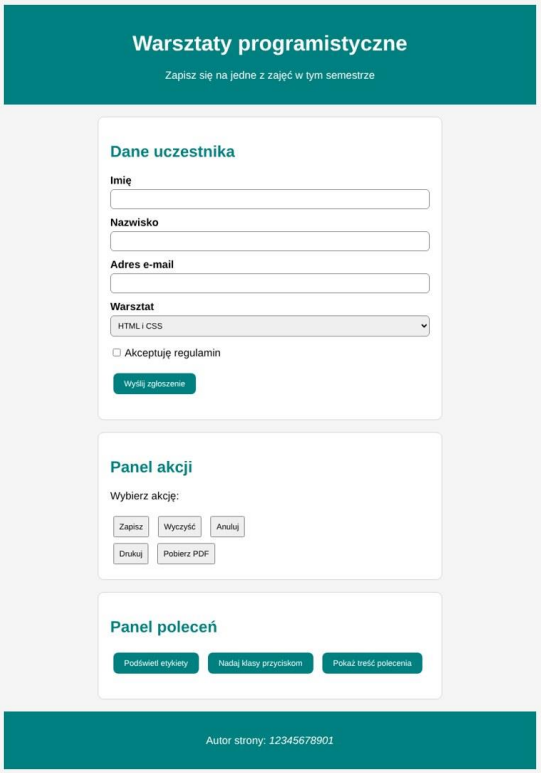
Zadanie 14: DOM – style i klasy elementów

Treść zadania

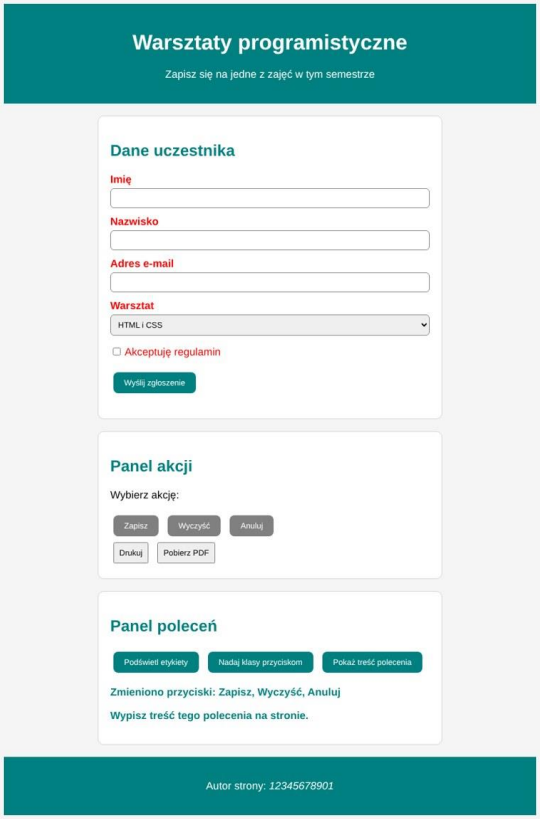
Wykonaj stronę internetową zapisów na warsztaty programistyczne, wykorzystując edytor zaznaczający składnię oraz przeglądarkę internetową. Utwórz folder o nazwie zadanie14. Po skończonej pracy wszystkie pliki zapisz w tym folderze.

Witryna internetowa

Wygląd strony przed kliknięciem jakiegokolwiek przycisku przedstawia ilustracja 1, a po kliknięciu kolejno wszystkich trzech przycisków z panelu poleceń – ilustracja 2.



Ilustracja 1. Strona warsztaty.html po otwarciu w przeglądarce



Ilustracja 2. Strona po kliknięciu przycisków „Podświetl etykiety”, „Nadaj klasy przyciskom” i „Pokaż treść polecenia”

Cechy strony warsztaty.html:

- Zapisana w języku HTML5
- Zadeklarowany polski język zawartości witryny
- Jawnie zastosowany standard kodowania polskich znaków UTF-8
- Tytuł strony „Zapisy na warsztaty”
- Arkusz stylów w pliku o nazwie styl.css prawidłowo połączony z kodem strony
- Skrypt w pliku o nazwie skrypt.js, połączony ze stroną znacznikiem `<script>` umieszczonym na końcu bloku `<body>`
- Podział strony na bloki: nagłówkowy, główny i stopkę, za pomocą znaczników `<header>`, `<main>` i `<footer>`. Blok główny zawiera trzy bloki `<section>` ułożone jeden pod drugim, zgodnie z ilustracją 1
- **Zawartość bloku nagłówkowego:** nagłówek pierwszego stopnia o treści „Warsztaty programistyczne” oraz paragraf o treści „Zapisz się na jedne z zajęć w tym semestrze”
- **Zawartość pierwszej sekcji** (`id="formularz"`):
- nagłówek drugiego stopnia o treści „Dane uczestnika”
- formularz (`<form>`) zawierający cztery pola, każde z podpisem w znaczniku `<label>` powiązanym z polem atrybutem `for` i formatowanym klasą o nazwie `etykieta`:
- „Imię” – pole tekstowe (`type="text"`, `id="imie"`)
- „Nazwisko” – pole tekstowe (`id="nazwisko"`)
- „Adres e-mail” – pole typu `email` (`id="email"`)

