Omtentamen i kurs JavaScript med jQuery - EMMFEH17.

Datum: 2018-05-20

Tid: 7h.

Hjälpmedel: Dator med valfri utvecklingsmiljö.

Förberedelse

Skapa ett nytt repo på Github med namnet *tentamen-blog*.

Klona repot och kopiera innehållet i tentamenspaketet - allt i mappen `contents` - till det klonade repot.

Inlämning

När tentamen är färdig för inlämning:

- *add* + *commit* + *push* alla ändringar i ditt repo till Github.
- Skapa en tag för repot (= lås tentamen) så här:

  git tag exam-submitted
  git push origin –tags

Miljö

Innehållet i tentamenspaketet är följande:

- `/g`

  I `index.html`, `app.js` och `styles.css` i denna mapp ska klientkod läggas som ska uppfylla krav för Godkänd.

- `/vg`

  I `index.html`, `app.js` och `styles.css` i denna mapp ska klientkod läggas som ska uppfylla krav för Väl Godkänd. I din kod i `app.js` kommer du använda variablerna `curses` och `FormValidator` som är tillhandahållna via filerna `curses.js` respektive `validator.js`.

- `demo.gif`

  En GIF-animation som visar hur en fullständig VG-lösning skulle kunna se ut.

Installation och skript

Vi har inget backend denna gång, så ingenting behöver installeras.

Öppna `index.html` - för respektive betygsnivå - i webbläsaren för att se resultatet av skriven klientkod.

Beskrivning

Detta praktiska prov syftar till att påvisa studentens kunskaper i JavaScript och jQuery genom att dels bygga en enkel webbapplikation - webshop, dels implementera formulärvalidering.

Kravspecifikationen för att få Godkänd respektive Väl Godkänd är som följer:

- *Godkänd*
  - [A] Hämta postdata från ett API.
  - [B] Slumpa fram antal likes för varje post som har hämtats (detta finns ej i produktdata från API:et).
  - [C] Vy: *Postlista* - visa alla poster med *titel*, *innehåll* och *likes*.
  - [D] Varje post har möjlighet att också ladda alla *kommentarer* för den posten. Dessa ska sedan visas i anslutning till posten.
- *Väl Godkänd*
  - [E] Vy: *NyPost* - Formulär för att skriva in en ny blogpost. Man ska ange *title* och *content*.

    Då formuläret “submittas” ska det valideras. Statustext ska sedan visas i vyn, där status är antingen att validering lyckades eller en lista på samtliga valideringsfel.

Implementation

Allmänt gäller att

- CSS/styling är inte obligatoriskt (om du inte använder den för att implementera navigeringen mellan vyer på VG-nivån).
- Ingen felhantering vid AJAX anrop behövs.

Notera att fler sidor än de `index.html` som redan finns *ej* ska skapas.

(För Godkänd finns bara en vy - för Väl Godkänd ska man programmatiskt kunna växla mellan de två vyerna)

Nedan följer implementationsdetaljer för några av kraven A-E ovan.

- [A]

  Blogposterna hämtas från siten JsonPlaceHolder. Se dess sida för vilken URL man kan nå `posts` ifrån.

- [B]

  För varje produktobjekt måste en ny egenskap “likes” sättas; egenskapen ska ha ett slumpmässigt värde mellan 0 och 100. Använd följande kod:

  ```
  Math.floor(Math.random() * 100)
  ```

  Observera: likes sätts (slumpas fram) varje gång sidan laddas och behöver inte sparas mellan sidvisningar.

- [D]

  JsonPlaceHolder har två olika URL:er för att hämta kommentarer för ett givet `postId`. Använd vilket som av dessa.

  Varje enskild post ska ha möjlighet att hämta relaterade kommentarer. Här används med fördel jQuery's Event Delegation syntax.

  När kommentarerna anländer ska de visas i anslutning till just den blogposten. Det är viktigt att du hämtar kommentarer till rätt blogpost!

- [E]

  Skapa HTML för ett formulär för att mata in data om en ny produkt. De fält som ingår - och den validering som krävs i JavaScript - är:

  - *title* - Måste innehålla minst 5 tecken.
  - *content* - Måste innehålla minst 5 tecken, och får inte innehålla något av orden som finns i listan i variabeln `curses` (som skapas i filen `vg/curses.js`).

  När användaren submittar formuläret så ska datan valideras med hjälp av `FormValidator` som definieras i `vg/validator.js`. Se den filen för en detaljerad beskrivning.

  Alla fel som uppstår ska visas på skärmen. Om allting är ok ska ett meddelande om det också visas.

  Observera: Ingen POST behöver göras, vi sparar inte datan någonstans.