Aplikacje WWW

Lista 4/2021

1. (2 pkt) Umieść w folderze `img` zdjęcia o nazwach od `a0.jpg` do `a9.jpg`.

   (a) W ciele dokumentu wpisz (skopiowanie kodu, z tego dokumentu może powodować komplikacje ze znakami cudzysłowu):

   ```
   <button onclick="f(-1)"> << </button>
   <img src="img/a0.jpg">
   <button onclick="f(1)"> >> </button>
   <script>
     var i=0;
     var img=document.getElementsByTagName("img")[0];
     function f(x){
           i=i+x;
           img.src="img/a"+i+".jpg";
     }
   </script>
   ```

   Zastosuj style `img{vertical-align:middle;}` oraz `body {text-align:center}` i ustal wysokość obrazka na `400px` i wypróbuj działanie guzików `<<` oraz `>>`. W przeglądarce chrome kliknij prawym przyciskiem myszy na obrazku i wybierz opcję `zbadaj` i zaobserwuj, jak atrybut `src` zmienia się w trakcie klikania na guziki `<<` oraz `>>`.

   Wyjaśnienie: Zmienna `i` pamięta numer zdjęcia. Po kliknięciu guzika funkcja `f(x)` dodaje do niej `x` (czyli 1 lub -1) i aktualizowany jest atrybut `src` obrazka, co powoduje załadowanie odpowiedniej grafiki.

   Popraw kod funkcji `f()` tak, by nie było możliwe wyjście zmiennej `i` poza zakres 0...9. Tzn. jeśli po instrukcji `i+=x;` zmienna `i` jest za duża, to powinna być zmieniona na 0, a jeśli jest ujemna, to powinna być zmieniona na 9.

   (b) Wykonaj drugą wersję tego zadania, dodając do zmiennych skryptu tablicę z nazwami plików: `var adr=["img/a0.jpg, "img/a1.jpg", ... , "img/a9.jpg"]` i zmieniając `img.src="img/a"+i+".jpg";` na `img.src=adr[i];` Po sprawdzeniu, że program działa poprawnie, zastąp lokalne adresy w tablicy `adr` odnośnikami do grafik w internecie. Dodatkowo, zamiast liczby 10 zastosuj `adr.length` czyli faktyczną liczbę obrazków. Analogicznie 9. Zwiększ ilość obrazków i sprawdź, czy w dalszym ciągu galeria działa poprawnie (przeglądanie ma charakter cykliczny).

   (c) Inny sposób wykonania galerii jest następujący. W dokumencie z punktu (a) zamiast jednego umieść 10 obrazków, ale 9 z nich ukryj stosując styl `display:none`. Zmodyfikuj funkcję `f()` dodając przed modyfikacją zmiennej `i` ukrycie `i`-tego obrazka: `document.images[i].style.display='none'` a po zmianie `i` pokazanie `i`-tego obrazka: `document.images[i].style.display='inline'`.

   (d) Zmodyfikuj zadanie (c) tak, aby zmiana zdjęcia na kolejne następowała samoczynnie co sekundę. Wskazówka: można napisać funkcję `function next(){f(1)}` i wywoływać ją co sekundę za pomocą `setInterval()`. Pomocne może być zrozumienie, jak działa przykład `swiatla.html` z wykładu 3.

2. Zmień zadanie 1.(d) tak, by zamiast obrazków, były pokazywane po kolei początkowo ukryte paragrafy tekstu. Ukrycie wykonaj w globalnym znaczniku `<style>`, a nie przez atrybut `style` w każdym z paragrafów. Tekst powinien być sensowny, a poszczególne paragrafy mogą zawierać zdjęcia ilustrujące ze stylem np `float:left`. Zachowaj guziki `(<<) (>>)` do ręcznej zmiany paragrafu, ale dodaj guziki 'start', 'stop' które

pozwolą na uruchamianie i zatrzymywanie pokazu slajdów funkcjami `setInterval()` i `clearInterval()`. Użyj funkcji `document.getElementsByTagName('p')[i]` zamiast `document.images[i]`, a zamiast liczby 10 używaj faktycznej ilość paragrafów w dokumencie otrzymanej z własności `length`. Czas wyświetlania paragrafów dostosuj do ich przeciętnej długości.

