

Aplikacje webowe

DOM – elementy – zdarzenia
JavaScript

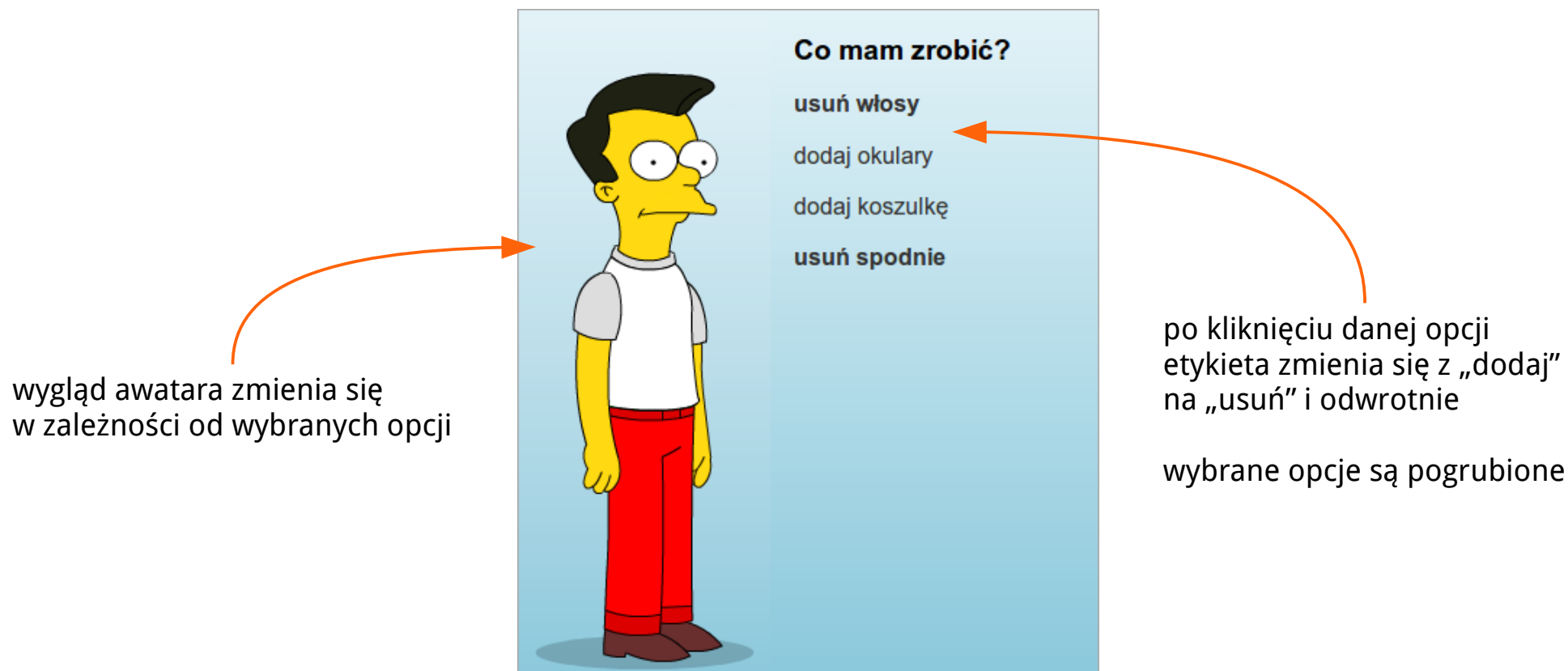
Aleksander Lamża
ZKSB · Instytut Informatyki
Uniwersytet Śląski w Katowicach

aleksander.lamza@us.edu.pl

- Obiektowy model dokumentu
- Operacje na DOM w czystym JavaScriptcie
- Obsługa podstawowych zdarzeń

Zadanie – tworzymy prosty edytor awatarów

Chcemy przygotować taką aplikację...



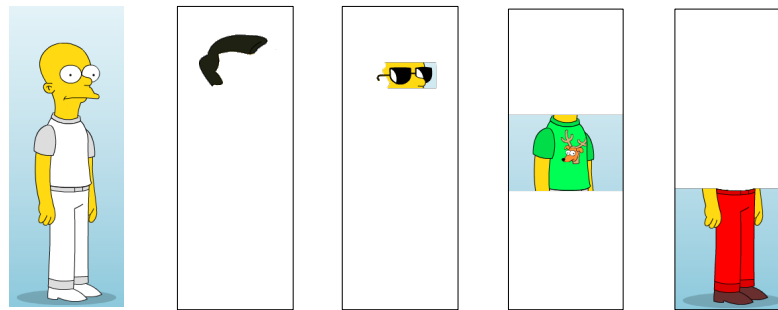
Dokument HTML

Zaczynamy od dokumentu HTML:

```
<div id="avatar">  
  <div id="avatar-base" style="background: url(images/sim.png)"></div>  
  <div id="avatar-glasses" style="background: url(images/glasses.png)"></div>  
  <div id="avatar-hair" style="background: url(images/hair.png)"></div>  
  <div id="avatar-shirt" style="background: url(images/shirt.png)"></div>  
  <div id="avatar-trousers" style="background: url(images/trousers.png)"></div>  
</div>
```



