

NSWI142 Webové aplikace – Zkouškový test

Jméno: Standa Pilný

11.2.2015

U každé otázky může být více správných odpovědí, ale nemusí být správná žádná. Zaškrtněte pouze odpovědi, které jsou zcela pravdivé. Z čistě formálního hlediska hodnotíme správnost odpovědí vzhledem k ECMAScriptu verze 5, PHP 5.3 (a vyšší) a rozhraní, která jsou implementována v aktuálních verzích prohlížečů Firefox a Chrome.

Pokud není uvedeno jinak, každá otázka je hodnocena jako celá dobře, nebo celá špatně. Z testu je možné získat až 100 bodů a počet získaných bodů určuje výslednou známku takto:

- 91 bodů a více: výborně,
- 90 – 76 bodů: velmi dobře,
- 75 – 60 bodů: dobře,
- 59 bodů a méně: neprospěl(a).

Pokud jste vypracovali a úspěšně odevzdali bonusový domácí úkol, bude vám přičteno 15 bodů k výsledku testu.

1. Uvažujte následující příklad HTML5 kódu a rozhodněte, která z následujících možností je správně. [3 body]

```
<article><h1>Heading A</h1>
  <section><h1>Heading B</h1>
    <section><h1>Heading C</h1></section>
    <section><h1>Heading D</h1></section>
  </section>
</article>
```

- ☐ Heading A, B, C a D jsou nadpisy první úrovně, neboť je použit element `<h1>`.
- ☒ Pouze "Heading A" je nadpis první úrovně. Heading B je nadpis druhé úrovně, C a D jsou nadpisy třetí úrovně. Úroveň totiž určuje počet zanořených elementů `<section>`.
- ☐ Prohlížeč zobrazující HTML stránku nahlásí chybu, neboť HTML dokument není validní.

2. Uvažme následující HTML formulář. Formulář vhodně doplňte/upravte tak, aby byl obsah vyplněný uživatelem při kliknutí na tlačítko "Odeslat" odeslán skriptu na adrese `http://example.cz/zpracuj.php` a aby byly vyplněné hodnoty předány přímo v URL.

[4 body]

```
<form method="get" action="http://example.cz/zpracuj.php">
  Jméno: <input type="text" name="firstName" /><br/>
  Příjmení: <input type="text" name="surname" /><br/>
  <input type="submit" value="Odeslat" />
</form>
```

3. Nakreslete, jak bude následující HTML kód zobrazen v prohlížeči.

[5 bodů]

```
<table>
  <tr>
    <td colspan="2">Adult</td><td>2</td>
  </tr>
  <tr><td>A1</td><td>34</td>
    <td rowspan="2">2</td></tr>
  <tr><td>A2</td><td>32</td></tr>
  <tr>
    <td colspan="2">Child</td><td>3</td>
  </tr>
  <tr><td>C1</td><td>4</td><td>1</td></tr>
  <tr><td>C2</td><td>8</td>
    <td rowspan="2">2</td></tr>
  <tr><td>C3</td><td>12</td></tr>
</table>
```

Adult		2
A1	34	2
A2	32	
Child		3
C1	4	1
C2	8	2
C3	12	

4. K čemu slouží hodnota "hidden" atributu "type" HTML elementu <input>?

[4 body]

- ☐ Taková hodnota neexistuje a nelze ji tedy použít.
- ☐ Definuje vstupní pole formuláře, které není pro uživatele viditelné. Místo něj ukáže obvykle prohlížeč malou ikonku, na kterou uživatel může kliknout a tím dojde ke zobrazení celého pole.
- ☒ Definuje vstupní pole formuláře, které není pro uživatele viditelné a uživatel ani nemůže přímo měnit jeho hodnotu. Ta je přímo zakódována v HTML a může být měněna JavaScriptem na straně klienta.

5. Vyberte všechny fragmenty kódu, které **nesplňují** pravidla jazyka HTML.

[4 body]

- ☐ <table id=1><tr><td>Martin</td></tr><tr><td>Novák</td></tr></table>
- ☒ <table id="1"><tr><td>Martin</tr></td><tr><td>Novák</tr></td></table>
- ☐ <table id="1"><tr><td>Martin<tr><td>Novák</table>

6. Uvažme následující fragment HTML kódu. Hledáme co nejkratší CSS selektor, který zacílí všechna telefonní čísla uvedená uvnitř buňky tabulky.

