**Dokumentation – projekt TodoMVC**

Projektet gick ut på att återskapa webbsidan [TodoMVC](http://todomvc.com/examples/backbone/) med det utseende och dom funktioner som finns där. Vi har valt att använda oss av renodlad HTML, CSS och JavaScript.

**Arbetssätt**

Vi genomförde projektet så att vi satt tillsammans och planerade hur vi skulle lägga upp arbetet med projektet och hur vi skulle implementera dom olika funktionerna på bästa sätt. Vi satt också och skrev det mesta av koden gemensamt på våra ”studiedagar”. Vi delade alltså inte upp projektet sinsemellan utan gjorde det mesta gemensamt. Endast mindre ändringar och kodskrivningar gjordes enskilt i hemmet. På det sättet kunde vi lära av varandra och jobba mot ett och samma mål.

**HTML**

Vi började med att skapa sidans innehåll med HTML. Vi har försökt att göra det så semantiskt som möjligt med en <head> och en <body>. I <body> har vi <header>, <main> och <footer>. Det som skall synas på webbsidan har vi lagt i <body> i lämpliga element t.ex. <form>, <div>, <ul>, <label>, <button> mm. I HTML läser vi in en stylesheet (”index.css”) som hanterar större bildskärmar och en annan stylesheet (”mobile.css) som hanterar mindre skärmar (se nedan). Vi gör också en koppling till vår JavaScript-fil (”index.js”). Vi har även lagt till kod för att göra sidan responsiv.

I HTML har vi inaktiverat autocomplete i textfältet där man skriver in nya Todo’s.

Vi skapar en ”todo-item-blueprint”-klass som vi använder för att klona den aktuella Todo-texten som skrivs in så att vi lättare kan hålla reda på den. Denna klass sätter vi till ”display none” i CSS.

Vi skapar både en ”todo-textbox” och en ”todo-label” som båda kommer att innehålla samma todo-text. Todo-textboxen är från början dold men synliggörs vid dubbelklick så att todo-texten skall bli redigerbar. När redigeringen är gjord sparas texten även till todo-labeln som sedan är det som visas på skärmen.

Filter-knapparna (All, Active och Completed) har vi lagt i en <ul> och gjort till radio-buttons.

Vi har använt oss av egna ikoner för att markera/avmarkera alla Todo’s och för att ta bort enstaka Todo’s. Vi har använt oss av tecknen ”❯” och ”**×**” som vi sedan v.b. roterat med CSS.

Med ”aria-label” har vi också lagt till visst stöd för skärmläsare t.ex. så att när man hovrar över ”ta-bort”-knappen så läses ”remove knapp” istället för bara ”knapp”.

**CSS**

Efter att vi skapat innehållet med HTML så fixade vi till utseendet på sidan med CSS. Vi har en CSS-fil (”index.css”) som stylar fullstora skärmar och en fil (”mobile.css”) som stylar mindre skärmar med max-bredd 850px. I praktiken bestämmer den sistnämnda utseendet på mobiler inkl. när dessa är i ”landskapsvy”. Här har vi tagit bort hovrings-möjligheterna eftersom man inte kan hovra på mobilskärmar. Vi har även gjort så att Todo-överskriften glider upp och ut ur skärmvyn om man matar in många Todo’s. Utseendet på mobiltelefoner testade vi med DevTools.

På fullstora skärmar har vi försökt att efterlikna original-sidan så mycket som möjligt. Utseendet med "pappershög"-effekten löste vi med skuggningar i form av ljusa, omväxlande med mörka skuggningar som vi försköt nedåt i förhållande tillvarandra.

Vi har också använt oss av display-flex och display-grid för att få elementen att hamna som vi önskade i förhållande till varandra.

Med ”hover” har vi gjort så att ”ta-bort”-knappen blir synlig när man hovrar över Todo-labeln och blir mörkare när man hovrar över själva ”ta-bort”-knappen. Dessutom har vi lagt ”hover” på ”filter-knapparna” (All, Active och Completed) så att det dyker upp en ”border” runt knappen och som blir mörkare vid klick på den. Även länkarna i footern har vi lagt ”hover” på.

**JavaScript (JS)**

När vi var klara med HTML och CSS gick vi vidare med JS. Den börjar med en ”start-funktion” som alltid körs initialt och där bl.a. data från LocalStorage (LS) läses in och anpassar utseendet på skärmen utifrån det som är sparat. Här har vi också lagt in diverse ”EventListners” som skall känna av interaktioner med sidan.

Därefter har vi funktionen ”createNewTodo” som skapar en ny Todo och lägger till den på sidan. Här finns också ett antal ”EventListners” som t.ex. känner av dubbel-klick i labeln för redigering av dess innehåll (se ovan).

Funktionen ”updateLocalStorage” anropas från ett flertal ställen i programmet, så fort en ändring av sidans innehåll skett. Denna funktion läser in det som ”syns” på skärmen och sparar det till LS. Till LS har vi valt att skapa två key-value-par. Den första, ”text”, spara innehållet i Todo-s och den andra, ”state”, sparar om Todo-n är ”active” eller ”completed”.

Därefter kommer ett antal funktioner som vi anropar vid klick på olika knappar, uppdaterar ”nr items left”, uppdaterar sidans utseende beroende på om Todo är ”active” eller ”completed” mm.

**Mål**

På fullstora skärmar har vi försökt att efterlikna original-sidan så mycket som möjligt både utseende- och funktionsmässigt och har nog lyckats med det till största delen. Vi har gjort många manuella testningar för att hitta ev. skillnader som vi sedan åtgärdat. Den tydligaste och mest uppenbara skillnaden mot originalet är att vi har lagt in ett påskägg i vårt program. Försök aktivera den om du kan! I mobilversionen har vi för användarvänlighetens skull gjort en del anpassningar av layouten som skiljer sig från originalsidan men funktionaliteten är densamma.

**VG-kraven**

Vi har även försökt att implementera VG-kraven i projektet. URL-hantering med ”hashchange” gör vi i ”start”-funktionen i JS. Vi har implementerat LocalStorage. Man kan editera Todo-s med dubbelklick beskrivet enligt ovan. ”Pappershögs”-effekten har vi fixat i CSS med skuggningar, också beskrivet ovan. Vi har också lagt till egna ikoner istället för ”substituten” som var länkat till på projektsidan.