Hangman

Innføring i programmering

Natalie Smebye

2020

Deleksamen 4

I denne deleksamen skulle vi lage vårt eget spill fra bunn av. Jeg valgte da å lage hangman-spillet da det er dette spillet jeg kan best av de vi fikk å velge mellom.

HTML og CSS

Jeg startet med å sette opp et HTML-dokument. I bodyen la jeg så til en h1-tagg med spillets navn, og en p-tagg under som sa litt om spillet. Under der plasserte jeg en image-tagg hvor startbildet med den tomme galgen blir vist. Jeg valgte å endre navnet på dette bildet, grunnen til dette kommer vi til å se senere. Under der la jeg til en tom div-tagg med id knapper. I denne delen har jeg senere i script lagt til alfabetet. I div-taggen under vil vi ved hjelp av javascript legge til de tomme strekene hvor ordet skal bli fylt inn. Under der igjen vil gjenstående liv bli vist, og nederst på siden har vi en spill-igjen knapp. I CSS-en har jeg også lagt til media queries, slik at spillet vil endres dersom man spiller på telefon eller pc. Jeg har også slik at alfabetet ikke kan markeres, dette er noe jeg foretrekker selv.

Javascript

Jeg har startet koden min med en funksjon som jeg kaller window.onload. Onload er en event som vil utføre javascripten så fort siden har blitt lastet inn. Jeg har derfor plassert resten av koden inn i denne funksjonen. Under der har jeg lagt inn en del variabler. Jeg startet med å lage en array for alfabet, med alle bokstavene i alfabetet i den. Deretter lagde jeg en variabel for ordet spilleren skal gjette. Løsningsord arrayen inneholder ordene spilleren skal gjette. Deretter har jeg en variabel for antall liv, og en annen for antall riktige.

Videre lagde jeg en liste hvor bokstavene kom til å bli plassert når spilleren har valgt riktig. Lagde først en variabel kalt gjetteFeltListe, og brukte document.createElement for å lage en uordnet liste. Deretter satte jeg en id på den kalt mittOrd. Så hentet jeg gjette-felt diven fra html-koden og satte listen i den ved hjelp av . Jeg lagde også en uordnet liste for bokstavene i alfabetet som skal trykkes på. Jeg ga variabelen bokstaver, id alfabet, og hentet div taggen kalt knapper, og i den skal altså bokstavene plasseres. Sist hentet jeg siste elementet fra html-koden kalt liv. Her vil antall liv vises ved hjelp av variabelen visLiv.

Det første jeg begynte med var å lage bokstavene som spilleren kunne trykke på.  Jeg kalte funksjonen knapper, og i den lagde jeg en for løkke som ville gå gjennom alfabet arrayen jeg lagde tidligere i koden. Innenfor den lagde jeg en del instrukser. Jeg lagde en lokal variabel kalt bokstavElement ved hjelp av let. Jeg lagde så en liste som kommer til å bli plassert i HTML-koden ved hjelp av innerHTML propertyen, der vil da alle bokstavene fra alfabet arrayen bli skrevet ut. Siste instruksen vil være at vi kjører sjekk funksjonen, men vi kaller på bokstavElement slik at den kjører riktig variabel.

Neste funksjon er kalt resultat, i denne vil vi også kjøre en for løkke. Poenget med denne er å fylle gjetteFeltListe. Det vil si at vi skal lage understreker som skal gjenspeile antall bokstaver i ordet som gjettes. Vi bruker derfor variabelen ord i for løkken. Deretter lager vi en liste kalt gjetteFelt. Dette gjetteFeltet vil endre HTML-koden ved hjelp av innerHTML, og vil plassere ut understreker. Listen gjetteFelt vil så bli plassert i den uordnede listen gjetteFeltListe som vi lagde tidligere, ved hjelp av appendChild.

