Асинхронный рендеринг компонентов

Когда стоит подождать

Зуев Дмитрий
Компания Тензор

Асинхронный рендеринг компонентов

Когда стоит подождать

Зуев Дмитрий
Компания Тензор

facebook

2017

Асинхронный рендеринг компонентов

Шаблонизация

$799.99

BUY

ДАННЫЕ

Где многие ошибаются?

ПОЛУЧЕНИЕ ДАННЫХ

Интернет-магазин

$799.99

BUY

Google

manolo blahnic

Все Карты Видео Картинки Новости Ещё Настройки Инструменты

По запросу **manolo blahnic** ничего не найдено.

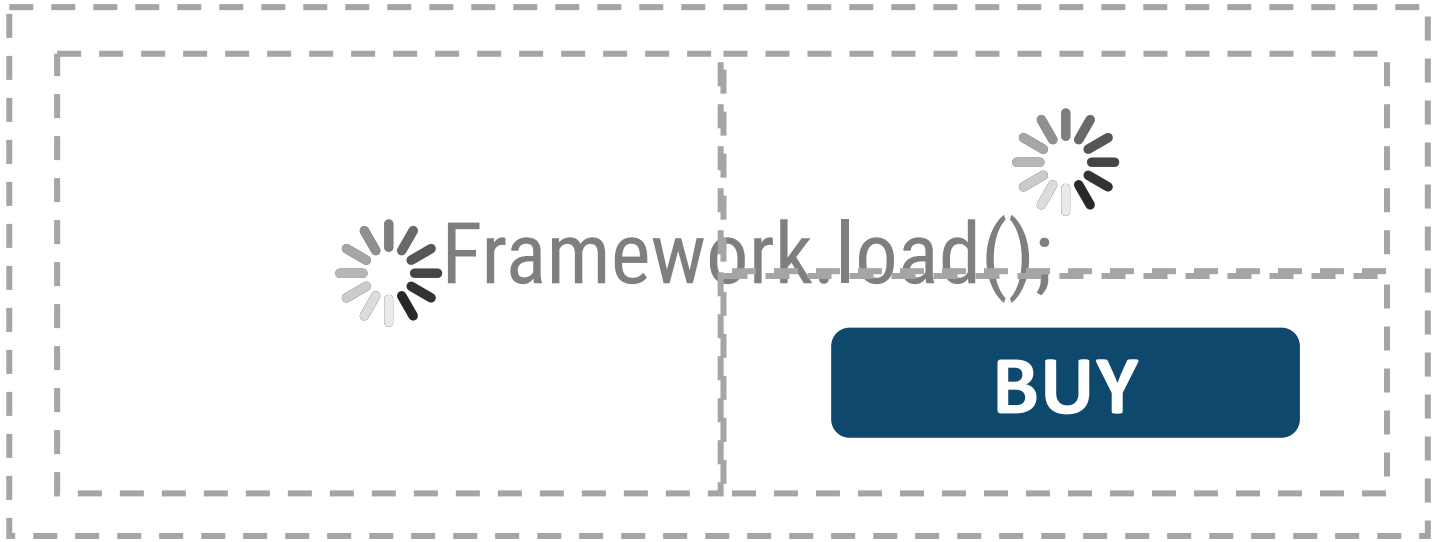
Рекомендации:

- Убедитесь, что все слова написаны без ошибок.
- Попробуйте использовать другие ключевые слова.
- Попробуйте использовать более популярные ключевые слова.

