данном этапе определяется точка центра карты по координатам г. Биробиджана, создаётся объект карты с элементами управления и добавляется метка с подсказкой и баллуном. Также предусмотрена дополнительная задержка, позволяющая убедиться, что DOM-элемент с `id="map"` успел загрузиться и имеет размеры, достаточные для корректной инициализации. Код инициализации карты представлен на рисунке (рис. 4). При инициализации карты использовалась стандартная структура API Яндекс.Карт: объект `Map` создавался с указанием параметров `center`, `zoom` и набора встроенных `controls`. Координаты геолокации здания в г. Биробиджане были заданы явно. Добавление метки осуществлялось с использованием объекта `Placemark`, которому передавались параметры `balloonContent` и `hintContent`. Эти элементы позволяют пользователю получить дополнительную информацию при наведении и клике.

```

184 // Инициализация Яндекс.Карты при загрузке компонента
185 useEffect(() => {
186   const initMap = async () => {
187     try {
188       // Загружаем скрипт, если не загружен
189       await loadYandexMapsScript();
190
191       if (window.ymaps && !mapLoaded) {
192         // Проверяем, что элемент карты существует и видим
193         const mapElement = document.getElementById('map');
194         if (!mapElement || mapElement.offsetWidth === 0) {
195           // Если элемент не готов, ждем еще немного
196           setTimeout(initMap, 500);
197           return;
198         }
199       }
200
201       window.ymaps.ready(() => {
202         try {
203           // Координаты Биробиджана
204           const birobidzhanCoords = [48.758344, 132.887870];
205
206           // Создаем карту
207           const map = new window.ymaps.Map('map', {
208             center: birobidzhanCoords,
209             zoom: 15,
210             controls: ['zoomControl', 'fullscreenControl']
211           });
212
213           // Добавляем метку техникума
214           const placemark = new window.ymaps.Placemark(birobidzhanCoords, {
215             hintContent: 'Политехнический техникум',
216             balloonContent: 'П. Биробиджан, ул. Техникумовская, д. 15'
217           });
218
219           map.geoObjects.add(placemark);
220           setMapLoaded(true);
221         } catch (error) {
222           console.error('Ошибка инициализации карты:', error);
223           setMapError(true);
224         }
225       });
226     } catch (error) {
227       console.error('Не удалось загрузить Yandex Maps API:', error);
228       setMapError(true);
229     }
230   };

```

Рис. 4. Логика инициализации карты в хуке `useEffect`

Завершающим этапом стала реализация JSX-разметки карты. Контейнер `div` с идентификатором `map` служит целевым элементом для инициализации карты. В процессе загрузки интерфейс отображает анимированный ладер, информирующий пользователя о том, что карта загружается. В случае ошибки отображается `fallback`-блок с информационным сообщением. Такой подход позволяет повысить удобство взаимодействия, обеспечивая корректное информирование пользователя о текущем состоянии компонента. Пример JSX-разметки карты представлен на рисунке (рис. 5).

```

391      {/* Карта */}
392      <div className="bg-white rounded-lg overflow-hidden shadow-lg">
393        <div id="map" className="h-96 w-full">
394          {!mapLoaded && !mapError && (
395            <div className="h-full w-full flex items-center justify-center bg-gray-100">
396              <div className="text-center">
397                <div className="animate-spin rounded-full h-12 w-12 border-b-2 border-blue-600 mx-auto">
398                  <p className="text-gray-600">Загрузка карты...</p>
399                </div>
400              </div>
401            )}
402            {mapError && (
403              <div className="h-full w-full flex items-center justify-center bg-gray-100">
404                <div className="text-center">
405                  <div className="text-gray-400 mb-2">
406                    <svg className="w-16 h-16 mx-auto" fill="none" stroke="currentColor" viewBox="0 0 24">
407                      <path strokeLinecap="round" strokeLinejoin="round" strokeWidth={1} d="M9 20l-5.47-5.47" />
408                    </svg>
409                  </div>
410                  <p className="text-gray-600 font-medium">Карта недоступна</p>
411                  <p className="text-gray-500 text-sm mt-1">Проверьте подключение к интернету</p>
412                </div>
413              </div>
414            )}
415          </div>
416        </div>
417      </div>
418    </div>

```

Рис. 5. JSX-разметка компонента с контейнером карты и обработкой состояний

Работающий интерфейс с отображением карты и установленной меткой представлен на рисунке (рис. 6).

