Интерфейсные события

Основные события мыши

Основные события мыши

Простые события

`mousedown/mouseup`

Кнопка мыши нажата/отпущена над элементом.

`mouseover/mouseout`

Курсор мыши появляется над элементом и уходит с него.

`mousemove`

Каждое движение мыши над элементом генерирует это событие.

`contextmenu`

Вызывается при попытке открытия контекстного меню, как правило, нажатием правой кнопки мыши. Но, заметим, это не совсем событие мыши, оно может вызываться и специальной клавишей клавиатуры.

Основные события мыши

Комплексные события

`click`

Вызывается при `mousedown` , а затем `mouseup` над одним и тем же элементом, если использовалась левая кнопка мыши.

`dblclick`

Вызывается двойным кликом на элементе.

Комплексные события состоят из простых, поэтому в теории мы могли бы без них обойтись. Но хорошо, что они существуют, потому что работать с ними очень удобно.

Основные события мыши

Получение информации о кнопке: which

События, связанные с кликом, всегда имеют свойство `which`, которое позволяет определить нажатую кнопку мыши.

Это свойство не используется для событий `click` и `contextmenu`, поскольку первое происходит только при нажатии левой кнопкой мыши, а второе – правой.

Но если мы отслеживаем `mousedown` и `mouseup`, то оно нам нужно, потому что эти события срабатывают на любой кнопке, и `which` позволяет различать между собой «нажатие правой кнопки» и «нажатие левой кнопки».

Есть три возможных значения:

- `event.which == 1` – левая кнопка
- `event.which == 2` – средняя кнопка
- `event.which == 3` – правая кнопка

Основные события мыши

Модификаторы: shift, alt, ctrl и meta

Все события мыши включают в себя информацию о нажатых клавишах-модификаторах.

Свойства объекта события:

- `shiftKey`: `Shift`
- `altKey`: `Alt` (или `Opt` для Mac)
- `ctrlKey`: `Ctrl`
- `metaKey`: `Cmd` для Mac

Они равны `true`, если во время события была нажата соответствующая клавиша.

Например, кнопка внизу работает только при комбинации `Alt+Shift+`клик:

Основные события мыши

Итого

События мыши имеют следующие свойства:

- Кнопка: `which`.

- Клавиши-модификаторы (`true` если нажаты): `altKey`, `ctrlKey`, `shiftKey` и `metaKey` (Mac).
  - Если вы планируете обработать `Ctrl`, то не забудьте, что пользователи Mac обычно используют `Cmd`, поэтому лучше проверить `if (e.metaKey || e.ctrlKey)`.
- Координаты относительно окна: `clientX/clientY`.

- Координаты относительно документа: `pageX/pageY`.

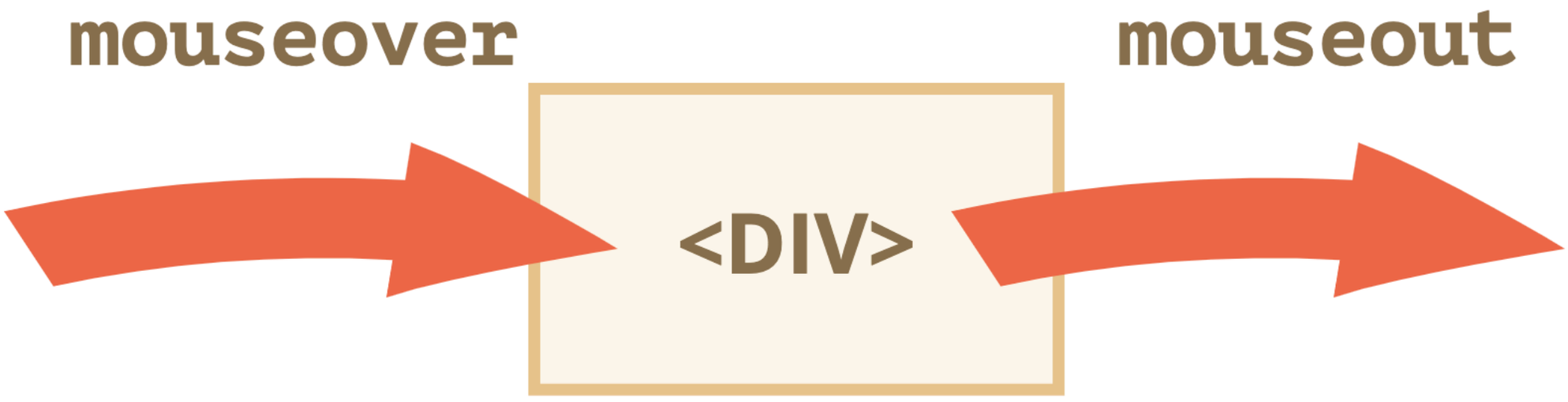
Действие по умолчанию события `mousedown` – начало выделения, если в интерфейсе оно скорее мешает, его можно отменить.

