



НИ Е ВЯРВАМЕ ВЪВ ВАШЕТО БЪДЕЩЕ

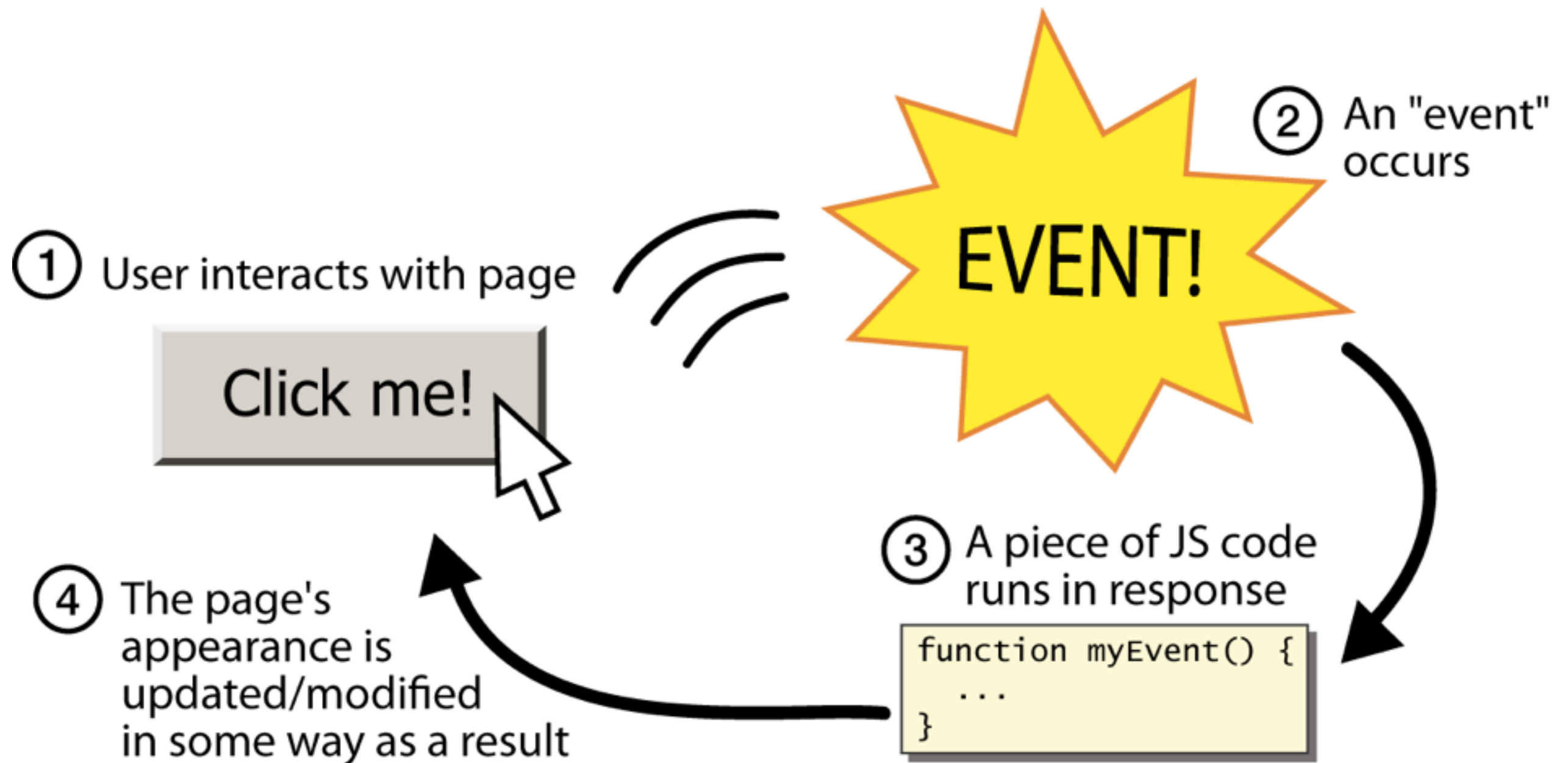
# Събития





# И така .. бърз преговор :)

- Кое от следните е събитие?
  - звъни ми телефона —> **Да**
  - страницата зарежда бавно —> **не е събитие, а състояние**
  - компютъра ми забива —> **Да**
  - няма кафе (даден артикул е изчерпан)  
\\ —> **това е state**



# Събития

- Събитията настъпват в момента, в който нещо премине от едно състояние в друго
- Събитията възникват, а **eventListener**-ите ги регистрират
- Използваме ги, за да следим какво се случва и да зададем съответна реакция
- Реакцията ни се нарича обработка на събитието
- По подробно - в лекция 11:  
<http://zenlabs.pro/courses/lessons/lesson11/lesson.pdf>



# Пример:

click counter

И сега един бърз преговор от лекция 14...