Server Side Rendering

BROWSER CODE ≠ SERVER CODE

Интернет-магазин



facebook®

Flux

Синхронная шаблонизация

$799.99

BUY

Flux

ДАННЫЕ

TEAM 1

TEAM 2

TEAM 3

Flux

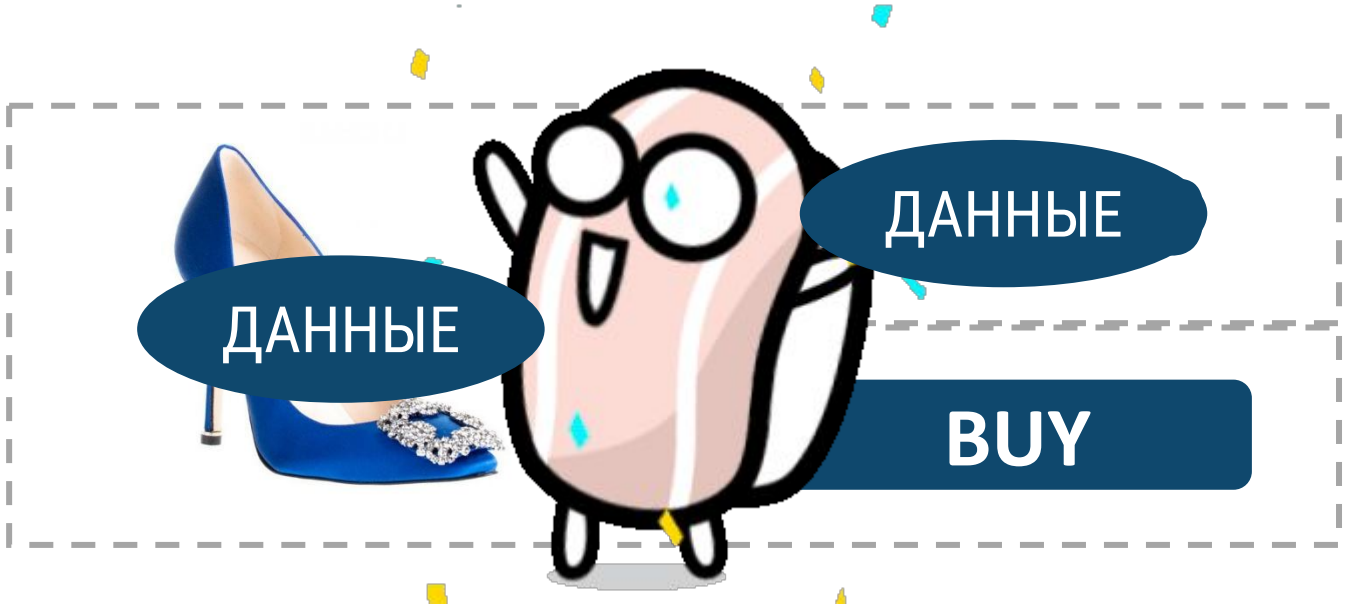
ДАННЫЕ

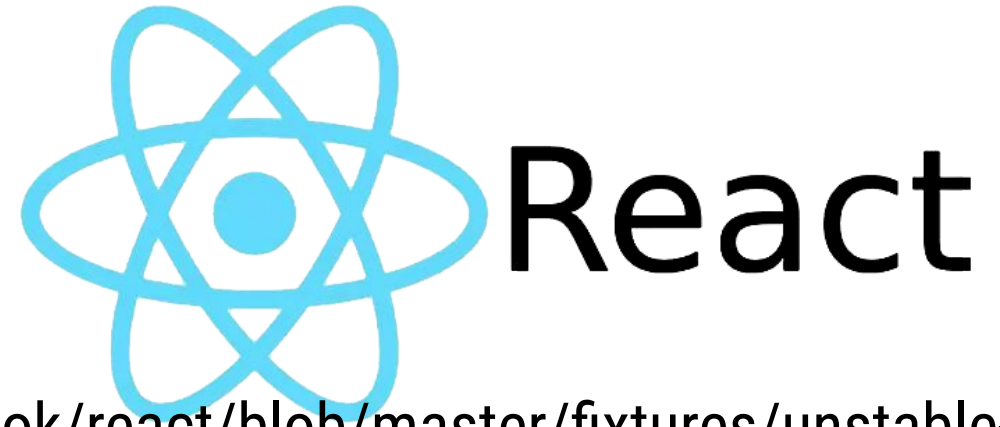
TEAM 4

TEAM 5

TEAM

Асинхронный рендеринг компонентов





https://github.com/facebook/react/blob/master/fixtures/unstable-async/suspense/README.md

