# Webbutveckling grunder

## Förberedelse

- Börja med att skapa ett repository på github.

- Gå till mappen projects på gitbash. Var noga med att du är i den mappen.
    - Klona sedan ner dit repository från github.

- Starta ditt favorit-redigerings program.
    - Jag föredrar Sublime text.
    - Finns även Notepad++

- Vi kommer att använda Chrome som webbläsare.
- Har du inte den så ladda ner den.
- Fixa sedan då du har Chrome på ena sidan av skärmen och din editor på den andra.
    - På så vis kan vi se resultatet direkt när vi kodar.

- Vi ska även ladda ner Visual Studio för när vi ska köra kursen ”Practical HTML 5”
    - Ni kan köra den kursen med samma verktyg som denna, men författaren använder Visual Studio. Då blir det lättare att köra övningsfilerna.

    - Gå till: https://www.visualstudio.com/en-us/products/visual-studio-community-vs.aspx
    - Ni behöver bara ladda ner för Webapplication.

## Inledning

När man jobbar som webbutvecklare brukar man jobba som antingen frontend eller backend.

**Frontend är:**

- Hur en sida ser ut. Den ska vara tilltalande och snygg.
- Gränssnittet ska vara användarvänligt.
- Det ska vara lätt att hitta på sidan.
- Tekniker för frontend är framförallt:
    - HTML – Själva innehållet på sidan.

    - CSS – Färger och former på sidan men också mer.

    - Javascript – Det som skapar interaktivitet.
        - Javascript är det som får saker att hända på sidan, t.ex. om du trycker på en knapp.

    - Som ni kommer att märka blandas teknikerna ibland.

**Backend är:**

- Det som sker ”bakom kulisserna”.
- Backend kan vara Java, PHP eller C# t.ex.
- Det ni kommer att lära er i andra delen av kursen.

**Klient och server (clientside and serverside)**

Man brukar tala om klient och server.

Klient är den dator du sitter vid lokalt. När du använder Javascript är det lokalt som det körs oftast. På så vis kan du få interaktivitet utan att behöva ha internetuppkoppling.

Server är oftast en avlägsen dator. Där sker t.ex. kopplingar till en databas och annat som behöver ske centralt. När du loggar in på en sida som Facebook t.ex. kan inte uppgifterna ligga lokalt på din dator. Facebook måste kunna kolla dina inloggningsuppgifter med alla andra i världen. Då måste uppgifterna sparas centralt någonstans.

# HTML

Gå till **example.com**. Det är en exempelsida för att visa hur en sida kan var uppbyggd.

Vi ska titta på hur sidan är uppbyggd med HTML-kod.

Högerklicka och välj ”visa källkod”.

Här ser vi hur koden är uppbyggd. (Förklara de olika delarna)

## Vår första sida

Se till att du har din editor på en skärm och din browser på den andra.

Börja skriv <html>. Då ska du få upp förslag bl.a. just html. Klicka på html, då ska du få upp en grundläggande html-sida.

(Gå igenom stegen igen)

Spara sidan i er git-mapp som du klonade.

Kalla sidan index.html.

*(När du har en sida online kommer browsern att leta efter en fil som heter index.html och starta den.)*

Nu ska vi testa lite saker för att få upp ångan.

- Prova att skriva lite mellan taggarna för body. Här finns huvuddelen av det vi skapar.
- Skriv hej eller vad du vill.
- Spara och i Chrome trycker du **ctrl + O** för att öppna en fil.
- Leta upp filens du sparade.

- Nu behöver du bara uppdatera Chrome varje gång du har gjort en ändring.
- Nu bör du se det du skrev i din kod.
- Ändra lite i det du skrev eller lägg till något.
- Spara och uppdatera Chrome igen.
- Nu ser du att texten ändras.

## Titel och svenska tecken

I huvud-taggarna finns titel-taggen.

