SWIFTACADEMY

НИЕ ВЯРВАМЕ ВЪВ ВАШЕТО БЪДЕЩЕ

Събития

И така .. бърз преговор :)

- Кое от следните е събитие?
  - спира тока
  - някой ми е изял сандвича
  - търся си някой, за да му разкажа всичките си проблеми
  - натискам бутон

DOMContentLoaded

- Браузърът винаги бърза да зареди всичко, което е в head-а
- По тази причина изпълнява скрипта/скриптовете, които са там още преди да зареди страницата
- Пак по тази причина, измисляме най-различни варианти как да заредим скрипта по-късно (за да ускорим зареждането на страницата и да позволим на скрипта да използва DOM-а)
- Най-изпитаният и сигурен начин, е да словим скрипта (скриптовете) в края на страницата, точно преди ‘</body>’

window.onload

- Както и да ползваме събития които показват кога всичко се е заредило (скриптове, картинки и т.н.):

```
window.onload = function() {
  init();
  doSomethingElse();
};
```

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

Отложено зареждане

- За външни сриптове, можем да ползваме 2 атрибута, които отлагат зареждането: defer и async

- Въпреки, че имат много удобно приложение, не се препоръчват, защото се поддържат от малко браузъри (>IE12)
  - повече може да прочетете тук:
    - http://peter.sh/experiments/asynchronous-and-deferred-javascript-execution-explained/

addEventListener() и dispatchEvent()

- За да регистрираме и обработим събитие, използваме **addEventListener()** метода на DOM обектите

- синтаксис:
  `document.addEventListener('DOMContentLoaded', funcRef);`

- където funcRef референция към функция или анонимна функция, т.е.:

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

- тази функция се нарича обработваща (или handler)

- За да предизвикаме събитие, използваме **dispatchEvent()**

Event обекта

- При възникване на събитие, то се присвоява на обект от класа Event
- не ни трябва да знаем всичко за този клас. Това, което трябва да знаем за ивента са точно 2 неща:
  - винаги се подава като параметър на обработващата функция (handler-a)
  - има си дефолтен handler (функция за обработка по подразбиране), който винаги се изпълнява при възникването на ивента, независимо дали ние сме го хендълнали или не
- За да откажем изпълнението на дифолтния хендлър използваме метода preventDefault():

```
e.preventDefault()
```

Примери

addEvent!.. No ..dispatch! 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/

Bootstrap.js

- http://getbootstrap.com/javascript/
- tabs
- modals
- tooltips
- scrollspy
- carousel

Други JQuery плъгини

- https://plugins.jquery.com/
- http://www.appelsiini.net/projects/lazyload
- http://plugins.compzets.com/animatescroll/
- http://sorgalla.com/jcarousel/
- http://oridomi.com/

Въпроси?

Примери

http://zenifytheweb.com/courses/lessons/lesson13/example/

Домашно

Да се направи таб навигация или scrollspy навигация, като се използва някой от разгледаните методи (pure JS, jQuery, jQueryUI, jQuery plugins, Bootstrap.js)

https://github.com/zzeni/swift-academy-homeworks/tree/master/tasks/L13