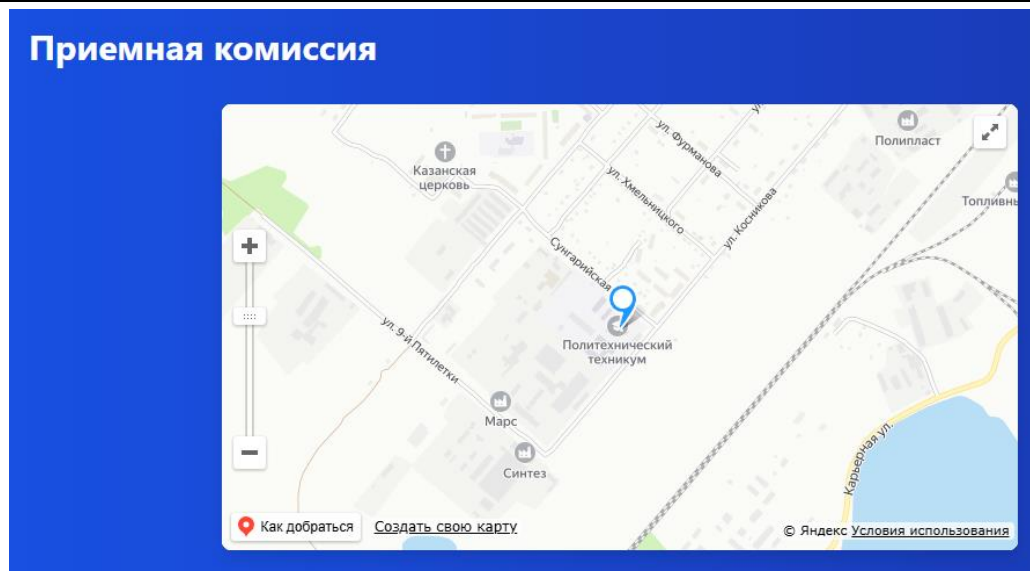


Рис. 6. Итоговый вид карты на сайте

4 Выводы

В ходе работы была реализована интеграция Яндекс.Карт в React/TypeScript-приложение с динамической загрузкой JavaScript API и последующей инициализацией карты в компоненте Admission. Использование хуков состояния и эффектов позволило обеспечить корректную обработку загрузки скрипта и ошибок, а также адаптивное отображение интерфейса в зависимости от состояния карты. Реализация метки и центра карты обеспечила точное отображение местоположения Здания. Проведённое тестирование подтвердило стабильную работу компонента, корректность отображения карты и информативность fallback-интерфейса, что демонстрирует эффективность предложенного подхода и его пригодность для интеграции картографических сервисов в современные веб-приложения.

Библиографический список

1. Север А.А. Использование API карт Google Maps, Yandex.Maps и 2GIS в приложении JavaFX // Научные труды магистрантов и аспирантов. Нижневартонск, 2019. С. 265–268.
2. Филиппов Н.И., Лошаков А.В. Сравнительный анализ картографических сервисов Google Maps и Yandex Maps // Актуальные научные исследования в современном мире. 2021. № 4-3 (72). С. 128–133.
3. Хайруллин Д.А., Богданов М.Р. Изучение использования TypeScript в React-приложениях, преимущества статической типизации, подходы к типизации компонентов, хуков и состояния // Международный журнал гуманитарных и естественных наук. 2024. № 12-3 (99). С. 233–238.
4. Удальцов К.Р. Формализация шаблонов безопасных pull request-ов для React-приложений с TypeScript в командах с Git Flow // Актуальные проблемы социально-гуманитарного и научно-технического знания. 2025. № 4 (44). С. 35–36.

-
5. Воронко Т.М. Создание React-библиотеки с использованием TypeScript // Электронные системы и технологии. Материалы 58-й научной конференции аспирантов, магистрантов и студентов БГУИР. Минск, 2022. С. 200–202.