Podgląd awatara jest zbudowany z pięciu warstw. Każda z nich to div z ustawionym obrazkiem tła.



Ciąg dalszy dokumentu:

```
<div id="options">
  <h2>Co mam zrobić?</h2>
  <ul>
    <li id="hair"><a href="#"><span>dodaj</span> włosy</a></li>
    <li id="glasses"><a href="#"><span>dodaj</span> okulary</a></li>
    <li id="shirt"><a href="#"><span>dodaj</span> koszulkę</a></li>
    <li id="trousers"><a href="#"><span>dodaj</span> spodnie</a></li>
  </ul>
</div>
```

Co mam zrobić?

usuń włosy

dodaj okulary

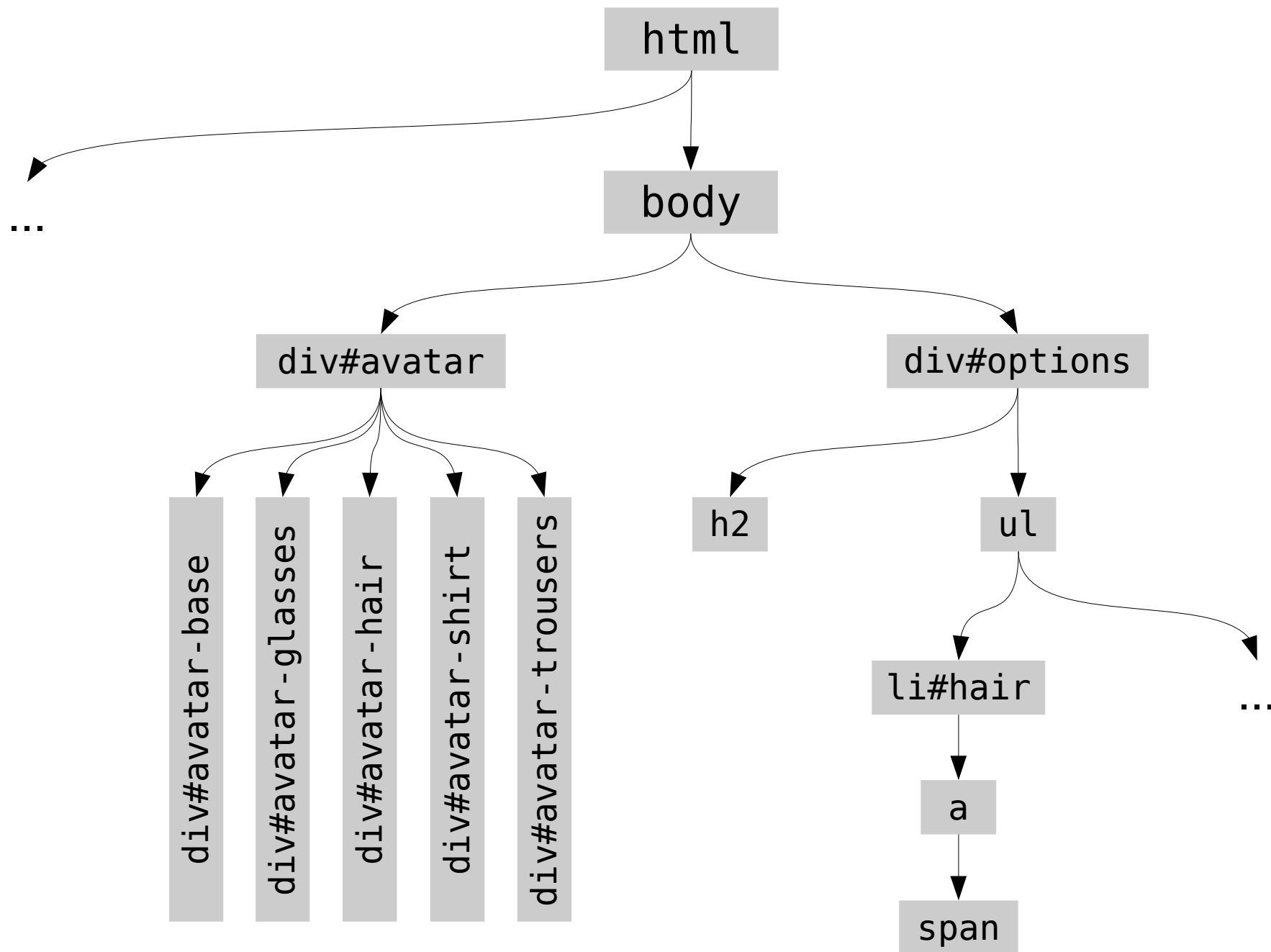
dodaj koszulkę

usuń spodnie

Opcje to linki umieszczone w liście ul.

Elementy span będą potrzebne później do zmiany „dodaj” na „usuń”.

Struktura dokumentu – DOM



Teraz trochę interakcji

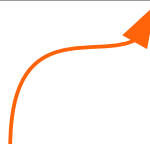
Chcemy, by po kliknięciu opcji pojawiła się odpowiednia warstwa na podglądzie awatara.

