JavaScript. Уровень 2. Расширенные возможности

Модуль 5. События

- Наборы и модели событий
- Назначение обработчиков
- Получение ссылки на событие
- Отмена действия по умолчанию
- Отмена всплытия события
- Делегирование
- Собственные события

События

- JavaScript использует асинхронную и управляемую событиями программную модель
- Любое действие пользователя генерирует событие
- Изменения в асинхронных процессах (загрузка файла, ответ от сервера, изменения в браузере)
- События могут возникать на отдельных элементах и документе в целом

Структура события

- Тип – вид происшедшего события (mousemove, click, load)
- Цель (target) – объект, с которым ассоциировано событие
- Обработчик или прослушиватель – функция, обрабатывающая событие
- Объект события – объект, ассоциированный с событием
- Распространение событий – определение объектов, которые должны обрабатывать события

Категория событий

- Зависимые от устройства – привязаны к конкретному устройству ввода
- Независимые от устройства – не привязаны к конкретному устройству
- Пользовательского интерфейса – возникают только в графическом интерфейсе
- Изменения состояния – связанные с жизненным циклом разных объектов, с событиями за пределами браузера
- API-интерфейс – специфичные для различных API-интерфейсов элементов HTML

Регистрация обработчика

- Установка свойств
  - window.onload = ...
- Установка атрибутов HTML-элементов
  - <button onclick="...">
- Метод addEventListener

Вызов обработчика

- Автоматически после возникновения события
- Аргумент – объект Event (или производный)
  - type – тип события
  - target – объект, в котором произошло событие
  - currentTarget – объект, в котором зарегистрирован обработчик
  - timeStamp – время возникновения события
  - isTrusted – отправлено ли событие браузером
- Контекст this – объект события (кроме стрелочных функций)
- События срабатывают в порядке их регистрации

Распространение события

- Всплытие события (event bubbling) – сначала у самого глубокого в дереве элемента, затем по иерархии вверх до документа
  - Прекращение всплытия event.stopPropagation
  - Не запускать все остальные обработчики event.stopImmediatePropagation()
- Перехват событий (event cepturing) – возникновение событий в обратном порядке
  - addEventListener(..., {capture: true}) или addEventListener(..., true)
- Выполнение стандартных действий браузера в ответ на события
  - preventDefault()

Собственные события

- Разработчик может создать свой тип события
  - Конструктор Event (CustomEvent)
- Необходимо принудительно отправить событие
  - Метод dispatchEvent

Модуль 7. Введение в асинхронный JS

- Promise
- async/await
- Введение в Fetch API

Асинхронное программирование

- Запуск асинхронного кода отдельно от основного потока выполнения программы
- Функция обратного вызова – один из способов определить результат при асинхронном вызове
- Промисы – продвинутая обработка асинхронного кода

Объекты Promise

- API-интерфейсы, основанные на Promise
- Цепочки объектов Promise
- Разрешение объектов Promise
- Обработка ошибок
- Параллельное выполнение
- Последовательное выполнение

async и await

- Выражения await
- Функции async
- Ожидание множества объектов Promise

Асинхронная итерация

- Цикл for/await
- Асинхронные итераторы
- Асинхронные генераторы

Fetch API

- Строго асинхронные запросы на удаленные ресурсы
- Метод fetch отправляет запрос и возвращает Promise
- Ответ может быть в текстовом формате (JSON, XML)
- С ответом приходят заголовки и метаданные ответа (статус)