

Раздел

[Введение в события](#)

Навигация по уроку

Отмена действия браузера

Опция «passive» для
обработчика

event.defaultPrevented

Итого


Задачи (3)

Комментарии

Поделиться



Редактировать на GitHub

[🏠](#) → [Браузер: документ, события, интерфейсы](#)
→ [Введение в события](#) 5-го февраля 2020

Действия браузера по умолчанию

Многие события автоматически влекут за собой действие браузера.

Например:

- Клик по ссылке инициирует переход на новый URL.
- Нажатие на кнопку «отправить» в форме – отсылку её на сервер.
- Зажатие кнопки мыши над текстом и её движение в таком состоянии – инициирует его выделение.

Если мы обрабатываем событие в JavaScript, то зачастую такое действие браузера нам не нужно. К счастью, его можно отменить.

Отмена действия браузера

Есть два способа отменить действие браузера:

- Основной способ – это воспользоваться объектом `event`. Для отмены действия браузера существует стандартный метод `event.preventDefault()`.
- Если же обработчик назначен через `on<событие>` (не через `addEventListener`), то также можно вернуть `false` из обработчика.

В следующем примере при клике по ссылке переход не произойдёт:

```
1 <a href="/" onclick="return false">Нажми здесь</a>
2 или
3 <a href="/" onclick="event.preventDefault()">здесь</a>
```

[Нажми здесь](#) или [здесь](#)

Возвращать `true` не нужно

Обычно значение, которое возвращает обработчик события, игнорируется.

Единственное исключение – это `return false` из обработчика, назначенного через `on<событие>`.

В других случаях `return` не нужен, он никак не обрабатывается.

Пример: меню

Рассмотрим меню для сайта, например:

```
1 <ul id="menu" class="menu">
2   <li><a href="/html">HTML</a></li>
3   <li><a href="/javascript">JavaScript</a></li>
4   <li><a href="/css">CSS</a></li>
5 </ul>
```

Данный пример при помощи CSS может выглядеть так:

HTML

JavaScript

CSS



Раздел

[Введение в события](#)

Навигация по уроку

Отмена действия браузера

Опция «passive» для
обработчика

event.defaultPrevented

Итого

Задачи (3)

Комментарии

Поделиться



[Редактировать на GitHub](#)



В HTML-разметке все элементы меню являются не кнопками, а ссылками, то есть тегами `<a>`. В этом подходе есть некоторые преимущества, например:

- Некоторые посетители очень любят сочетание «правый клик – открыть в новом окне». Если мы будем использовать `<button>` или ``, то данное сочетание работать не будет.
- Поисковые движки переходят по ссылкам `` при индексации.

Поэтому в разметке мы используем `<a>`. Но нам необходимо обрабатывать клики в JavaScript, а стандартное действие браузера (переход по ссылке) – отменить.

Например, вот так:

```
1 menu.onclick = function(event) {
2   if (event.target.nodeName !== 'A') return;
3
4   let href = event.target.getAttribute('href');
5   alert( href ); // может быть подгрузка с сервера, ген
6
7   return false; // отменить действие браузера (переход
8 };
```

Если мы уберём `return false`, то после выполнения обработчика события браузер выполнит «действие по умолчанию» – переход по адресу из `href`. А это нам здесь не нужно, мы обрабатываем клик сами.

Кстати, использование здесь делегирования событий делает наше меню очень гибким. Мы можем добавить вложенные списки и стилизовать их с помощью CSS – обработчик не потребует изменений.

События, вытекающие из других

Некоторые события естественным образом вытекают друг из друга. Если мы отменим первое событие, то последующие не возникнут.

Например, событие `mousedown` для поля `<input>` приводит к фокусировке на нём и запускает событие `focus`. Если мы отменим событие `mousedown`, то фокусирования не произойдёт.

В следующем примере попробуйте нажать на первом `<input>` – происходит событие `focus`. Но если вы нажимаете по второму элементу, то события `focus` не будет.

```
1 <input value="Фокус работает" onfocus="this.value=
2 <input onmousedown="return false" onfocus="this.v
```

Это потому, что отменено стандартное действие `mousedown`. Впрочем, фокусировка на элементе всё ещё возможна, если мы будем использовать другой способ. Например, нажатием клавиши `Tab` можно перейти от первого поля ввода ко второму. Но только не через клик мышью на элемент, это больше не работает.