Aby to zrobić, musimy zareagować na **zdarzenie** kliknięcia na elemencie `li`.

Teraz trochę interakcji

Zaczynamy od znalezienia elementu:

```
var optHair = document.getElementById("hair");
```



Metoda `getElementById()` odszukuje w strukturze DOM element o wskazanym identyfikatorze.

Teraz możemy podpiąć do niego funkcję obsługi zdarzenia „click”:

```
optHair.onclick = function() {  
}
```



Ta funkcja zostanie wywołana po kliknięciu elementu `li` o identyfikatorze `hair`.

Teraz trochę interakcji

W funkcji obsługi zdarzenia włączamy wyświetlanie odpowiedniej warstwy:

Odszukujemy div wyświetlający włosy (avatar-hair)...

```
optHair.onclick = function() {  
    ▶ var avatarHair = document.getElementById("avatar-hair");  
    ▶ avatarHair.style.display = "block";  
}
```

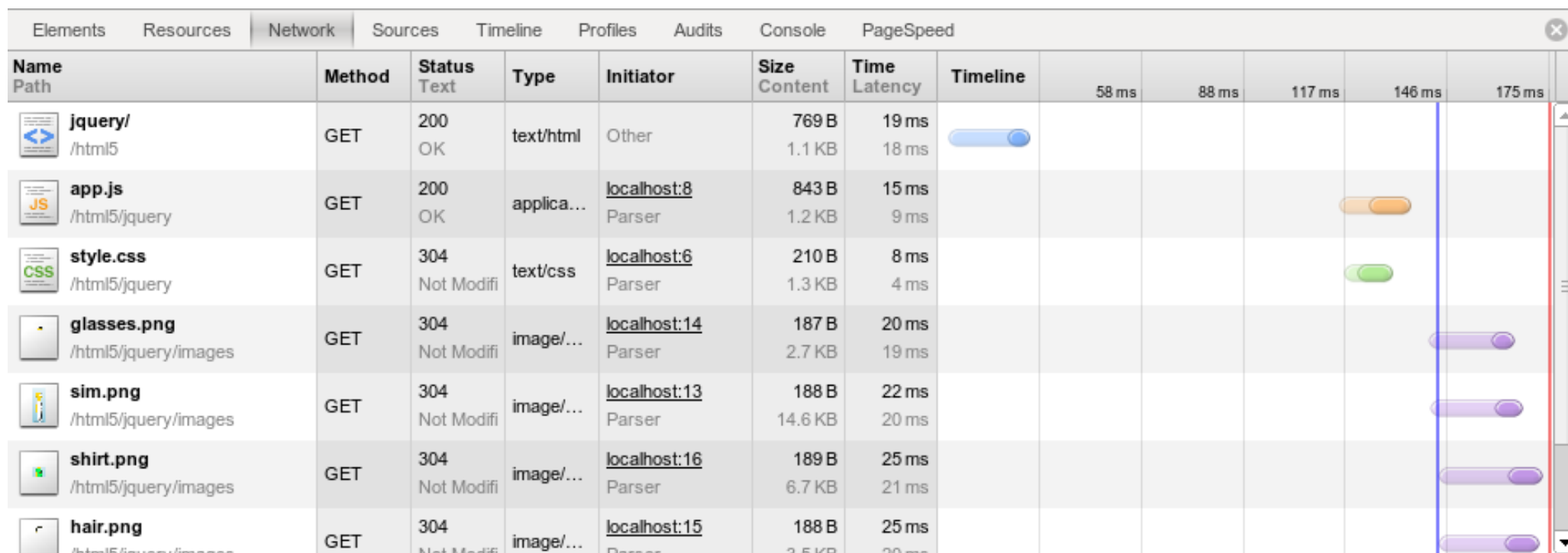
...i ustawiamy właściwość CSS display na block.