3. Zegar.

   (a) Cyfrowy. W dokumencie jest tylko element `<H1>` wyśrodowany za pomocą stylu. Co sekundę w tym elemencie zmienia się `innerHTML` tak, by zawsze pokazywać aktualną godzinę w formacie gg:mm:ss czyli np. `12:34:45`. Wskazówka: `setInterval()`, `https://www.w3schools.com/js/tryit.asp?filename=tryjs_date_gethours`, `https://www.w3schools.com/js/tryit.asp?filename=tryjs_date_getminutes` itd..

   (b) Analogowy. W dokumencie są trzy ramki (wskazówki) o szerokości 10px i wysokościach 200px, 300px, 310px (dostosuj wymiary i kolory wg własnego uznania). Ich kąty odchylenia w prawo od pionu zmieniają się co sekundę i wynoszą odpowiednio (gg+mm/60)/12 pełnego obrotu, (mm+ss/60)/60 pełnego obrotu i ss/60 pełnego obrotu (obrót to `360deg` lub `1turn`). Wskazówka: Użyj na przykład `position:absolute;bottom:50%;left:50%` oraz `transform-origin: 50% 100%;` w CSS oraz `.style.transfom="rotate("+kąt+"deg)"` w skrypcie.

4.
```
<table><tr><td rowspan=2 style='font-size:3em'>{</td>
<td><input>x+<input>y=<input></td></tr>
<tr><td><input>x+<input>y=<input></td></tr>
</table><button onclick="solve()">Rozwiąż</button>
<div id='wynik'></div>
```

   Wpisz powyższy kod do ciała dokumentu `.html`, ustaw w nagłówku szerokość elementów `input` na `30px`, a w skrypcie poniżej tabeli napisz funkcję `solve()`, która metodą wyznacznikową, rozwiąże powyższy układ równań a wynik umieści w ramce o `id=wynik`. Możesz wzorować się na rozwiązaniu zadania 10 z listy 2. Twoja funkcja powinna prawidłowo zachowywać się dla każdego z 3 przypadków: (a) $W \neq 0$ - jedno rozwiązanie (b) $W = 0$ a $W_x \neq 0$ lub $W_y \neq 0$ - brak rozwiązań (c) $W = W_x = W_y = 0$ - nieskończenie wiele rozwiązań.

5. Zmodyfikuj przykładowy plik `6b.piłeczki.html` z wykładu 3 umieszczając

   wewnątrz elementów `<b>` elementy `<img>` tak, by zamiast strzałek po ekranie poruszały się samoloty, ptaki, owady lub inne obiekty wg Twojego uznania. Zmodyfikuj skrypt i style tak, aby obiekty nie wychodziły poza obramowanie. Zmień również rozmiar sceny oraz liczbę obiektów.

   Przeczytaj na temat `css {transform:rotate(30deg)}`. Zauważ polecenie `b[i].style.transform="rotate("+Math.atan2(b[i].vy,b[i].vx)+"rad)"`, które sprawiało, że strzałki były zawsze ustawione w kierunku ruchu. Zaktualizuj je (dodając odpowiedni kąt), by przez ciebie dodane obiekty (samoloty/ ptaki itp) zawsze były skierowane w kierunku przemieszczania się.

6. (3pkt) Dodaj do poprzedniego zadania paragraf, w którym będzie widoczna aktualna liczba obiektów poruszających się w ramce, oraz obsługę zdarzenia `onmousedown`, która spowoduje zatrzymanie (albo zniknięcie) kliknięty obiektu, oraz zmniejszenie liczby widocznej w paragrafie. Po zatrzymaniu (zniknięciu) ostatniego obiektu w paragrafie powinien pokazać sie napis „Mission completed in " oraz czas jaki upłynął od kliknięcia guzika „start". Dodaj guzik „Nowa gra", który spowoduje ponowne pokazanie wszystkich obiektów, oraz nadanie im nowych położeń i prędkości.