В следующей главе мы поговорим о событиях, которые возникают при передвижении мыши, и об отслеживании смены элементов под указателем.

Движение мыши:
mouseover/out, mouseenter/leave

Движение мыши: mouseover/out, mouseenter/leave

Событие mouseover происходит в момент, когда курсор оказывается над элементом, а событие mouseout – в момент, когда курсор уходит с элемента.



Движение мыши: mouseover/out, mouseenter/leave

Эти события являются особенными, потому что у них имеется свойство `relatedTarget`. Оно «дополняет» `target`. Когда мышь переходит с одного элемента на другой, то один из них будет `target`, а другой `relatedTarget`.

Для события `mouseover`:

- `event.target` – это элемент, *на который* курсор перешёл.
- `event.relatedTarget` – это элемент, *с которого* курсор ушёл (`relatedTarget` → `target`).

Для события `mouseout` наоборот:

- `event.target` – это элемент, *с которого* курсор ушёл.
- `event.relatedTarget` – это элемент, *на который* курсор перешёл (`target` → `relatedTarget`).

Рассмотрим пример:

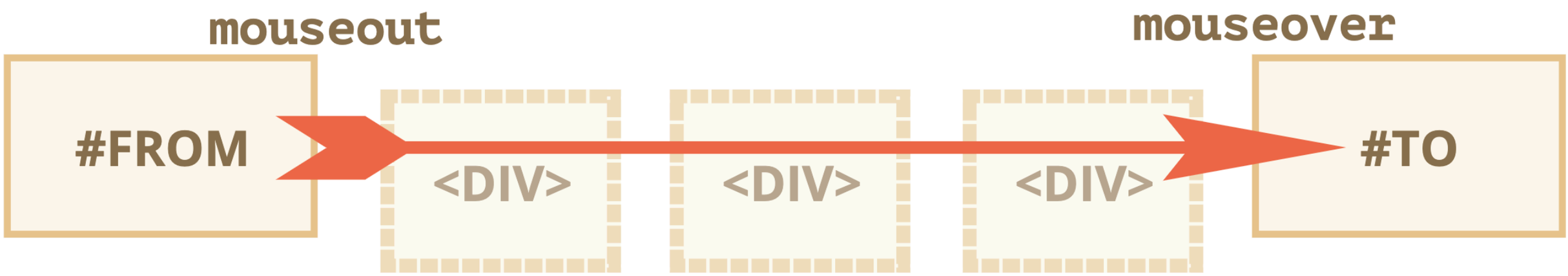
⚠️ Свойство `relatedTarget` может быть `null`

Свойство `relatedTarget` может быть `null`.

Это нормально и означает, что указатель мыши перешёл не с другого элемента, а из-за пределов окна браузера. Или же, наоборот, ушёл за пределы окна.

Следует держать в уме такую возможность при использовании `event.relatedTarget` в своём коде. Если, например, написать `event.relatedTarget.tagName`, то при отсутствии `event.relatedTarget` будет ошибка.