I co, działa?

Teraz trochę interakcji – problem

Nie działa! :(

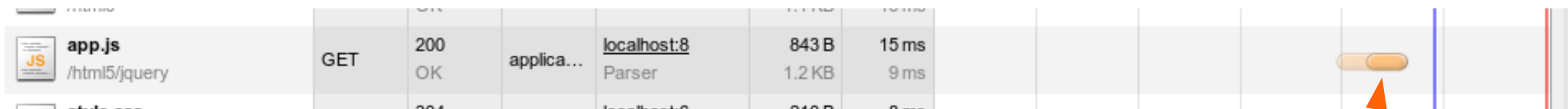
Przeanalizujmy oś czasu przedstawiającą proces ładowania strony:



Te dwie linie przedstawiają moment odpalenia zdarzeń związanych z utworzeniem struktury DOM (niebieska linia) i załadowaniem wszystkich zasobów strony (czerwona linia).

Teraz trochę interakcji – problem

No i mamy rozwiązanie problemu:



app.js	GET	200 OK	applica...	localhost:8	843 B	15 ms
style.css	GET	200 OK	style.css	localhost:8	210 B	8 ms

Nasz skrypt ładuje się i uruchamia w tym momencie...

...a struktura DOM (na której skrypt operuje) jest tworzona dopiero tutaj.

Wniosek – korzystamy z czegoś, co jeszcze nie istnieje!

Jak temu zaradzić?

Wystarczy umieścić cały nasz kod w funkcji obsługi zdarzenia load.

```
window.onload = function() {  
    ...  
}
```

Teraz trochę interakcji

Żeby dało się na zmianę włączać i wyłączać, musimy dopisać trochę kodu:

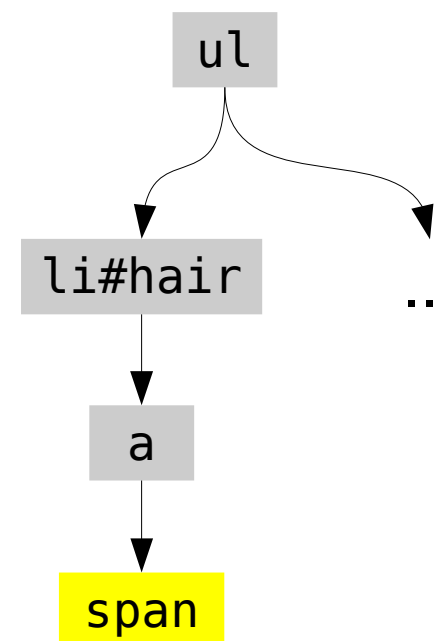
```
optHair.onclick = function() {  
    var avatarHair = document.getElementById("avatar-hair");  
  
    if (avatarHair.style.display !== "block") {  
        avatarHair.style.display = "block";  
    } else {  
        avatarHair.style.display = "none";  
    }  
}
```

A co z etykietą opcji? Po włączeniu powinna się zmienić na „usuń”...

Teraz trochę interakcji

Tu przyda nam się element span, w którym umieściliśmy tekst „dodaj”:

```
<ul>
  <li id="hair"><a href="#"><span>dodaj</span> włosy</a></li>
  ...
</ul>
```



Ponieważ span nie ma identyfikatora, znajdziemy go po nazwie:

```
var span = optHair.getElementsByTagName("span")[0];
```

optHair to element li#hair

metoda `getElementsByTagName()` zwraca tablicę elementów – wyciągamy pierwszy (i jedyny)

Teraz trochę interakcji

Teraz wystarczy uzupełnić `if`-a. Cały kod wygląda tak:

```
var optHair = document.getElementById("hair");
optHair.onclick = function() {
    var avatarHair = document.getElementById("avatar-hair");
    var span = optHair.getElementsByTagName("span")[0];

    if (avatarHair.style.display !== "block") {
        avatarHair.style.display = "block";
        span.innerHTML = "usuń";
    } else {
        avatarHair.style.display = "none";
        span.innerHTML = "dodaj";
    }
}
```

Uff... I teraz to samo trzeba będzie powtórzyć dla pozostałych opcji?

Nie – teraz zobaczmy, jak to samo (a nawet więcej) zrealizować za pomocą **jQuery**.