- Skriv min **första sida** mellan titeltaggarna. Spara och uppdatera.
    - Du ser nu att åäö visas fel i titeln på sidan.
    - För att det ska visas rätt ska vi lägga till en metatagg.
    - **<meta charset="utf-8">** - utf-8 gör så att svenska tecken visas.
    - Spara och uppdatera igen. Nu bör det se rätt ut.

## Bara text

Ska man bara skriva text på en sida behöver man bara skriva något mellan <body>-taggarna. Prova!

Skriv något, spara HTML-sidan och uppdatera din browser sedan.

## Stycke

För att dela upp en text i stycken kan man skriva 

Prova att skriva två stycken med lite text.

Här skriver vi något intressant

Här skriver vi något ännu mer intressant

## Länkar

Bland det viktigaste som finns på Internet är länkar.
Inom <body> skriver vi en enkel länk.

```
<a href="http://www.aftonbladet.se">aftonbladet</a>
```

## Listor

För att skapa en lista utan särskild ordning skriver vi **<ul>** som står **för unordered list**.

Varje sak i listan skrivs <li>

Skriv detta i body:

```
<ul>
    <li>Skor</li>
    <li>Dator</li>
    <li>Korg</li>
</ul>
```

Ordnad lista skrivs **<ol>.** Annars ser det likadant ut.

```
<ol>
    <li>Skor</li>
    <li>Dator</li>
    <li>Korg</li>
</ol>
```

Detta visar en numrerad lista. Du kan ändra siffrorna m.m. genom att skriva **type**

T.ex.

<ol type=”I”>

Prova även lilla i, lilla L, stora och lilla a.

Listor används också i menyer när man gör menyer tillsammans med CSS.

## Bilder

Gör en folder i din mapp och kalla den t.ex. img.

Leta upp en bild, ladda ner den och lägg den i din mapp. (eller ta en på din dator)

Skriv följande tagg:

<img src=”img/koala.jpg”>

Behövs ingen slut-tagg.

Blev bilden för stor?

Vi kan begränsa med width=”100”. Bilden anpassar sig automatiskt.

# CSS

Nu ska vi testa lite CSS. Det används dels för att göra vår HTML snyggare, men också för att göra sidor lättare att förstå och använda.

Det finns tre sorters CSS:

## Inline – För att förändra en HTML-tagg

Vi ska nu prova att skriva lite CSS för att färga vår text. Det gör vi genom att skriva lite CSS i vår tagg. Kallas för *attribute.* Ibland vill man bara ändra på en särskild tagg.

** Vår text **

Lägg märke till att vi sätter CSS-koden som en del av första taggen.

Spara och uppdatera browsern.

Om det fungerade så prova att göra andra stycket rött t.ex. eller annan färg.

## Internal CSS – För att sätta stilen för hela dokumentet

Istället för att sätta CSS-kod för varje enskild tagg kan man bestämma hur t.ex. alla stycke-taggar ska se ut i hela dokumentet.

För att bestämma hur vår CSS ska styra hela sidan sätter man upp regler inom style-taggar. Oftast sätter man dem i HEAD-delen i vårt HTML-dokument.

```
<head>
    <meta charset="utf-8">
    <title>Min första sida</title>

    <style type="text/css">

        p {

            color:green;
        }


    </style>
</head>
```

Här sätter vi att alla stycketaggar ska vara gröna. Eftersom vi har satt annan CSS på våra stycketaggar så kommer inte detta att påverka dom. Ta bort era style-attribut eller från era p-taggar eller skapa nya.

Spara koden och uppdatera browsern.

Nu ska alla taggar som inte har ”inline-CSS” vara gröna.

## Klasser och id

Ett annat sätt att tilldela flera attribut som färg, storlek m.m. till en eller flera former är klasser och id:n

Skillnaden på klasser och id:n är att klasser kan användas på flera ställen medan id:n är unika och bara kan användas en gång. ID:n kan (i vissa fall) användas flera gånger men bör inte.