- „Warsztat” – lista rozwijana (`<select>`, `id="warsztat"`) z elementami „HTML i CSS”, „JavaScript”, „Bazy danych”
- Pola tekstowe, pole e-mail i listę rozwijaną formatuje klasa o nazwie `pole`
- paragraf zawierający pole wyboru (`type="checkbox"`, `id="regulamin"`) oraz podpis `<label>` „Akceptuję regulamin” powiązany z tym polem. Ten `<label>` nie ma klasy
- paragraf z przyciskiem „Wyślij zgłoszenie” (`<button>`) formatowanym klasą o nazwie `glowny`. Przycisk nie ma funkcjonalności
- **Zawartość drugiej sekcji** (`id="zadanie2"`):
- nagłówek drugiego stopnia o treści „Panel akcji” oraz paragraf o treści „Wybierz akcję:”
- **bezpośrednio w sekcji** trzy przyciski `<button>` o treści „Zapisz”, „Wyczyść”, „Anuluj”
- blok `<div>` zawierający dwa przyciski o treści „Drukuj” i „Pobierz PDF”
- przyciski w tej sekcji nie mają klas ani funkcjonalności
- **Zawartość trzeciej sekcji** (`id="polecenia"`):
- nagłówek drugiego stopnia o treści „Panel poleceń”
- trzy przyciski formatowane klasą `glowny`: „Podświetl etykiety” (`id="przyciskEtykiety"`), „Nadaj klasy przyciskom” (`id="przyciskKlasy"`), „Pokaż treść polecenia” (`id="przyciskTresc"`)
- dwa puste paragrafy: pierwszy z `id="zmienione"`, drugi z `id="wynik"`
- **Zawartość bloku stopki:** paragraf o treści „Autor strony: ”, dalej wstawiony numer zdającego, zapisany za pomocą znacznika semantycznego oznaczającego tekst uwypuklony, formatowany domyślnie jako pochylony

Styl CSS witryny internetowej

Styl CSS zdefiniowany jest w całości w zewnętrznym pliku o nazwie styl.css. Cechy formatowania CSS działające na stronie:

- Domyślnie, dla wszystkich selektorów: krój czcionki „Verdana”, w przypadku braku sans-serif
- Dla ciała strony: kolor tła WhiteSmoke, rozmiar czcionki 110%
- Dla bloku nagłówkowego i stopki: kolor tła Teal, kolor czcionki White, wyrównanie tekstu do środka, marginesy wewnętrzne 20 px
- Dla bloków sekcji: kolor tła White, szerokość 60%, marginesy zewnętrzne górny i dolny 20 px, lewy i prawy wyliczane automatycznie przez przeglądarkę, marginesy wewnętrzne 20 px, obramowanie linią ciągłą o grubości 1 px i kolorze LightGray, zaokrąglenie rogów 10 px
- Dla nagłówków drugiego stopnia: kolor czcionki Teal
- Dla klasy etykieta: wyświetlanie blokowe, marginesy zewnętrzne górny 12 px, prawy 0, dolny 4 px, lewy 0, czcionka pogrubiona
- Dla klasy pole: model pudełkowy border-box, szerokość 100%, marginesy wewnętrzne 8 px, obramowanie linią ciągłą o grubości 1 px i kolorze Gray, zaokrąglenie rogów 6 px
- Gdy element klasy pole jest aktywny (pseudoklasa `:focus`): kolor obramowania Teal
- Dla selektora przycisku: marginesy zewnętrzne 5 px, marginesy wewnętrzne 8 px
- Dla klasy glowny: kolor tła Teal, kolor czcionki White, marginesy wewnętrzne 10 px (góra i dół) i 18 px (lewy i prawy), brak obramowania, zaokrąglenie rogów 8 px, kursor pointer
- Dla klasy btn: marginesy wewnętrzne 10 px (góra i dół) i 18 px (lewy i prawy), brak obramowania, zaokrąglenie rogów 8 px, kursor pointer
- Dla klasy btn-secondary: kolor tła Gray, kolor czcionki White