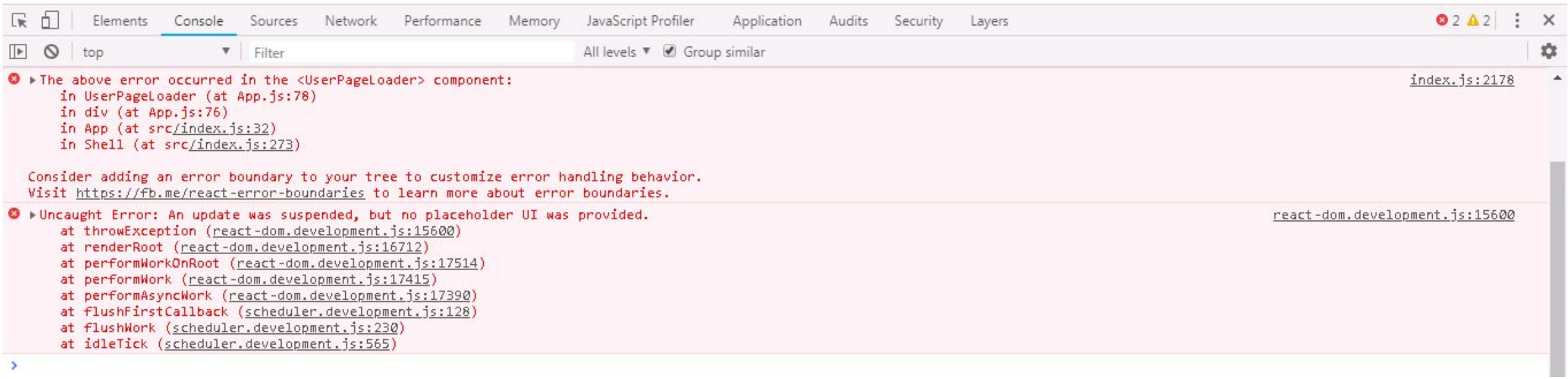
```
renderDetail(id) {
 return (
  <div>
    <Placeholder delayMs={2000} fallback={<Spinner size="large" />}>
      <UserPageLoader id={id} />
    </Placeholder>
  </div>
 );
}
```

```
render() {
  const {currentId, showDetail} = this.state;
  return this.renderList(currentId);
}
```

```
function UserPageLoader(props) {
  const UserPage = UserPageResource.read(cache).default;
  return <UserPage id={props.id} />;
}
```

```
export default function UserPage({id}) {
  return (
    <div>
      <UserDetails id={id} />
      <Placeholder delayMs={1000} fallback={<Spinner size="medium" />}>
        <Repositories id={id} />
      </Placeholder>
    </div>
  );
}
```

```
function UserPageLoader(props) {
  const UserPage = new Promise();
  return <UserPage {...props} />;
}
```


Elements
Console
Sources
Network
Performance
Memory
JavaScript Profiler
Application
Audits
Security
Layers
2 2
top
Filter
All levels
Group similar
The above error occurred in the <UserPageLoader> component:
in UserPageLoader (at App.js:78)
in div (at App.js:76)
in App (at src/index.js:32)
in Shell (at src/index.js:273)
Consider adding an error boundary to your tree to customize error handling behavior.
Visit https://fb.me/react-error-boundaries to learn more about error boundaries.
index.js:2178
Uncaught Error: An update was suspended, but no placeholder UI was provided.
at throwException (react-dom.development.js:15600)
at renderRoot (react-dom.development.js:16712)
at performWorkOnRoot (react-dom.development.js:17514)
at performWork (react-dom.development.js:17415)
at performAsyncWork (react-dom.development.js:17390)
at flushFirstCallback (scheduler.development.js:128)
at flushWork (scheduler.development.js:230)
at idleTick (scheduler.development.js:565)
react-dom.development.js:15600

componentWillMount

componentWillReceiveProps

componentWillUpdate

getDerivedStateFromProps

gaearon commented 28 days ago Member

Don't use `getDerivedStateFromProps` for data fetching. It's not meant for this.

