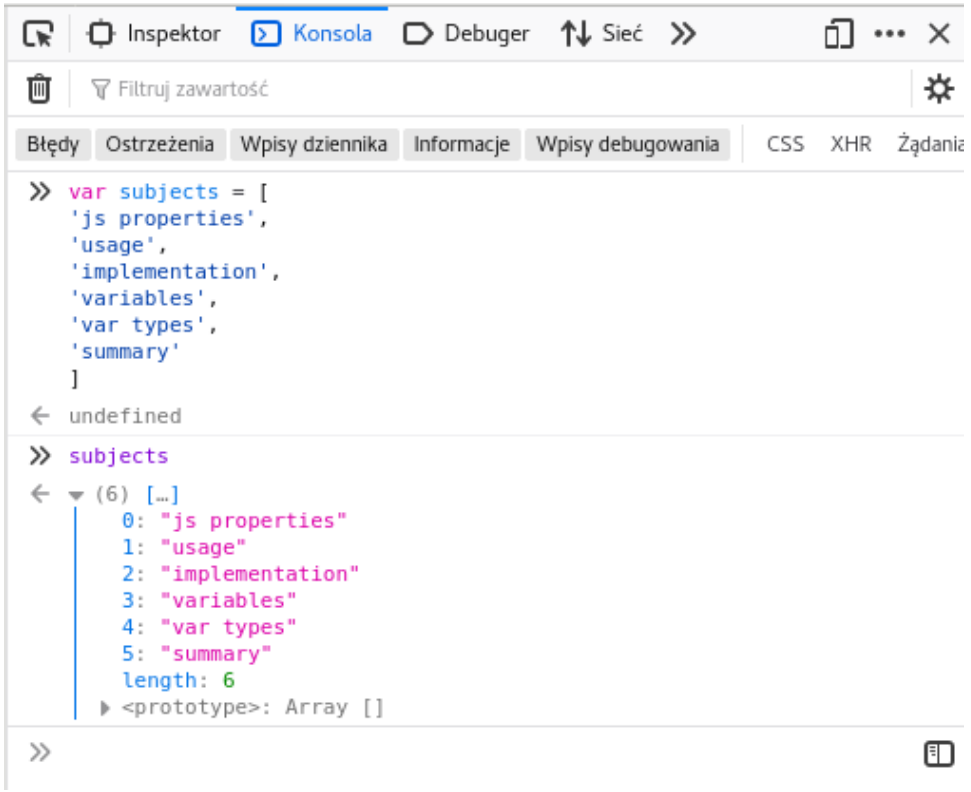
Programowanie Internetowe

- Standard ECMA Script
- Obiektowy model dokumentu
- Data, Czas, Wyrażenia regularne
- Przechowywanie danych
- Przetwarzanie asynchroniczne

Opracował: inż. Grzegorz Petri

Przegląd zagadnień

- Czym jest zdarzenie
- Podział zdarzeń
- Obsługa zdarzeń
- Bąbelkowanie / Propagacja
- Domyślna akacja oraz Przerwanie zdarzeń
- Obiekt Event (zdarzenia)



Zdarzenia DOM

Czyli co się dzieje w domu?

- Zdarzenia to rodzaj wysyłanych **powiadomień**
- Środowisko uruchomieniowe (przeglądarka) informuje kod, by umożliwić wykonanie przypisanej do zdarzenia funkcjonalności
- Każde zdarzenie jest reprezentowane przez obiekt bazujący na interfejsie Event
- Każdy obiekt zdarzenia posiada właściwości oraz funkcje używane, by jak najlepiej móc poinformować element docelowy, co i gdzie się zdarzyło
- Zdarzenia mogą reprezentować wiele kategorii działań

Podział zdarzeń

Kategorie grupujące podobne typu zdarzeń

Mysz
Mouse

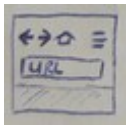
Klawisze
Keyboard

Dotyk
Touch

Przeciąganie
Drag & Drop

Skupienie uwagi
Focus

Widok
View



Historia przeglądania
SessionHistory

Drukowanie
Printing

Formularze
Form

Schowek
Clipboard

Skład tekstu
TextComposition

Sieć
Network

Gniazda
WebSocekt

Multimedia
Media

Postęp przesyłania
Progress

Zasoby
Resources

Obsługa zdarzeń

Do trzech razy sztuka

Istnieją trzy sposoby obsługi zdarzeń DOM – podobnie jak w CSS:

1) ⛔ **In-line Element** – na elemencie w dokumencie HTML (*podobnie jak w CSS*)

```
<element event="kod JavaScript"> … </element>
<b><b><s onTouchStart="alert('UcantTouchThis')">(.|.)</s></b></b>
```

2) ⚠️ **Event Handler*** – obsługa zdarzenia na elemencie (*w skrypcie JS*)

```
element.event = „kod JavaScript";
body.onload = alert(„Musisz mieć ukończone 18 lat");
```

3) ✅ **Event Listener** – nasłuchiwanie zdarzeń wg specyfikacji DOM Level 2

```
body.addEventListener(event, function, bubble);
body.addEventListener('click', bait, false);
```

**na wskazanym elemencie obsługę jednego typu zdarzenia można przypisać tylko jeden raz*

Obsługa zdarzeń

”Trójkę wybierz Panie”

Jak zwykle pojawiają się wątpliwości – jakiej metody należy użyć – więc:

1) metody **In-line Element** nie należy stosować w dzisiejszych projektach

- zasada rozdzielania warstw: wygląd-logika-dane,
- obniża to elastyczności stosowania kodu oraz utrudnia zarządzania kodem projektu,

2) metoda **Event Handler** – powinna być stasowana jedynie w ostateczności

- w przypadku konieczności zachowania wstecznej kompatybilności ze starszymi pseudo-przeglądarkami (*tj. Internet-Explorer 8*),
- ogranicza obsługę na pojedynczym elemencie do jednej instancji tego samego typu zdarzenia,

3) metoda **Event Listener** – daje najwięcej możliwości i elastyczności

- większa złożoność oraz mniejsze wsparcia dla przeglądarek wstecz (*dopiero od IE 9*),
- oprócz dodawania Słuchacza, **dla więcej niż jednego zdarzenia tego samego typu**, można dowolnie usuwać Słuchaczy ze wskazanych elementów

