BRUKERINPUT

CSS for brukerinput

Nå har vi sett på noen av HTML-elementene vi kan bruke for å ta inn brukerinput. Resultatet blir ikke spesielt pent; nå skal vi se på hvordan vi kan forbedre utseendet med CSS.

Som nevnt tidligere er det lurt å lage CSS-dokumenter (stilark) som kan gjenbrukes. Det samme gjelder her. Hver gang du skal ha brukerinput på en side, bør du ha et stilark som du kan bruke.

I eksemplene som følger, kommer vi til å bruke den samme generelle CSS-koden som vi har brukt tidligere:

```
body {
  background-color: #f5f5f5;
  font-family: Georgia, serif;
}
#hoved {
  width: 60%;
  margin: 50px auto;
  padding: 40px;
  background-color: white;
  border: 1px solid #cccccc;
}
h1, h2, h3 {
  font-family: Arial, sans-serif;
}
```

Den eneste forskjellen er at vi også har lagt til `h3` .

La oss først ta en titt på et skjema der vi ikke har brukt CSS på elementene i det hele tatt:

Det er ganske åpenbart at vi trenger litt CSS her.

Tekst, tall og nedtrekksmenyer

Du har kanskje allerede lagt merke til at et skjema består av flere `<input>` -elementer. Det betyr at vi ikke kan bruke `input` som CSS-selektor fordi vi ønsker at forskjellige `<input>` -elementer skal ha ulikt utseende.

Vi må innføre en ny selektorvariant for å få tak i de ulike elementene. Vi kan nemlig spesifisere *både* hva slags element vi vil velge, *og* en av attributtene til elementet. Vi kan derfor bruke selektoren `input[type="text"]` for å gi CSS til `<input>` -elementer med attributtet `type="text"` . Denne skrivemåten kan brukes på alle HTML-elementer med attributter, for eksempel vil selektoren `img[src="bil.jpg"]` velge alle bilder der `src="bil.jpg"` .

Tekstfelter, tekstbokser, tallfelter og nedtrekksmenyer kan alle få det samme generelle utseendet, derfor kan vi starte med CSS-koden nedenfor:

```
input[type="text"],
input[type="number"],
textarea,
select {
  /* CSS-koden kommer her */
}
```

Her velger vi `<input type="text">` , `<input type="number">` , `<textarea>` og `<select>` . La oss først legge til CSS-kode som rydder opp i elementenes plassering:

```
display: block;       /* elementene havner på egne rader */
margin-top: 2px;      /* mellomrom til elementet ovenfor */
margin-bottom: 10px; /* mellomrom til elementet nedenfor */
```

Nå havner beskrivelsene våre ovenfor tekstfeltene, og det blir litt luft mellom elementene. Vi tar også med oss denne koden for merkelappene (`<label>`) ved avkrysningsbokser og radioknapper:

```
.checkboxradio > label {
  display: block;
  margin-top: 2px;     /* mellomrom til elementet ovenfor */
  margin-bottom: 10px; /* mellomrom til elementet nedenfor */
}
```

Her bruker vi `display: block;` som tidligere, slik at merkelappene havner nedenfor hverandre. Da havner også avkrysningsknapper og radioknapper mellom merkelappene, og dermed nedenfor hverandre. I tillegg legger vi til litt luft ovenfor og nedenfor elementene.

Utseendet er allerede en del bedre:

HTML CSS Re

LIVE

```css
body {
  background-color: #f5f5f5;
  font-family: Georgia, serif;
}
#hoved {
  width: 60%;
  margin: 50px auto;
  padding: 40px;
  background-color: white;
  border: 1px solid #cccccc;
}
h1, h2. h3 {
  font-family: Arial, sans-serif;
}
input[type="text"],
input[type="number"],
textarea,
select {
  display: block;
  margin-top: 2px;
  margin-bottom: 10px;
}
.checkboxradio > label {
  display: block;
  margin-top: 2px;
  margin-bottom: 10px;
}
```

Men det er fremdeles mye vi kan gjøre. La oss pynte litt på elementenes utseende:

Resources 1×

```
font-family: inherit;                    /* arver nettsidens skri
color: #666666;                          /* mørk grå tekstfarge *
border: 1px solid rgba(0,0,0,0.1);       /* lys grå kantlinje */
border-radius: 2px;                      /* avrundet kantlinje */
padding: 5px;                            /* luft inni tekstbokser
box-shadow: 0 1px 2px rgba(0,0,0,0.7);   /* skygge på tekstbokser
box-sizing: border-box;                  /* teller med kantlinjer
```

Her lar vi tekstboksene bruke samme skrifttype som resten av nettsiden (`inherit`), med en mørk grå tekstfarge. Vi legger også til en lys avrundet kantlinje og litt skygge. Den siste linjen (`box-sizing: border-box;`) lar oss medregne kantlinjen i boksenes bredde. Denne er nødvendig hvis vi ønsker å gi boksene bredden 100 %. Hvis vi ikke har med denne, vil tekstbokser med bredde 100 % bli bredere enn `<div>` -elementet vi har plassert dem i.

Nå ser skjemaet vårt slik ut:

Tekstboksene har blitt fine, men de er litt smale. For at skjemaet skal være lett å fylle ut, bør spesielt tekstfeltene bli litt bredere. Vi kan løse det med denne koden:

```
input[type="text"] {
  min-width: 130px; /* aldri smalere enn 130 px */
  width: 60%;
}
input[type="number"], select {
  min-width: 130px;
  width: 30%;
}
textarea {
  min-width: 130px;
  width: 60%;
  height: 60px;
}
```

Legg merke til at vi her skiller mellom de ulike elementene.

Her gir vi først tekstfeltet bredden 60 % og en minimumsbredde som gjør at elementet aldri blir smalere enn 130 piksler bredt. Vi gjør tilsvarende for tallfeltet og nedtrekksmenyen, men der setter vi bredden til 30 % i stedet for 60 %. Tekstboksen får samme størrelse som tekstfeltet, og en høyde på 60 piksler i tillegg.

Husk at disse breddene forholder seg til elementet vi plasserer innholdet vårt i. Hvis du for eksempel ønsker å lage et eget `<div>` -element der du plasserer disse elementene, kan du angi 100 % som bredde for flere av elementene, og styre bredden ved å justere `<div>` -elementets bredde.

Resultat vårt så langt ser slik ut:

Avkrysningsbokser og radioknapper

Vi har allerede gjort litt med disse elementene, men vi kan gjøre dem enda tydeligere, slik at det blir lettere å se *hvor* man kan trykke for å velge en avkrysningsboks eller radioknapp.

Vi starter med å definere en bredde for `<div>` -elementet som inneholder de andre elementene (`.checkboxradio`). Videre utvider vi CSS-koden for `<label>` -elementene som er inni dette `<div>` -elementet. Vi legger til luft på innsiden av

dem, litt luft nedenfor, en lys bakgrunnsfarge, litt avrunding av hjørnene, og vi sørger for at musepekeren forandrer seg når den føres over en `<label>` .

```
.checkboxradio {
  min-width: 130px;
  width: 30%;
}
.checkboxradio > label {
  display: block;
  margin-bottom: 8px;
  padding: 5px;
  background-color: #e4e4e4;
  border-radius: 2px;
  cursor: pointer;
}
```

Nedenfor kan du se hvordan det blir. Legg merke til at vi kan klikke hvor som helst i det grå området, fordi det er et `<label>` -element. Legg også merke til hvordan musepekeren forandrer seg når du holder over de grå områdene.

Nå begynner det å ligne noe! Da er det bare knappen som gjenstår.