**Vi testar med id. Ett id föregås av en hashtag:**

```
<head>

    <style type="text/css">


        #oneTime {

            color:red;
            font-size: 2em;
            font-style: italic;

        }

    </style>
</head>

<p id="oneTime">Denna text styrs av ett id</p>
```

**Vi testar med klass. En klass föregås av en punkt:**

```
<head>

    <style type="text/css">


        .manyTimes {

            color:blue;
            font-size: 3em;
            font-style: normal;

        }

    </style>
</head>

<p class="manyTimes">Denna text styrs av klass</p>
```

**Vi kan också använda flera klasser på en gång:**

Vi skapar en ny klass som gör texten i fetstil:

```
        .boldText {

            font-weight: bold;

        }
```

Nu kan vi använda båda klasserna för att få text som är blå, 3em i storlek, normal och fet.

```
<p class="manyTimes boldText">Denna text styrs av klass</p>
```

## DIV och External Style Sheet

Div är en HTML-tag som används för att dela upp en sida i olika delar. Ni kommer att se sedan när ni lär er mer om HTML5 det finns andra taggar som fungerar som en div men har andra namn.

Man kan säga att en div skapar en behållare.

## Övning 1

Vi ska testa att skapa en sida som är uppdelad i olika delar.

Först ska vi dock prova den tredje varianten av CSS - **External Style Sheet**

Att lägga in CSS i varje dokument är oftast en dålig idé, åtminstone när man kör ”skarpt”.

Har man t.ex. en viss färg eller stil för alla rubriker måste man gå in på varje sida för att ändra.

En bättre idé är att bara ha ett ställe där man ska ändra på. För att lösa det använder man en separat sida där man har sin CSS.

Ta allting från mellan <style>-taggarna och lägg det i en ny fil.

Spara filen som t.ex. style.css. Tillägget css är det viktiga.

För att vi ska tala om att vi vill använda det dokumentet lägger vi in en länk i <HEAD>.

```
<link rel="stylesheet" type="text/css" href="style.css">
```

Det som är viktigt här är det som står vid **href**. Det är namnet på dokumentet vi länkar till. Har du sparat under annat namn så skriv det istället.

Spara och uppdatera browsern.

## Övning 2

Nu ska vi testa att skapa en sida uppbyggd med <DIV> och CSS. Gör en ny sida med HTML.

Vår HTML-sida ska se ut så här innanför <body>-taggarna:

```
<div id="container">
       <div id="header">
              Sidhuvud
        </div>

        <div id="content">
              Innehåll
        </div>

        <div id="sidePanel">
              Sidpanel
        </div>

        <div id="footer">
              Sidfot
        </div>
</div>
```

Spara och uppdatera

Vi har i vår HTML delat in vår sida i olika div:s. Utan vår CSS syns dom dock inte.

För att ange var div:arna ska vara, hur de ska se ut, vilken storlek de ska ha m.m. ska vi använda CSS.

## Skapa CSS-fil

Nu ska vi skapa en CSS-fil för att placera och formatera våra div:ar.

Skapa en fil som heter t.ex. styles2.css. Namnet är inte viktigt men glöm inte tillägget .css.

Glöm inte heller att ha det namnet i länken till filen.

CSS-kod som läggs i nya filen:

```
#container
{
     width: 673px;
     border: solid 1px black;
     margin: 0px;
}

#header
{
     color: #FFFFFF;
     width: 100%;
     height: 100px;
     background-color: #778CB3;
}

#sidePanel
{
    width: 213px;
    height: 372px;
    margin-top: 5px;
    background-color: #778CB3;
    color: #FFFFFF;
    padding: 4px;
}

#content
{
    width: 446px;
    float: right;
    height: 380px;
    margin-top: 5px;
    background-color: #A0AFCB;
}

#footer
{
    background-color: #F7CB33;
    padding: 12px;
    width: 649px;
    color: #000000;
    font-size: 90%;
    text-align: center;
    clear: both;
    border-top: 5px solid #FFFFFF;
}
```

Spara och uppdatera.