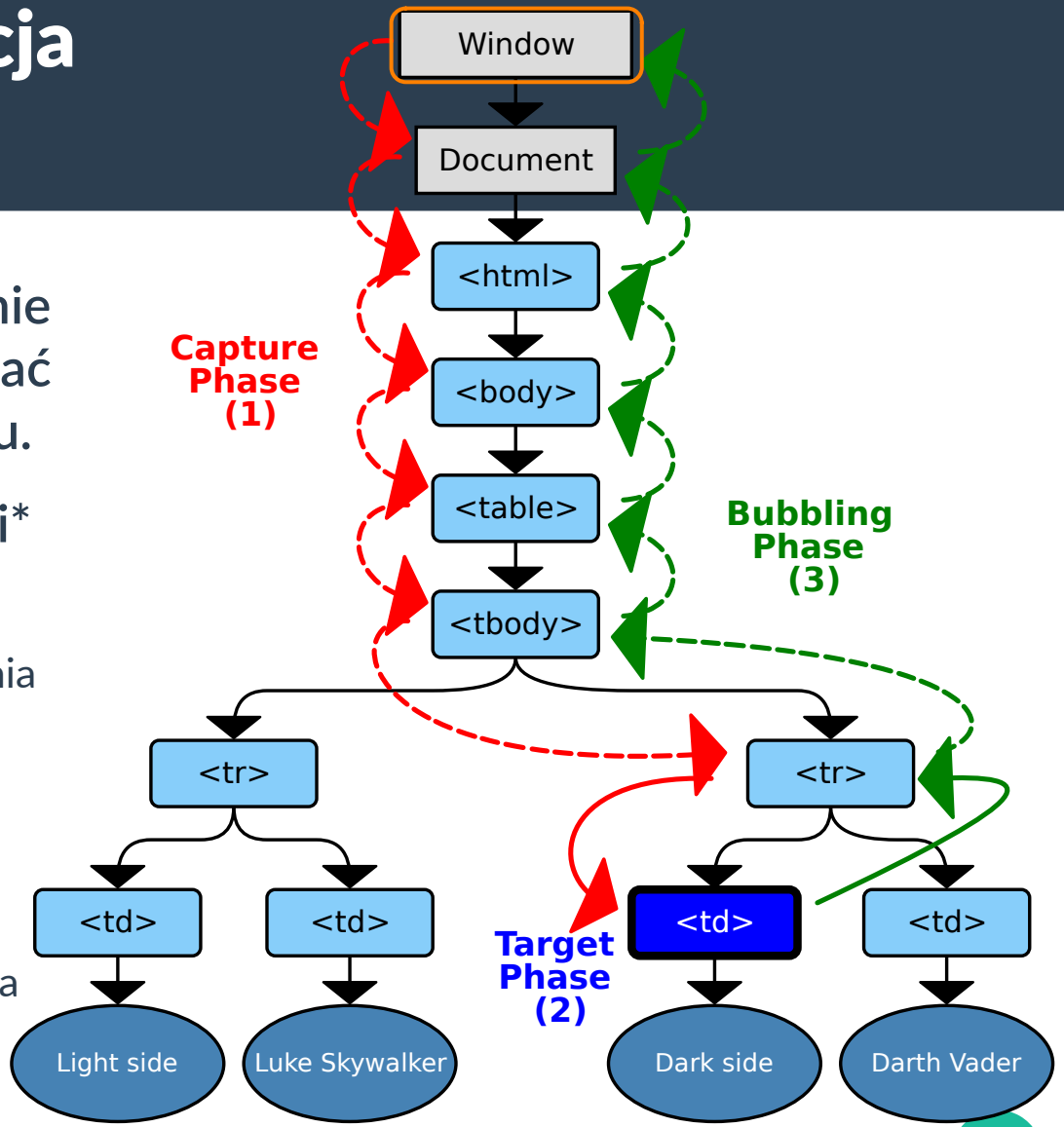
Bąbelkowanie / Propagacja

Event dispatch – wysyłanie zdarzeń

Aby Okno mogło wysłać zdarzenie do elementu docelowego musi zostać określona ścieżka dostępu do elementu.

Następnie obiekt Zdarzenia przechodzi* od fazy 1 do fazy 3 zdarzenia:

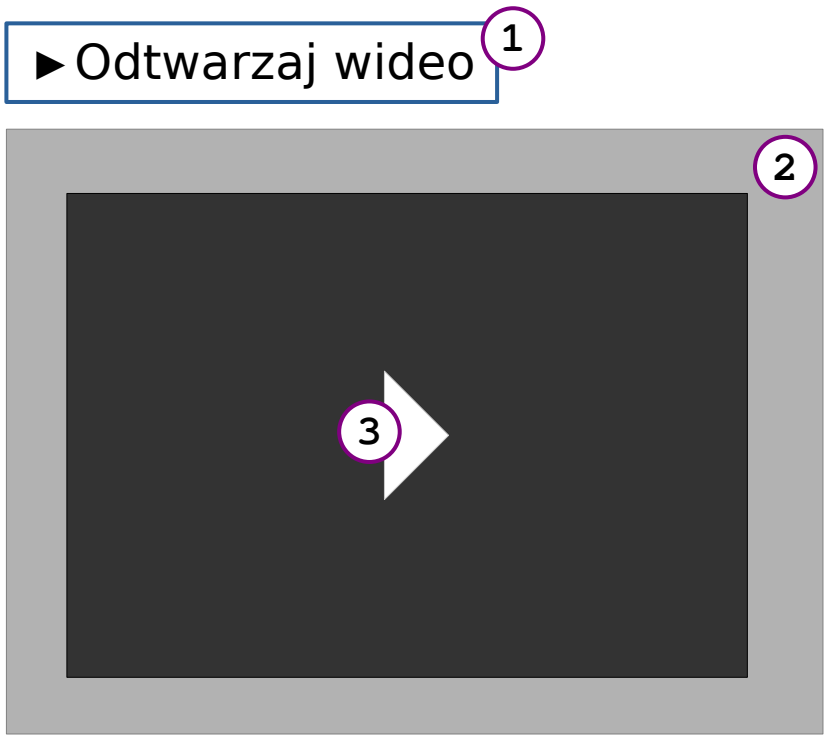
1) **przechwytywanie** (*capture*) – obiekt zdarzenia „*idzie*” od obiektu Window do rodzica celu
2) **namierzenie** (*target*) – obiekt zdarzenia dociera do elementu docelowego
3) **bąbelkowanie** (*bubbling*) – obiekt zdarzenia powraca** w odwrotnej kolejności od rodzica celu do obiektu Window



** z wyjątkiem, gdy faza jest nieobsługiwana lub **propagacja została wstrzymana*

Zatrzymanie rozprzestrzeniania się zdarzenia

Przystanek na żądanie



Przyciski wywołujące zdarzenia:

(1) Pokazanie szarego kontenera (A)
ze znacznikiem wideo (B)

(2) Rozpoczęcie odtwarzania
(zdarzenie w kontenerze (B)

(3) Ukrycie szarego kontenera (A)
(zdarzenie w kontenerze (A))

PROBLEM: zdarzenie Click (3) w (B) wywoła odtwarzanie filmu, ale przez propagację w górę wywoła też zdarzenie Click (2) w kontenerze (A), czyli ukrycie kontenera z odtwarzaczem (B)

ROZWIĄZANIE: dla elementu (B) dodać

```
event.stopPropagation();
```

Domyślne akcje oraz ich przerwanie

Lepiej zapobiegać niż wkurzać

Domyślne akcje (*default actions*) to działania wykonywane zaraz po* wysłaniu Zdarzenia (dispatch events).