Knapper

Koden nedenfor er et forslag til et utseende for knapper. Du kan selv velge hvor mye du ønsker å ta med av denne koden. Mange av egenskapene er selvforklarende, men vi har innført en litt spesiell bakgrunnsfarge her. Den litt avanserte koden

```
linear-gradient(rgba(255,255,255,0.2), transparent)
```

lager en rett (lineær) gradient fra en nesten gjennomsiktig hvitfarge (`rgba(255, 255, 255, 0.2)`) til fullstendig gjennomsiktig (`transparent`). Gradienten kommer i tillegg til den blå bakgrunnsfargen. Den kan altså brukes oppå en hvilken som helst farge. Resultatet er at knappen blir litt lysere på toppen.

```
button {
  padding: 10px;                                /* luft inni knappe
  color: white;                                 /* hvit tekstfarge
  text-shadow: 0 -1px 1px #335166;              /* skygge på tekste
  border: 1px solid rgba(0, 0, 0, 0.1);         /* lys grå kantlin
  border-radius: 2px;                           /* avrunder kantene
  background: linear-gradient(rgba(255, 255, 255, 0.2), transpar
  background-color: #5588AA;                    /* bakgrunnsfarge *
  box-shadow: 0 1px 2px rgba(0, 0, 0, 0.7);     /* skygge bak knapp
  outline: none;                                /* fjerner ekstra k
  cursor: pointer;                              /* gjør om musepeke
}
```

Vi kan også legge til en effekt når knappen blir trykket på. Det gjør vi med CSS-selektoren `button:active` . Her har vi brukt `transform` , som flytter knappen litt mot høyre og litt ned. Det gir et inntrykk av at knappen blir trykket ned.

```
button:active {
  transform: translate(1px,1px); /* 1px til høyre og 1px ned */
}
```

I dybden

transform

I koden ovenfor har vi brukt egenskapen `transform` . Dette er en egenskap med et stort antall muligheter. Vi kan for eksempel skalere eller rotere elementer. Verdien vi har brukt ovenfor, `translate()` , lar oss skyve et element til et annet sted. Dette skjer uten at andre elementer påvirkes, det vil si at elementet vi flytter på, ikke dytter på andre elementer.

Hvis du er interessert i å lære mer, kan du søke på «CSS transform» på internett.

Nå kan vi bruke CSS-koden vi skrev for den blå knappen, til å lage knapper med flere farger. Det er vanlig å bruke grønn knapp for å godkjenne noe og rød knapp for å avbryte. Vi kan derfor lage to CSS-klasser som er utvidelser av CSS-koden vi allerede har skrevet. Klassene heter `ok` og `avbryt` , og for å bruke dem skriver vi:

```
<button class="ok">OK</button>
<button class="avbryt">Avbryt</button>
```

CSS-koden for de to knappene kan du se nedenfor. Det er bare bakgrunnsfargen som er endret.

```
button.ok {
  background-color: #66BB00; /* grønn */
}
button.avbryt {
  background-color: #CC0000; /* rød */
}
```

Her er det egentlig nok å bare skrive `.ok` og `.avbryt` , men det kan være greit å spesifisere at disse klassene hører til `button` -elementet, slik som vi har gjort her.

Utseendet på de tre knappene blir slik:

Hvis du heller ønsker at knapper skal legge seg nedenfor hverandre, kan du legge til `display: block;` for knapper også. Det kan også være aktuelt å legge til `margin` ovenfor knappene slik at de ikke kommer altfor tett opptil andre elementer. Og tilsvarende for alle andre egenskaper vi har sett på her. Du lager dine egne nettsider, og vi har alle ulik smak. Lag et design som du synes er fint, men sørg for at det også blir brukervennlig.

Oppgave

7 Fullfør «visittkortgeneratoren» din ved å legge til CSS. Samle all CSS-kode som hører til brukerinput, i et eget CSS-dokument som du for eksempel kaller **brukerinput.css**. Dette

kan du gjenbruke senere og dermed spare mye tid.

I dybden

Gjenbruk av stilark

Vi har tidligere nevnt verdien av å lage egne stilark for ulike HTML-elementer. Her møter vi et godt eksempel på en slik situasjon. Hvis du lager et gjennomtenkt stilark som gir utseende til ulike brukerinputfelter og knapper, vil du spare mye tid i framtiden når du skal lage nettsider med slike elementer. Alt du trenger å gjøre, er å legge til stilarket i koden din.