OppdaterHTML funksjonen vil som navnet tilsier oppdatere HTML-koden. Først lagde jeg en string som vil endre på p taggen «liv» ved bruk av innerHTML. Den vil da skrive ut hvor mange liv som gjenstår ved hjelp av konkatinering av både tekst og variabelen liv. Under den la jeg til en if test. Denne sier da hvis liv er mindre enn 1 så vil det skrives ut at spillet er over, der hvor antall liv sto tidligere. Bokstavene vil da ikke kunne trykkes på lenger, dette ved å skrive inn koden:. Deretter skal hele ordet vises slik at spilleren kan se det. Dette gjorde jeg ved hjelp av en for løkke. Instruksen den for løkken da fikk var å fylle ut gjetteFeltListe sitt barn, altså gjetteFelt, som er listeelementet, med ordet spilleren egentlig skulle gjette. Koden så da slik ut: Dette vil da bli skrevet ut til gjette-felt div taggen. Dersom spilleren derimot gjetter riktig så vil neste del av if testen ordne det. Den sier da at dersom antall riktige bokstaver er lik bokstavene i ordet så vil det skrives ut i liv-paragrafen at spilleren vant, og bokstavene vil også forsvinne. Under if testen så endret jeg på bildet som viste hangman. Først lagde jeg en variabel kalt bilde, den satte jeg til å bli en string. Den ville skrive ut filnavnet til bildene, men den kom til å endre på nummeret til bildene ved hjelp av variabelen liv. Spillet har 6 liv, og da satte jeg 6 minus liv. Start bildet ble derfor endret til navnet hangman\_feil0, istedenfor hangman\_start. Stringen ble så sendt til img taggen i HTML-koden som jeg ga id’en hangmanImg. Dersom spilleren så mister et liv vil bildet oppdateres, hvis spilleren ikke mister et liv vil bildet forbli det samme.

I neste funksjon kalt sjekk, setter jeg inn en parameter kalt knapp. Deretter lager jeg en onclick funksjon, som vil kjøres når den ene knappen i spillet blir trykket på. Det jeg begynte med da var å hente ut bokstaven som ligger i knappen. Jeg laget da en lokal variabel kalt trykketbokstav, og satte den til å være knapp.innerHTML. Deretter satte jeg en class på knapp, som jeg kalte aktiv. I CSS koden har jeg lagt til en stil, for hva som vil skje dersom knappen blir trykket på. Den har fått økt opasitet, og vil dermed bli litt grå, og jeg har også brukt transition som bestemmer hvor fort knappen går fra slik den er utrykket, til sånn den blir når den er trykket på. I scriptet mitt fortsatte jeg så med å skrive inn koden: . Dette betyr at når knappen blir trykket på så vil ikke den fungere, på grunn av «null». Null er at det mangler en objekt-verdi, denne er satt med mening. Videre lagde jeg en for løkke, og i den lagde jeg en if test som sa dersom variabelen ord er lik trykketBokstav, altså om bokstavene i det tilfeldige ordet er lik bokstaven som blir trykket på, så vil gjetteFelt bli fylt ut med denne bokstaven dette på grunn av koden: . Antall riktige bokstaver i ord vil da bli pluss 1, ved hjelp av Under for løkken lagde jeg en variabel kalt bokstavIndex, deretter satt jeg den til å være lik:. Dette betyr at den vil lete gjennom ordet som er valgt for å se om den bokstaven som er trykket på er i ordet. Jeg lagde så en if test som sa at dersom bokstavIndex er lik minus 1 så vil man miste ett liv. Altså når bokstaven som blir trykket på ikke er i ordet så vil man miste ett liv. Det siste som skjer i funksjonen, er at jeg kaller på oppdaterHTML funksjonen.

Neste funksjonen er spill-funksjonen, den inneholder alt spillet trenger av regler og utgangspunkt. Den begynner med å velge et tilfeldig ord fra løsningsOrdet arrayen, og det ordet vil da havne i ord variabelen. Dette gjennom koden: Deretter kalte jeg på knapper funksjonen. Jeg satte så antall liv i spillet til å være 6, og antall riktige bokstaver man begynner med er 0. Til slutt kalte jeg på resultat- og oppdaterHTML funksjonen slik at disse også blir kjørt.

Til slutt lagde jeg reglene for spill-igjen knappen. Jeg startet med å hente ut knappen fra HTML-koden, og skrev så ut at når den blir trykket så vil en funksjon kjøres. I den funksjonen vil gjetteFeltListe forsvinne, og det vil også bokstavene. Til slutt vil spill-funksjonen kalles, slik at spillet vil starte på nytt med alle reglene og alt det trenger.