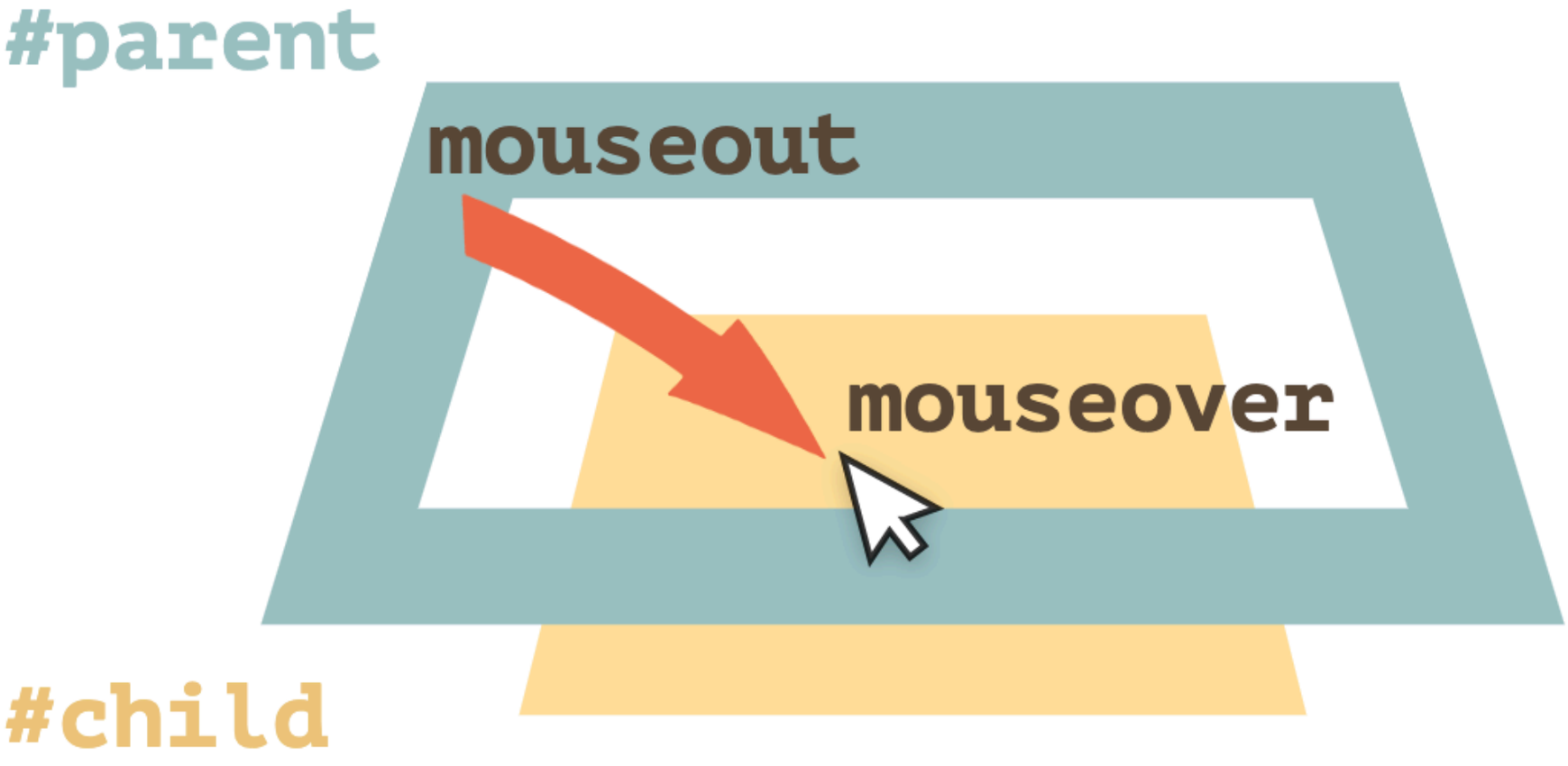
Пропуск элементов



Если курсор мыши передвинуть очень быстро с элемента `#FROM` на элемент `#TO`, как это показано выше, то лежащие между ними элементы `<div>` (или некоторые из них) могут быть пропущены. Событие `mouseout` может запуститься на элементе `#FROM` и затем сразу же сгенерируется `mouseover` на элементе `#TO`.