- Gdy kursor znajdzie się na elemencie klasy btn-secondary, kolor tła zmienia się na DimGray
- Dla elementów o identyfikatorach zmienione i wynik (selektor identyfikatora): kolor czcionki Teal, czcionka pogrubiona

Uwaga: nie należy zmieniać nazw klas ani identyfikatorów, gdyż jest to uwarunkowane projektem całej witryny. Klasy btn i btn-secondary nie są przypisane w kodzie HTML. Nadaje je skrypt.

Skrypt działający po stronie klienta

Wymagania dotyczące skryptu:

- Zapisany w języku JavaScript w pliku skrypt.js
- Należy stosować znaczące nazewnictwo zmiennych w języku polskim lub angielskim
- Każde z trzech poleceń uruchamia się po kliknięciu odpowiedniego przycisku z trzeciej sekcji. Po otwarciu strony żaden z elementów nie jest zmieniony

Polecenie 1. Po kliknięciu przycisku „Podświetl etykiety" skrypt dodaje do wszystkich elementów o znaczniku `<label>` styl „color" o wartości „red".

Polecenie 2. Po kliknięciu przycisku „Nadaj klasy przyciskom" skrypt dodaje klasy „btn" i „btn-secondary" do wszystkich dzieci elementu o identyfikatorze „zadanie2", które mają znacznik `<button>`. Następnie wypisuje w akapicie o identyfikatorze „zmienione" tekst według wzoru: „Zmieniono przyciski: <napisy>", gdzie <napisy> oznacza napisy ze zmienionych przycisków, oddzielone przecinkiem i spacją.

Polecenie 3. Po kliknięciu przycisku „Pokaż treść polecenia" skrypt wypisuje w akapicie o identyfikatorze „wynik" tekst „Wypisz treść tego polecenia na stronie.". Kolejne kliknięcie zastępuje wpisany tekst, a nie dopisuje go.

Pytanie do polecenia 2. Które elementy zostały zmienione i dlaczego? Odpowiedź zapisz w pliku tekstowym odpowiedz.txt.

Tabela 1. Wybrane pola i metody modelu DOM języka JavaScript

Wyszukiwanie i zmiana elementów	Dodatkowo dopuszczalne
`document.getElementById(id)`	`element.children`
`document.getElementsByTagName(TagName)`	`element.tagName`
`document.querySelectorAll(CSSselector)`	`element.classList.add(klasa)`
`element.innerHTML = "wartość"`	`element.textContent = "wartość"`
`element.style.property_name = "wartość"`	`element.addEventListener(zdarzenie, funkcja)`
`element.setAttribute(atrybut, wartosc)`	`element.className`

UWAGA: po zakończeniu pracy sprawdź działanie strony w przeglądarce. W folderze zadanie14 powinny znajdować się pliki: warsztaty.html, styl.css, skrypt.js, odpowiedz.txt.
Ocenie podlegać będą 3 rezultaty: zawartość witryny internetowej, styl CSS witryny internetowej, skrypt.

Rozwiązanie

Pytania teoretyczne

Pytanie 1. Czym różni się kolekcja zwracana przez `getElementsByTagName()` od listy z `querySelectorAll()`? Co oznacza, że kolekcja jest „żywa"?

Pytanie 2. Kiedy lepiej ustawić wygląd elementu przez `style`, a kiedy przez `classList`?

Pytanie 3. Który selektor wybierze tylko przyciski będące bezpośrednimi dziećmi elementu `#zadanie2`?

- **A.** `#zadanie2 button`
- **B.** `#zadanie2 > button`
- **C.** `#zadanie2 + button`
- **D.** `#zadanie2 ~ button`

Pytanie 4. Dlaczego zapis `document.getElementById("zadanie2").children.forEach(...)` zgłosi błąd? Jak go obejść?

Pytanie 5. Dlaczego porównanie `dziecko.tagName === "button"` nie znajdzie żadnego przycisku?

Pytanie 6. Dlaczego znacznik `<script>` ze skryptem umieszczamy na końcu `<body>`? Co się stanie, jeśli umieścisz go w `<head>` bez dodatkowych atrybutów? Jak temu zaradzić?

Pytanie 7. Czym różnią się `textContent` i `innerHTML`? Którego użyjesz do wstawienia tekstu wpisanego przez użytkownika?

Odpowiedzi do pytań teoretycznych