## Float

För att ni ska se hur float fungerar gör så här:

Markera bort den div som har id=”content”

Markera bort gör man i HTML med <! – och -->

Kallas rem för remark och används för att t.ex. sätta en kommentar som browsern struntar i. Fast som ni ser kan man även tillfälligt plocka bort kod.

I den div som har id=”sidepanel” skriver ni:

```
float: right;
```

Spara och uppdatera. Nu bör din panel vara till höger. Testa att skriva:

```
float: left;
```

Ta bort kommatering för att återställa.

### Float i meny

Float kan också användas i listor för att skapa en horisontell meny.

## Box-modellen

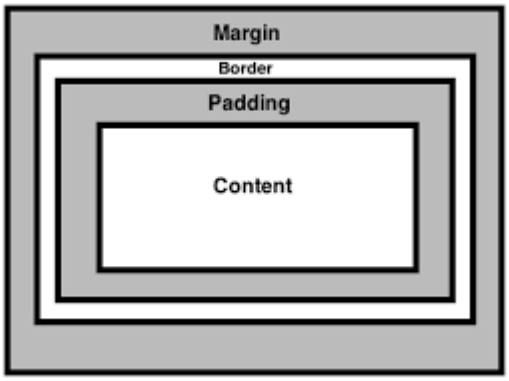
Boxmodellen används för att sätta ramar runt objekt och text i CSS.

http://www.w3schools.com/css/css_boxmodel.asp

Border är en ram som kan vara synlig om vi vill runt t.ex. en text.

Padding är mellanrum mellan texten och ramen som inte syns.

Margin är mellanrum utanför ramen som inte heller syns.



# Javascript

Javascript används för att göra sidor mer interaktiva. Att det händer något på sidan.

Det kan vara i ett separat dokument men är det i HTML-dokumentet ska det skrivas inom <script>-taggar:

<script>

</script>

Skapa en ny HTML-fil och lägg den i en egen folder.

Precis som med CSS kan Javascript vara inline, internal eller external.

## Inline

Skriv detta i <body> i din fil:

```
<button onclick="alert('Hello world')">Klicka här!</button>
```

Du ska då få en knapp och ett meddelande om du trycker på den.

## Internal

Lägg detta i body också.

```
<script>
    alert('Hello world');
</script>
```

Nu får du en alert så fort sidan laddas.

## Console och felsökning

Du ser inte när det blir ett fel på sidan. Du får inget felmeddelande.

Högerklicka på sidan och välj **inspektera**

Välj sedan console i övre menyn.

Gör ett medvetet fel och ladda sidan.

Du ska då få ett felmeddelande.

## Ändra text med javascript och DOM

```
<button id="myButton">Ändra text</button>

 <p id="text">Hello World</p>

 <script>

     document.getElementById("myButton").onclick = function() {
           document.getElementById("text").innerHTML = "Hello everybody";
     }

 </script>
```

Man kan komma åt ett element genom dess **id**.

Vår stycketagg har här id=text. Knappen har id=myButton.

- Först talar vi om att vi vill använda knappen genom att hämta den genom sitt id (myButton)
- Sedan talar vi om att vi vill använda metoden **onclick** som hör till knappen.
- När vi klickar ska vi använda en funktion som hämtar elementet med id = text vilket ju är vår stycketagg.

# JQuery

JQurey är ett ramverk som är Javascript och underlättar att använda Javascript.

För att JQuery ska fungera måste vi lägga en länk till det eller ladda ner det. Precis som med Bootstrap.

Vi lägger en länk istället för att ladda ner filer. Lägg denna i **<HEAD>** på din sida.