You want to put this logic into `componentDidUpdate`.

```
function UserPageLoader(props) {
  const UserPage =
UserPageResource.read(cache).default;
  return <UserPage {...props} />;
}
```


HA! HA!

Это работает только на клиенте

gaearon commented 26 days ago Member

Client implementation is more or less ready. The cache provider still needs work but the core implementation is already in production at FB.

We're just starting work on the SSR part. So it might come later.

2

Асинхронные компоненты

Иногда бывает удобно разделить крупное приложение на части и подгружать компоненты с сервера только тогда, когда в них возникнет потребность. Для этого Vue позволяет определить компонент как функцию-фабрику, асинхронно возвращающую определение компонента. Vue вызовет фабричную функцию только тогда, когда компонент действительно понадобится, и закэширует результат для дальнейшего использования. Например:

```JS
Vue.component('async-example', function (resolve, reject) {
  setTimeout(function () {
    // Передаём определение компонента в коллбэк resolve
    resolve({
      template: '<div>Я — асинхронный!</div>'
    })
  }, 1000)
})
```

Функция-фабрика принимает параметр `resolve` — коллбэк, который вызывается после того, как определение компонента получено от сервера. Кроме того, можно вызвать `reject(reason)`, если загрузка по какой-либо причине не удалась. Мы используем `setTimeout` исключительно в демонстрационных целях; как именно

Fetching After Navigation

When using this approach, we navigate and render the incoming component immediately, and fetch data in the component's `created` hook. It gives us the opportunity to display a loading state while the data is being fetched over the network, and we can also handle loading differently for each view.

```js
Vue.component('async-webpack-example', function (resolve) {
  // Специальный синтаксис require укажет Webpack
  // автоматически разделить сборку на части
  // для последующей асинхронной загрузки
  require(['./my-async-component'], resolve)
})
```

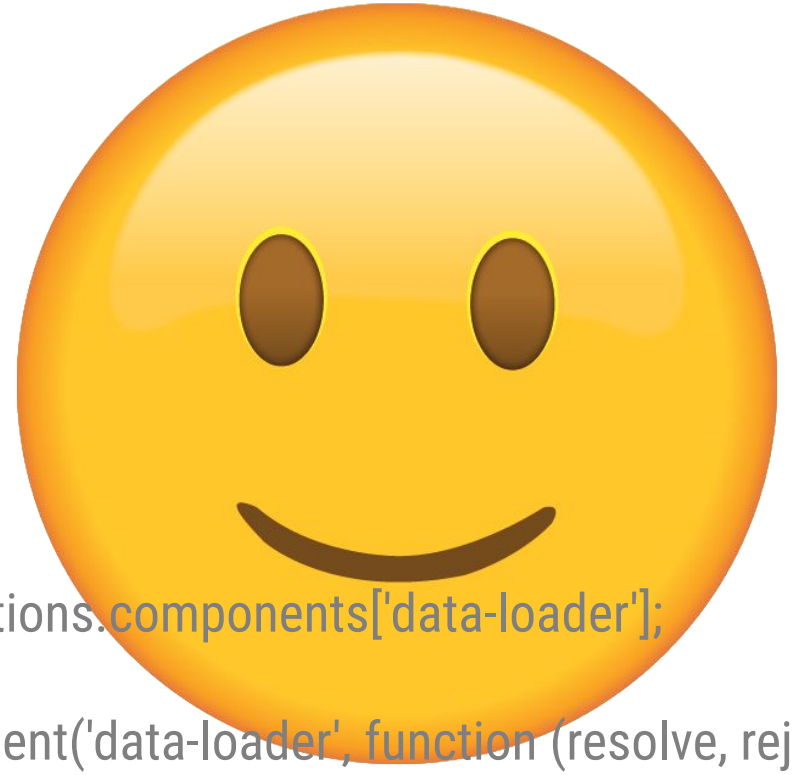
```
Router.setCurrentPage(req.url);
const app = new Vue({
  data: {
    url: req.url
  },
  created: function () {
     this.abc = 1;
  },
  template: `<div>VIPER by Vue <data-loader /></div>`
});


Vue.component('data-loader', function (resolve, reject) {
   Source.getDataFromStore( Router.getCurrentId() ).then( (dataFromBI)=> {
      resolve({
         created: function(){
            this.my = dataFromBI || {};
         },
         template: '<div> {{my.data}} </div>'
      })
   });
});
```

```
getDataFromStore: function(id){
   return new Promise(resolve => {
      setTimeout(()=>{
         resolve(this.storeBank[id]);
      });
   });
}

const Router = {
   ...
   getCurrentId: function(){
      return this.url.split('/')[1];
   }
}

resolve({
  created: function(){
    this.my = dataFromBI || {};
  },
  template: '<div> {{my.data}} </div>'
})
```



```
delete Vue.options.components['data-loader'];

  Vue.component('data-loader', function (resolve, reject) {
    Store.getDataFromStore( Router.url.split('/')[1] ).then( (dataFromBI)=> {
      resolve({…})
    });
  });
```