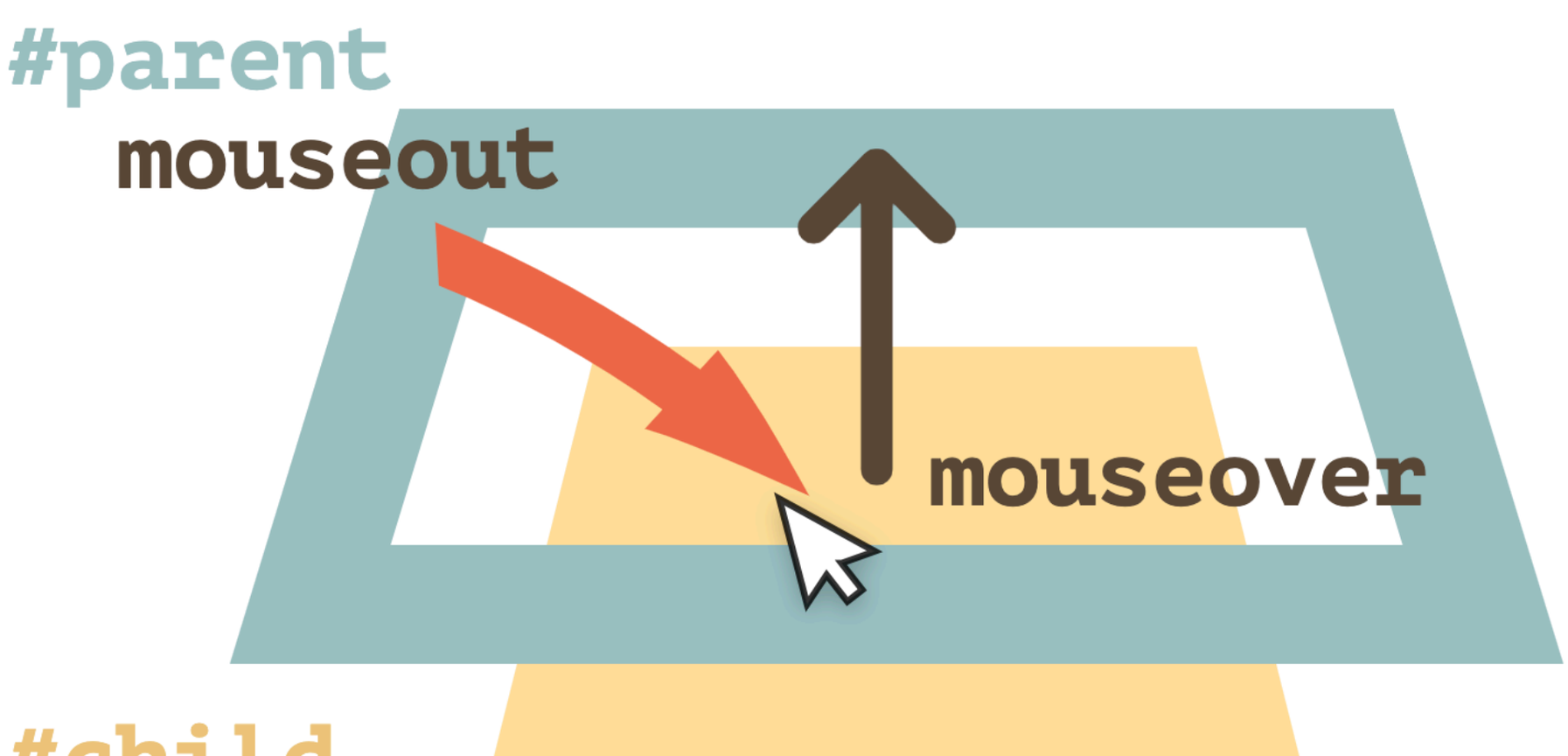
Событие mouseout при переходе на потомка

Важная особенность события `mouseout` – оно генерируется в том числе, когда указатель переходит с элемента на его потомка.



Событие mouseout при переходе на потомка

Событие `mouseover`, происходящее на потомке, всплывает. Поэтому если на родительском элементе есть такой обработчик, то оно его вызовет.



Событие mouseenter и mouseleave

События `mouseenter/mouseleave` похожи на `mouseover/mouseout`.

Они тоже генерируются, когда курсор мыши переходит на элемент или покидает его.

Но есть и пара важных отличий:

1. Переходы внутри элемента, на его потомки и с них, не считаются.
2. События `mouseenter/mouseleave` не всплывают.

Делегирование событий

События `mouseenter/leave` просты и легки в использовании. Но они не всплывают. Таким образом, мы не можем их делегировать.

Drag & Drop

Алгоритм Drag’n’Drop

Базовый алгоритм Drag’n’Drop выглядит так:

1. При `mousedown` – готовим элемент к перемещению, если необходимо (например, создаём его копию).
2. Затем при `mousemove` передвигаем элемент на новые координаты путём смены `left/top` и `position:absolute`.
3. При `mouseup` – остановить перенос элемента и произвести все действия, связанные с окончанием Drag’n’Drop.

События keydown и keyup

Свойство `key` объекта события позволяет получить символ, а свойство `code` – «физический код клавиши».

К примеру, одну и ту же клавишу `Z` можно нажать с клавишей `Shift` и без неё. В результате получится два разных символа: `z` в нижнем регистре и `Z` в верхнем регистре.

Свойство `event.key` – это непосредственно символ, и он может различаться. Но `event.code` всегда будет тот же:

Клавиша	**`event.key`**	**`event.code`**
`Z`	`z` (нижний регистр)	`KeyZ`
`Shift+Z`	`Z` (Верхний регистр)	`KeyZ`

Всем спасибо