[4 body]

```
<ul>
  <li class="phone">777000000</li>
</ul>
<table>
  <tr><td>Telefon:</td><td><span class="phone">7771111111</span>
    (alternativa: <i class="phone">777222222</i></td></tr>
  <tr><td>Email:</td><td class="email">css@guru.cz</td></tr>
</table>
```

Selektor: td .phone

7. Uvažme CSS selektor ".phone". Pro tento selektor platí:

[3 body]

- ☒ ".phone" je ekvivalentní s "[class~='phone']".
- ☐ ".phone" je ekvivalentní s "[class=phone]".
- ☐ ".phone" nelze přímo opsat pomocí jiného CSS selektoru.

8. Která tvrzení platí pro následující CSS selector?

[4 body]

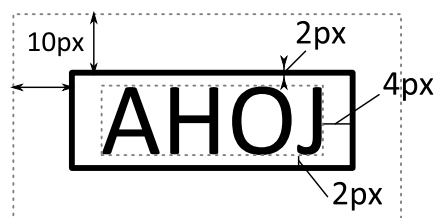
```
tr:not(:nth-child(-n+1)):not(:nth-last-child(-n+1))
```

- ☐ Selektor zacílí všechny liché řádky každé tabulky.
- ☐ Selektor zacílí první a poslední řádek každé tabulky.
- ☒ Selektor zacílí všechny řádky každé tabulky kromě prvního a posledního.
- ☐ Selektor nezacílí nic, protože pravidlo obsahuje spor.

9. Uvažme následující CSS pravidlo aplikované na element <div>AHOJ</div>. Zakreslete na papír, jak bude tento element vizualizován prohlížečem. Pokuste se zachytit všechny určené okraje a naznačte jejich velikosti v pixelech.

[4 body]

```
div {
  margin: 10px;
  padding: 2px 4px 2px 4px;
  border: 2px solid black;
}
```



10. Napište CSS pravidlo, které posune nápis AHOJ v následujícím fragmentu HTML kódu o 5px nahoru vůči jeho normální pozici. [5 bodů]

```
<div>Toto je pozdrav: <span class="petNahoru">AHOJ</span></div>
```

CSS (včetně selektoru): .petNahoru { position: relative; top: -5px; }

11. Pro proměnné v ECMAScriptu platí: [2 body/odpověď]

- ☐ Mají pevně definovaný datový typ, který je daný typem hodnoty, která je do nich poprvé přiřazena.
- ☒ Deklarují se klíčovým slovem `var`.
- ☒ Nevyhrazují konkrétní místo v paměti, ale pouze vytváří pojmenovanou vazbu k existující hodnotě, která vznikla jako literál nebo výsledek výrazu.

12. Uvažme následující fragment ECMAScriptu. Po vykonání tohoto skriptu bude platit: [5 bodů]

```
var foo = 1;
function barA() { foo = 2; }
function barB() { var foo = 3; }
barA();
barB();
```

- ☐ Existuje právě jedna proměnná `foo`, která má globální scope a hodnotu 3.
- ☒ Existuje jedna globální proměnná `foo` s hodnotou 2 a jedna lokální proměnná `foo` skrytá v uzávěru (closure) volání funkce `barB()`, takže k ní není možné nijak přistoupit.
- ☐ Existuje jedna globální proměnná `foo` s hodnotou 1 a jedna lokální proměnná `foo` skrytá v uzávěru (closure) volání funkce `barB()`, takže k ní není možné nijak přistoupit.
- ☐ Existuje jedna globální proměnná `foo` s hodnotou 2 a jedna lokální proměnná `foo` asociovaná s funkcí `barB()` (což je speciální typ objektu), ke které je možné přistupovat přes identifikátor funkce `barB` (t.j. jakožto k položce objektu `barB`).
- ☐ Globální proměnná `foo` při volání funkce `barA()` zanikne, protože je nahrazena lokální proměnou `foo`, přičemž lokální `foo` je odstraněna, protože neexistují reference dovnitř uzávěru (closure) funkce `barA()`.

13. Uvažme následující fragment HTML kódu. Při kliknutí myši kamkoli do oblasti ohraničené elementem `<div2>`, ale mimo element `<div3>`, zobrazí prohlížeč uživateli následující zprávy (zaškrtněte všechny možnosti, které jsou přípustné). [4 body]

```
<div id="div1">
  <div id="div2">
    <div id="div3">
      ...
    </div></div></div>
<script type="text/javascript">
  document.getElementById("div1").onclick = function() {
    window.alert("div1 clicked");
  };
  document.getElementById("div2").onclick = function() {
    window.alert("div2 clicked");
  };
  document.getElementById("div3").onclick = function() {
    window.alert("div3 clicked");
  };
</script>
```