Какво беше DOM?



# DOMContentLoaded

- Това е първото събитие, което настъпва изобщо
- Възниква в момента, в който DOM-а е зареден в `document` обекта и в който браузъра започва да визуализира страницата
- Това събитие е важно за скриптове, които не са разположени в края на `html`-а и се изпълняват преди DOM-а да се зареди
- Например, ако имаме скрипт, който трябва да генерира съдържание, то той ползва `appendChild` метода, за да закачи генерираното съдържание към елемент от DOM-а
- Обаче... Преди DOM-а да е готов, това няма как да стане.



# Как се ползва:

```
document.addEventListener('DOMContentLoaded', function() {  
    console.log('the DOM is ready!');  
});
```

**callback**

или с jQuery:

```
$(document).ready(function() {  
    console.log('the DOM is ready!');  
});
```

# window onload

- Това е BOM loaded събитието и настъпва след като се зареди абсолютно всичко на страницата (много неща продължават да се даунлоад-ват и да се парсват дори и след зареждането на DOM-а, като например скриптовете в края на html-а)
- Т.е. window onload се случва след DOMContentLoaded
- Освен това, когато се случи това събитие, то изпълнява функцията `window.onload`:

```
window.onload = function() {  
    console.log("The BOM is ready");  
};
```

- <https://developer.mozilla.org/en-US/docs/Web/API/GlobalEventHandlers/onload>



# addEventListener()

- За да регистрираме и обработим събитие, използваме `addEventListener()` метода на DOM обектите
- СИНТАКСИС:  
`document.addEventListener('DOMContentLoaded', callback);`
- където `callback` е референция към функция или анонимна функция
- тази функция се нарича обработваща (или `handler`) и приема като първи аргумент `event` обекта, който носи информация за текущия `event` (текущото събитие)

# Event обекта

- При възникване на събитие, се създава нов обект от класа Event, който съдържа в себе си информацията за събитието
- Важното, което трябва да знаем за event обекта е:
  - винаги се получава като първи аргумент в обработващата функция (handler-a)
  - носи информация за текущото събитие
  - може да се използва за отказване на стандартната обработка (тази по подразбиране) на събитието, чрез метода **preventDefault()**
- [http://www.w3schools.com/jsref/dom\\_obj\\_event.asp](http://www.w3schools.com/jsref/dom_obj_event.asp)



# target на събитието

- Елементът, чрез който е прихванато събитието (този на който е закачен `eventListener` за съответното събитие), се нарича **target** на събитието
- Този `target` може да се получи от `event` аргумента на `handler` функцията по следния начин: **`event.target`**

- Пример:

```
$("#button").click(function(event) {  
  console.log('here I am:', event.target); // => "here I am:" "<button>click me!</button>"  
});
```

- <https://developer.mozilla.org/en-US/docs/Web/API/Event/target>

# Примери



**addEventListener... No ..prevent! No ...  
AAAAAAGH!**



# Въпроси?

# JQuery events

- Всичко е на едно място и става по един начин
- cross-device & cross-browser (почти)
- бави зареждането
- но пък ускорява разработката и предпазва от cross-browser проблеми
- <https://api.jquery.com/category/events/>



# JQuery events

- `click()`
- `focus()`
- `keyup()`, `keydown()`, `keypress()`
- `mouseover()`, `mouseout()`
- `blur()`
- Drag & Drop example

# Въпроси?



# Полезни връзки

- MDN event docs:
  - <https://developer.mozilla.org/en-US/docs/Web/Events>
  - <https://developer.mozilla.org/en-US/docs/Web/API/Event>
  - <https://developer.mozilla.org/en-US/docs/Web/API/EventTarget/addEventListener>
- Code pens:
  - <http://codepen.io/jenie/pen/vKBxBz>
  - <http://codepen.io/jenie/pen/qNWojq?editors=1011>
  - <http://codepen.io/jenie/pen/oLvZLy?editors=1010>



**KEEP  
CALM  
AND  
LEARN  
JAVASCRIPT**



# Примери

<http://zenlabs.pro/courses/lessons/lesson17/examples.zip>

# Домашно

<https://github.com/zzeni/swift-academy-homeworks/blob/fe-03/tasks/L17>