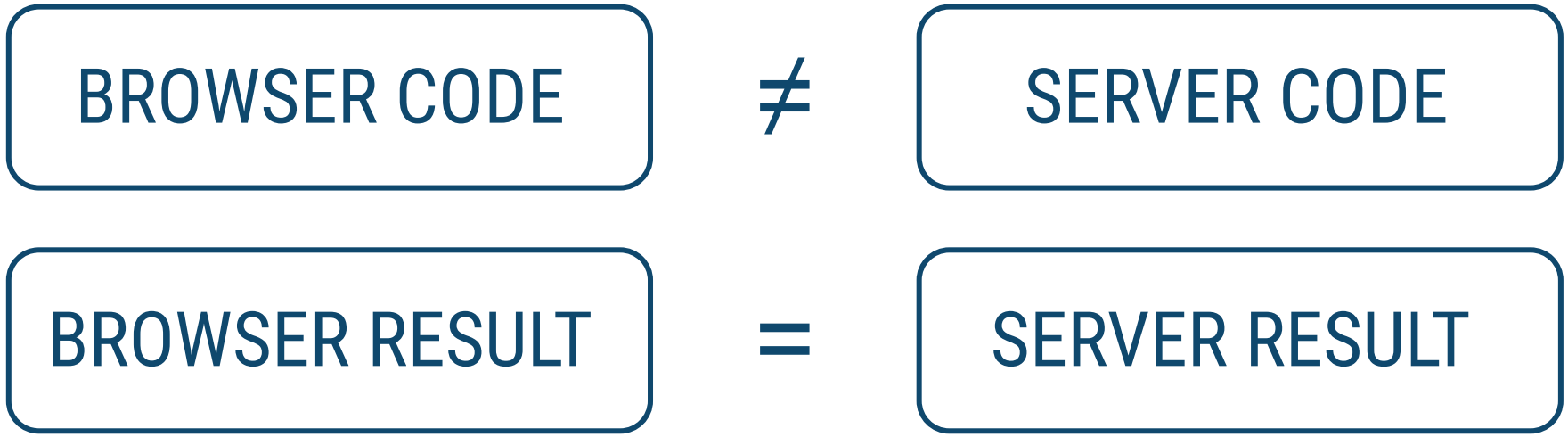
React

Vue.js

```
<Controls.Application>
        <Controls.Router mask="A/:id">
            <MyDataFetcher>
                <MyView>
                </MyView>
            </MyDataFetcher>
        </Controls.Router>
</Controls.Application>
```

```
_beforeMount: function(options){
   return new Promise(resolve => {
      … resolve(data); …
   });
}
```

Both Side Rendering



React

WaSaby

<Controls.Placeholder />

WaSaby + GitHub =  ❤️