Опция «passive» для обработчика

Необязательная опция `passive: true` для `addEventListener` сигнализирует браузеру, что обработчик не собирается выполнять `preventDefault()`.

Почему это может быть полезно?

Есть некоторые события, как `touchmove` на мобильных устройствах (когда пользователь перемещает палец по экрану), которое по умолчанию начинает прокрутку, но мы можем отменить это действие, используя `preventDefault()` в обработчике.

Раздел

[Введение в события](#)

Навигация по уроку

Отмена действия браузера

Опция «passive» для обработчика

event.defaultPrevented

Итого

Задачи (3)

Комментарии

Поделиться



[Редактировать на GitHub](#)



Поэтому, когда браузер обнаружит такое событие, он должен для начала запустить все обработчики и после, если `preventDefault` не вызывается нигде, он может начать прокрутку. Это может вызвать ненужные задержки в пользовательском интерфейсе.

Опция `passive: true` сообщает браузеру, что обработчик не собирается отменять прокрутку. Тогда браузер начинает её немедленно, обеспечивая максимально плавный интерфейс, параллельно обрабатывая событие.

Для некоторых браузеров (Firefox, Chrome) опция `passive` по умолчанию включена в `true` для таких событий, как `touchstart` и `touchmove`.

event.defaultPrevented

Свойство `event.defaultPrevented` установлено в `true`, если действие по умолчанию было предотвращено, и `false`, если нет.

Рассмотрим практическое применение этого свойства для улучшения архитектуры.

Помните, в главе [Всплытие и погружение](#) мы говорили о `event.stopPropagation()` и упоминали, что останавливать «всплытие» — плохо?

Иногда вместо этого мы можем использовать `event.defaultPrevented`, чтобы просигнализировать другим обработчикам, что событие обработано.

Давайте посмотрим практический пример.

По умолчанию браузер при событии `contextmenu` (клик правой кнопкой мыши) показывает контекстное меню со стандартными опциями. Мы можем отменить событие по умолчанию и показать своё меню, как здесь:

```
1 <button>Правый клик вызывает контекстное меню браузера</button>
2
3 <button oncontextmenu="alert('Рисую наше меню'); return false">
4   Правый клик вызывает наше контекстное меню
5 </button>
```

Правый клик вызывает контекстное меню браузера

Правый клик вызывает наше контекстное меню

Теперь в дополнение к этому контекстному меню реализуем контекстное меню для всего документа.

При правом клике должно показываться ближайшее контекстное меню.

```
1 <p>Правый клик здесь вызывает контекстное меню документа</p>
2 <button id="elem">Правый клик здесь вызывает контекстное меню кнопки</button>
3
4 <script>
5   elem.oncontextmenu = function(event) {
6     event.preventDefault();
7     alert("Контекстное меню кнопки");
8   };
9
10  document.oncontextmenu = function(event) {
11    event.preventDefault();
12    alert("Контекстное меню документа");
13  };
14 </script>
```

Правый клик здесь вызывает контекстное меню документа

Правый клик здесь вызывает контекстное меню кнопки

Проблема заключается в том, что когда мы кликаем по элементу `elem`, то мы получаем два меню: контекстное меню для кнопки и (событие всплывает вверх) контекстное меню для документа.

Как это поправить? Одно из решений — это подумать: «Когда мы обрабатываем правый клик в обработчике на кнопке, остановим всплытие», и вызвать `event.stopPropagation()`:

Раздел

[Введение в события](#)

Навигация по уроку

Отмена действия браузера

Опция «passive» для обработчика

event.defaultPrevented

Итого

Задачи (3)

Комментарии

Поделиться



[Редактировать на GitHub](#)



```
1 <p>Правый клик вызывает меню документа</p>
2 <button id="elem">Правый клик вызывает меню кнопки (доб.
3
4 <script>
5   elem.oncontextmenu = function(event) {
6     event.preventDefault();
7     event.stopPropagation();
8     alert("Контекстное меню кнопки");
9   };
10
11   document.oncontextmenu = function(event) {
12     event.preventDefault();
13     alert("Контекстное меню документа");
14   };
15 </script>
```

Правый клик вызывает меню документа

Правый клик вызывает меню кнопки (добавлен event.stopPropagation)

Теперь контекстное меню для кнопки работает как задумано. Но цена слишком высока. Мы навсегда запретили доступ к информации о правых кликах для любого внешнего кода, включая счётчики, которые могли бы собирать статистику, и т.п. Это слегка неразумно.