- ☐ "div2 clicked"
- ☐ "div1 clicked", "div2 clicked"
- ☐ "div1 clicked", "div2 clicked", "div3 clicked"
- ☐ "div1 clicked", "div2 clicked", "div3 not clicked"
- ☒ "div2 clicked", "div1 clicked"
- ☐ "div3 clicked", "div2 clicked", "div1 clicked"
- ☐ "div1 clicked", "div2 clicked", "div1 clicked"

14. Uvažme následující fragment HTML kódu. Napište fragment JavaScriptu, který nastaví barvu pozadí (pomocí CSS) elementu `<div>` s `id="text"` na červenou barvu. [5 bodů]

`<div id="text"> ... </div>`

JavaScript: `document.getElementById("text").style.backgroundColor = "red";`

15. Uvažme modelovou situaci: Na webové stránce máme formulář, jehož úkolem je umožnit uživateli vložit novou položku do databáze. Formulář je zpracováván na straně klienta JavaScriptem a údaje z něj jsou odesílány pomocí asynchronního HTTP požadavku (AJAX) na server, kde je zpracuje PHP skript. Která z následujících tvrzení musí bezpodmínečně platit, pokud je celá aplikace naimplementována v souladu s normami?

[2 body/odpověď]

- ☒ Asynchronní HTTP požadavek je prováděn metodou POST (eventuálně PUT).
- ☐ PHP skript musí jako odpověď vygenerovat HTML stránku, která se zobrazí místo formuláře.
- ☐ Odesílaná data musí být kódována buď ve formátu XML, nebo ve formátu JSON.
- ☐ Skript na straně serveru nesmí data přímo uložit, pouze ověří, zda je možné data zapsat. Formulář musí být následně odeslán standardním způsobem (spuštěním jeho události submit).
- ☒ Během zpracování asynchronního požadavku může uživatel se stránkou libovolně interagovat, dokonce může způsobit, že prohlížeč začne načítat jinou stránku (kliknutím na odkaz, na tlačítko "zpět", ...).

16. Které vlastnosti má protokol HTTP 1.1? Vyberte pravdivá tvrzení:

[3 body]

- ☐ Pro každý požadavek navazuje nové TCP spojení a po obslužení požadavku spojení ukončí.
- ☒ Protokol je bezstavový, každý požadavek zpracovává nezávisle na předchozích a souběžných dotazech.
- ☒ Protokol obsahuje podporu pro částečné dotazy (specifikující požadovaný rozsah bytů výsledného obsahu), aby bylo například možné navázat na přerušené stahování.

17. Mezi datové typy PHP patří:

[1 bod/odpověď]

- | | | |
|---|--|---|
| <input type="checkbox"/> number | <input checked="" type="checkbox"/> string | <input checked="" type="checkbox"/> boolean |
| <input checked="" type="checkbox"/> integer | <input type="checkbox"/> function | <input checked="" type="checkbox"/> array |

18. Řetězcové literály je v PHP možné zapisovat do uvozovek ("") nebo apostrofů (' '), přičemž platí:

[2 body/odpověď]

- ☐ Mezi těmito dvěma zápisy není žádný rozdíl, jedná se čistě o pohodlnost pro programátory.
- ☐ Mezi těmito dvěma zápisy není žádný rozdíl, avšak použití apostrofů je označené jako zastaralé a v některé z budoucích verzí PHP bude odstraněno (nemělo by se již používat).

- ☒ Při zápisu do uvozovek se nahrazují speciální escape sekvence (např. `\n`, `\t`, ...), které vkládají znaky, které by bylo obtížné/nemožné vložit jinak.
- ☒ Při zápisu do uvozovek se nahrazují proměnné (zapsané v syntaxi PHP) jejich obsahem.

19. Doplňte informace, které budou platné při spuštění skriptu `index.php` při zpracování HTTP požadavku s následujícím URL. [3 body/položku]

`index.php?page=welcome&offset=10&item[0]=a&item[1]=b&item[2]=c&last=`

- Kolik položek obsahuje pole `$_GET` ? 4
- Jakého typu je položka/položky `$_GET['item']` ? array
- Jakou hodnotu jakého typu má položka `$_GET['last']` ? string("")

20. Máte fragment prokládaného HTML a PHP kódu. Jak bude vypadat výsledné HTML poslané klientovi? (Bílé znaky a přesné formátování neřešte.) [4 body]

```
<html><body>
  <ul>
    <?php
      $data = array("first", "second", "third");
      for ($i = 1; $i < count($data); ++$i) { ?>
        <li><?= $data[$i]; ?></li>
      <? } ?>
    </ul>
  </body></html>
```

Řešení:

```
<html><body>
  <ul>
    <li>second</li>
    <li>third</li>
  </ul>
</body></html>
```

Celkem _____ bodů z maximálního počtu 100 bodů. Výsledná známka: _____.