PRZYKŁADY:

- Kliknięcie odnośnika **`A`**
- Zaznaczenie kontrolki **`INPUT`** checkbox lub radiobutton
- Zaznaczenie myszą tekstu
- Przeciągnięcie myszą obrazka

**z kilkoma wyjątkami, gdzie wykonywane są przed wysłaniem zdarzenia.*

Przerywane zdarzenia (*cancelable events*) to działanie, gdzie na wybranym elemencie dla wskazanego zdarzenia przerywa się domyślną akcję.

PRZYKŁAD: przycisk formularza `INPUT type="submit"` przesyła dane z formularza przeładowując dokument, w którym JavaScript przesyła dane formularza AJAX'em

ROZWIĄZANIE: wywołanie metody `event.preventDefault();`

Obiekt zdarzenia

Event objects

- Obiekt zdarzenia jest parametrem przekazywanym funkcji, która jest wywoływana przez dane zdarzenie.
- Posiada Własności oraz Metody specyficzne dla danego zdarzenia.

- Spotykane sposoby zapisu obiektu zdarzenia:
  - **`event`**
  - **`evt`**
  - **`e`**

```
document.onclick = clickbait;
function clickbait(e){
   console.log(„Pozycja X: ”+e.clientX);
   console.log(„Pozycja Y: ”+e.clientY);
}
```

Obiekt zdarzenia

Właściwości (wybrane) zdarzenia

- **`e.button`** (*zwraca przycisk myszy*)
- **`e.charCode`** (*zwraca kod Unicode klawisza znaku, który został wciśnięty podczas zdarzenia keypress*)
- **`e.keyCode`** (*zwraca kod Unicode dla klawisza nie będącego znakiem w zdarzeniu keypress lub dowolnego klawisza w każdym innym zdarzeniu związanym z klawiaturą*)
- **`e.pageX`** (*zwraca poziomą współrzędną miejsca, gdzie wystąpiło zdarzenie, względem całej strony*)
- **`e.screenX`** (*zwraca poziomą współrzędną miejsca, gdzie wystąpiło zdarzenie, względem ekranu*)
- **`e.pageY e.screenY`** (*pionową*)
- **`e.target`** (*referencja do elementu, do którego zdarzenie wysłano*)
- **`e.type`** (*nazwa zdarzenia*)
- **`e.view`** (*widok gdzie wygenerowano zdarzenie*)
- **`e.which`** (*kod Unicode klawisza, jeżeli zdarzenie pochodzi od klawiatury*)

Metody zdarzenia:

- **`e.initEvent();`**
- **`e.initKeyEvent();`**
- **`e.initMouseEvent();`**
- **`e.initUIEvent();`**
- **`e.preventDefault()`**
- **`e.stopPropagation();`**

Szczegółowa lista Właściwości Event object:
https://developer.mozilla.org/pl/docs/Web/API/Event

Podział zdarzeń

Podstawowe / popularne

Mouse

- `click`
- `contextmenu`
- `mousedown`
- `mouseenter`
- `mouseleave`
- `mousemove`
- `mouseover`
- `mouseout`
- `mouseup`
- `select`
- `wheel`

Keyboard

- `keydown`
- `keypress`
- `keyup`

Drag & Drop

- `drag`
- `dragstart`
- `dragenter`
- `dragleave`
- `dragover`
- `dragend`
- `drop`

Touch

- `touchstart`
- `touchmove`
- `touchcancel`
- `touchend`

Focus

- `focus`
- `blur`
- `focusin`
- `focusout`

Podział zdarzeń

Zaawansowane / specyficzne

View

- `fullscreenchange`
- `fullscreenerror`
- `resize`
- `scroll`

Form

- `reset`
- `submit`

Printing

- `beforeprint`
- `afterprint`

Network

- `offline`
- `online`

Progress

- `abort`
- `error`
- `load`
- `loadstart`
- `loadend`
- `progress`
- `timeout`

WebSockets

- `open`
- `message`
- `error`
- `close`

Clipboard

- *`cut`*
- *`copy`*
- *`paste`*

```
dialog(
  ‘Pytania?’
);
```