Альтернативным решением было бы проверить в обработчике document, было ли отменено действие по умолчанию? Если да, тогда событие было обработано, и нам не нужно на него реагировать.

```
1 <p>Правый клик вызывает меню документа (добавлена проверка</p>
2 <button id="elem">Правый клик вызывает меню кнопки</but
3
4 <script>
5   elem.oncontextmenu = function(event) {
6     event.preventDefault();
7     alert("Контекстное меню кнопки");
8   };
9
10   document.oncontextmenu = function(event) {
11     if (event.defaultPrevented) return;
12
13     event.preventDefault();
14     alert("Контекстное меню документа");
15   };
16 </script>
```

Правый клик вызывает меню документа (добавлена проверка event.defaultPrevented)

Правый клик вызывает меню кнопки

Сейчас всё работает правильно. Если у нас есть вложенные элементы и каждый из них имеет контекстное меню, то код также будет работать. Просто убедитесь, что проверяете event.defaultPrevented в каждом обработчике contextmenu.

event.stopPropagation() и event.preventDefault()

Как мы можем видеть, event.stopPropagation() и event.preventDefault() (также известный как return false) – это две разные функции. Они никак не связаны друг с другом.

Раздел

Введение в события

Навигация по уроку

Отмена действия браузера

Опция «passive» для обработчика

event.defaultPrevented

Итого

Задачи (3)

Комментарии

Поделиться



Редактировать на GitHub



Архитектура вложенных контекстных меню

Есть также несколько альтернативных путей, чтобы реализовать вложенные контекстные меню. Одним из них является единый глобальный объект с обработчиком `document.oncontextmenu` и методами, позволяющими хранить в нём другие обработчики.

Объект будет перехватывать любой клик правой кнопкой мыши, просматривать сохранённые обработчики и запускать соответствующий.

Но при этом каждый фрагмент кода, которому требуется контекстное меню, должен знать об этом объекте и использовать его вместо собственного обработчика `contextmenu`.

Итого

Действий браузера по умолчанию достаточно много:

- `mousedown` – начинает выделять текст (если двигать мышкой).
- `click` на `<input type="checkbox">` – ставит или убирает галочку в `input`.
- `submit` – при нажатии на `<input type="submit">` или при нажатии клавиши `Enter` в форме данные отправляются на сервер.
- `keydown` – при нажатии клавиши в поле ввода появляется символ.
- `contextmenu` – при правом клике показывается контекстное меню браузера.
- ...и многие другие...

Все эти действия можно отменить, если мы хотим обработать событие исключительно при помощи JavaScript.

Чтобы отменить действие браузера по умолчанию, используйте `event.preventDefault()` или `return false`. Второй метод работает, только если обработчик назначен через `on<событие>`.

Опция `passive: true` для `addEventListener` сообщает браузеру, что действие по умолчанию не будет отменено. Это очень полезно для некоторых событий на мобильных устройствах, таких как `touchstart` и `touchmove`, чтобы сообщить браузеру, что он не должен ожидать выполнения всех обработчиков, а ему следует сразу приступить к выполнению действия по умолчанию, например, к прокрутке.

Если событие по умолчанию отменено, то значение `event.defaultPrevented` становится `true`, иначе `false`.

Сохраняйте семантику, не злоупотребляйте

Технически, отменяя действия браузера по умолчанию и добавляя JavaScript, мы можем настроить поведение любого элемента. Например, мы можем заставить ссылку `<a>` работать как кнопку, а кнопку `<button>` вести себя как ссылка (перенаправлять на другой URL).

Но нам следует сохранять семантическое значение HTML элементов. Например, не кнопки, а тег `<a>` должен применяться для переходов по ссылкам.

Помимо того, что это «хорошо», это делает ваш HTML лучше с точки зрения доступности для людей с ограниченными возможностями и с особых устройств.