Dodaj przyciski „Poziom 0" „Poziom 1", itd. które będą działały tak jak „Nowa gra" ale dodatkowo będą zmieniać poziom trudności gry, czyli liczbę i prędkości poruszających się obiektów. Wskazówka: prędkość zależy do drugiego argumentu funkcji `setTimeout`.

7. Napisz grę "Dark room". Na planszy 10 × 10 (`<table>`)w losowych polach pojawiają się dwa kółka: czerwone i zielone, które na przemian (np. co pół sekundy) wykonują przypadkowe ruchy. Ruch może być w tylko o jedno pole w poziomie w pionie lub na ukos (jak król w szachach). Ruch nie może spowodować wyjścia poza planszę. Zwycięża to kółko, które wejdzie na poje zajmowane przez przeciwnika. Wtedy w ramce `H1` pojawia się rezultat np "1:0" i ruchy kółek się zatrzymują. Przycisk `(nowa gra)` powoduje zaczęcie wznowienie rozgrywki od nowych losowych pozycji. Ostatecznie wygrywa ten, który zdobędzie jako pierwszy 5 punktów. Wskazówka: jak wykonywać losowe ruchy sprawdź w pliku `5.15.html` w funkcji `mieszaj()`.

8. Dana jest tablica obiektów:

```
var student=[ {imie:"Ola",nazwisko:"Lis",kierunek:"fizyka",wzrost:170},
              {imie:"Ala",nazwisko:"Góral",kierunek:"chemia",wzrost:165},
              {imie:"Jan",nazwisko:"Nowak",kierunek:"fizyka",wzrost:180}];
```

   Dopisz do niej kilkunastu studentów z różnych kierunków. Napisz funkcję:

   (a) `function wszyscy()`, która w elemencie `<div>` o `id="lista"` umieści nagłówek `<h2>Studenci</h2>` oraz listę `<ol><li>` wszystkich studentów w formacie "imię nazwisko (kierunek)".

   (b) `function pokaz(kierunek)`, która w elemencie `<div>` o `id="lista"` umieści element `<h2>` z nazwą kierunku oraz wyliczenie `<ol>` z imionami i nazwiskami studentów podanego kierunku.

   (c) Dodaj do elementu `<li>` zawierającego dane studenta atrybut `style` tak, aby kolor tła elementu zależał od kierunku (np. fizyka – niebieski, chemia – czerwony, itp.) a szerokość (`width`) była proporcjonalna od wzrostu studenta.

   Wypróbuj działanie funkcji za pomocą kilku przycisków podobnych do:

```
<button onclick="wszyscy()">Wszyscy</button>
<button onclick="pokaz('fizyka')">Fizyka</button>
<button onclick="pokaz('chemia')">Chemia</button>
```

9. Naucz się stosować w dokumencie google fonts (`https://fonts.google.com/`). Wykonaj prosty dokument, w którym zastosujesz ciekawą, nietypową czcionkę do nagłówków: `H1, H2, H3`, oraz inną czcionkę do `body` oraz paragrafów `p`. Styl czcionki powinien, w miarę możliwości, pasować do treści dokumentu.

10. Wykonaj logo fikcyjnej firmy rozmieszczając poszczególne litery zamknięte (każda osobno) w znacznikach `span` wewnątrz ramki `div`. Zastosuj styl
    `div {positon:relative}`
    `div>span {position:absolute}`
    a w selektorach `div>span:nth-child(1)` itd.. ustaw różne wartości altrybutów `left, top, font-size, color, background, transform, text-shadow, border-radius`, itd.. tak, aby poszczególne litery różniły się wielkością, kolorem i krojem położeniem w ramce a nawet nachyleniem (`transform` pozwala obracać litery). Ważne jest niekonwencjonalne pozycjonowanie lub przeplatanie się liter. Wskazanie jest zastosowanie google fonts.