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/2.2.2/jquery.min.js"></script>
```

jQuery syntaxen är skräddarsydd för att välja HTML-element och utföra någoting med elementen.

Grundläggande syntax är: **$(selektor).handling()**

**Ex:** `$("#circle").click`

- Man startar med en dollarsymbol ($)
- En "selektor" för att fråga (query) välja HTML-element.
- En jQuery "handling" som ska utföras på elementen.

## En övning med Cirklar och kvadrater

Vi kan också manipulera <HTML> m.m. genom JQuery men på ett enklare sätt än med vanlig "vanilla" javascript.

Vi börjar med att skapa ett id för en cirkel och en klass för en kvadrat. Lägg detta i <head>

<style>

```
  #circle {
        width: 150px;
        height: 150px;
        border-radius: 50%; /* To create circle */
        background-color: green;
        margin: 10px;
  }

  .square {
        width: 150px;
        height: 150px;
        background-color: red;
        margin: 10px;
  }

.square2 {
        width: 150px;
        height: 150px;
        background-color: blue;
        margin: 10px;
  }
```

```
</style>
```

I <body> skriver vi sedan HTML för att visa en cirkel och två kvadrater.

```
<div id="circle"></div>

<div class="square"></div>

<div class="square"></div>
```

Sedan är det dags att skriva lite JQuery-kod.

Med jQuery väljer du HTML-element och gör något med dom

```
<script type="text/javascript">

	//Välj cirkeln att klicka på

	$("#circle").click(function() {
		alert("cirkeln är klickad");
	});

	//Välj kvadraterna att klicka på.

	$(".square").click(function() {
		alert("kvadraterna är klickade");
	});




</script>
```

Eftersom **circle** är ett CSS-id sätter vi tecknet ”#” framför.

Square är en klass i CSS och då sätter vi punk (.) framför.

Här är det alltså inte ett element med id som vi letar upp (som vi gjorde ovan i Javascript med **getElementById**).

Här är det ett id och en klass som i CSS som är kopplat till en div. Så det är div:en med ett viss CSS-id och div:en med en viss klass som vi letar upp.

Lägg in länk till JQuery, CSS och JQuery-kod på en sida.

Testa att klicka på kvadrat och cirklar.

Man kan också välja t.ex. att något ska hända när man klickar i en <DIV>

Kommentera bort den JQuery-kod du har (Inte länken alltså och inte CSS)

Lägg istället in följande kod:

```
$("div").click(function() {
	alert("Div:en är klickad");
});
```

Innan letade vi upp en viss div med en CSS-id eller CSS-klass som vi skulle klicka på.

Nu säger vi istället att alla div som vi klickar på ska starta en alert.

Prova koden. Nu ska både cirkeln och kvadraten ge en textruta med samma text.

Så nu ska alla div ge samma meddelande.

Testa genom att skapa en ny div och skriv t.ex. en text. Om du klickar på texten ska det ge samma meddelande.

**JQueryUI**

Det finns även plugins för JQuery. En sådan är JQureyUI som underlättar att skapa ett gränssnitt (UI) med JQuery.

Gå till JQureyui.com. Klicka på download i rutan och välj **Quick downloads** och **stable**.

Lägg den i samma mapp som JQuery-filen.

Packa upp filen. Det är en mapp som innehåller nödvändiga filer och exempel.

Kör index-filen i en webbläsare.

Titta på exemplen och testa.

Döp om helt mappen till jquery-ui och lägg sedan dessa länkar till jquery-ui.
Tänk på att sökvägen är rätt.

Lägg sedan dessa länkar under din JQuery-länk.

```
<link href=" jquery-ui/jquery-ui.css" rel="stylesheet">
<script src="jquery-ui/jquery-ui.js"></script>
```

Se till att den mappen ligger direct under din mapp.

Testa en pluggin.

Gör en ny div.

```
<div class="square2" id="draggable"></div>
```

Gör  en ny klass och lägg den bland de andra.

```
.square {
    width: 150px;
    height: 150px;
    background-color: blue;
    margin: 10px;
}
```

Skriv följande rad som queryUI-kod:

```
$("#draggable").draggable();
```

Du ska nu ha fått en blå fyrkant som går att flytta runt.

Testa!