Также, если мы рассматриваем пример с тегом `<a>`, то обратите внимание: браузер предоставляет возможность открывать ссылки в новом окне (кликая правой кнопкой мыши или используя другие возможности). И пользователям это нравится. Но если мы заменим ссылку кнопкой и стилизуем её как ссылку, используя CSS, то специфичные функции браузера для тега `<a>` всё равно работать не будут.

✓ Задачи

Раздел

[Введение в события](#)

Навигация по уроку

Отмена действия браузера

Опция «passive» для
обработчика

event.defaultPrevented

Итого

Задачи (3)

Комментарии

Поделиться



[Редактировать на GitHub](#)



Почему не работает return false?

важность: 3

Почему в коде ниже `return false` не работает?

```
1 <script>
2   function handler() {
3     alert( "..." );
4     return false;
5   }
6 </script>
7
8 <a href="http://w3.org" onclick="handler()">браузер отк
```

[браузер откроет w3.org](#)

Браузер переходит по указанной ссылке, но нам этого не нужно.

Как поправить?

решение

Поймите переход по ссылке

важность: 5

Сделайте так, чтобы при клике на ссылки внутри элемента `id="contents"` пользователю выводился вопрос о том, действительно ли он хочет покинуть страницу, и если он не хочет, то прерывать переход по ссылке.

Так это должно работать:

```
#contents

Как насчёт того, чтобы прочитать Википедию или посетить W3.org и
узнать о современных стандартах?
```

Детали:

- Содержимое `#contents` может быть загружено динамически и присвоено при помощи `innerHTML`. Так что найти все ссылки и поставить на них обработчики нельзя. Используйте делегирование.
- Содержимое может иметь вложенные теги, в том числе *внутри ссылок*, например, `<i>...</i>`.

[Открыть песочницу для задачи.](#)

решение

Галерея изображений

важность: 5

Создайте галерею изображений, в которой основное изображение изменяется при клике на уменьшенный вариант.

Например:

Раздел

[Введение в события](#)

Навигация по уроку

Отмена действия браузера

Опция «passive» для
обработчика

event.defaultPrevented

Итого

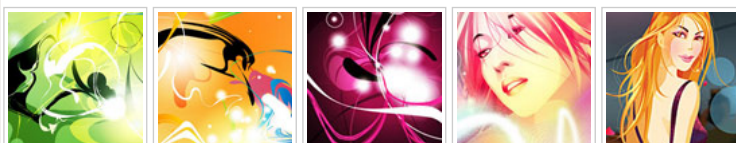
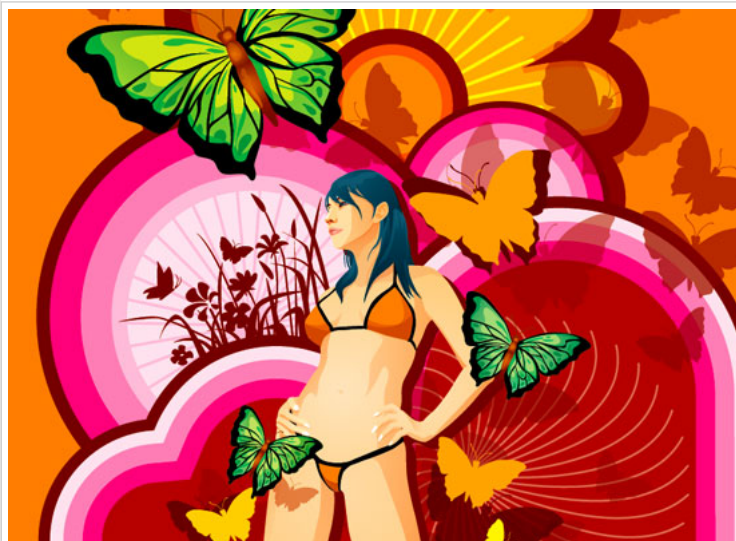
Задачи (3)

Комментарии

Поделиться



[Редактировать на GitHub](#)



P.S. Используйте делегирование.

[Открыть песочницу для задачи.](#)

решение



Проводим [курсы по JavaScript и фреймворкам.](#)



Комментарии

перед тем как писать...