Encoding:

Vi har tidligere set på spørgsmålet om et XML dokuments encoding. Det er generelt altid en god ide at gemme et dokument med en XML erklæring og en bestemt encoding! Dette er i al fald **nødvendigt** hvis der anvendes tegn som ikke er såkaldte ASCII tegn – altså 'engelske' tegn!

Et eksempel på et dokument **uden** en xml erklæring:

```
<data>I dette år besøges mange ærtesteder!</data>
```

Dette dokument kan ikke vises i en browser – der er et 'ugyldigt' tegn i position 15! Dette skyldes ikke at XML eller Unicode ikke forstår tegnet 'å' men at dokumentet bliver kodet forkert.

Hvis vi ændrer dokumentet og skriver en eksplicit encoding bliver det vist OK:

Men de forskellige XML parsere 'forstår' kun et begrænset antal kodninger! F. eks. findes en forbedret version af ISO-8859-1 som hedder ISO-8859-15 – men den forstås ikke af f. eks. Internet Explorer – selv om den er en officiel anerkendt encoding! Alle XML parsere SKAL kunne håndtere to slags encoding – UTF8 og UTF16 – men ikke nødvendigvis andre! Der kan være meget stor forskel på hvilke kodninger forskellige parsere kender.

Et tegn sæt (character set):

Et tegn sæt er en række af gyldige tegn som er definerede i det konkrete tegnsæt! Et tegn kan være et bogstav, et symbol som '_' eller et kinesisk skrifttegn! Unicode 3.1.1 definerer flere end 90.000 forskellige tegn og de er alle gyldige i et XML dokument! Alle Unicode tegn er definerer i en såkaldt Backus Naur produktion 'Letter'. Det er ikke sikkert at de kan skrives i Notesblok – men de er gyldige tegn i et XML dokument!

Alle tegn har en bestemt tal kodning – et såkaldt kode **punkt**! De første kode punkter er nr 0 til og med nummer 127. Bogstavet 'a' har kodepunktet 97 i ALLE tegnsæt! Uanset hvordan vi sætter vores encoding i xml erklæringen betyder 97 altid det samme!

Men for værdierne over 127 er de forskellige tegnsæt mere eller mindre forskellige! Dvs. nr 4444 er ikke altid det samme tegn – det afhænger af tegnsættet og vores encoding!

UTF-8 og UTF-16 (Unicode):

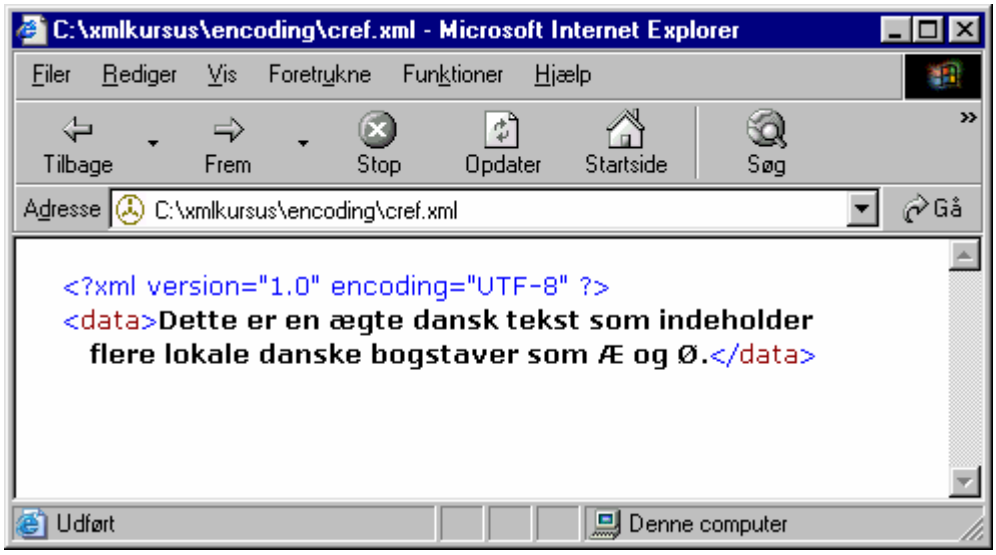
Hvis der IKKE skrives en encoding erklæring bliver XML dokumentet af parseren opfattet som et UTF-8 dokument - som er en speciel kodning hvor hvert tegn oversættes til mellem 1 og 6 bytes afhængigt af om det er et kinesisk skrifttegn eller et engelsk 'a'! UTF-8 koder altså med et **variabelt** antal bytes for hvert tegn! Alle parsere oversætter tekstens tegn til UTF8 – forud for selve bearbejdningen – derfor er det vigtigt at parseren ved hvilken encoding den skal oversætte fra!

UTF-16 som ofte kaldes **'Unicode'** koder derimod med et fast antal bytes og bits: Hvert eneste tegn oversættes til nøjagtigt 2 bytes eller 16 bits – altid! UTF-16 filer vil derfor ofte fylde mere end UTF-8 eller ISO-8859-1 filer! Hvis teksten kun (eller mest) indeholder engelske tegn – bliver filen **dobbelt** så stor! Dette er baggrunden for at man opfandt kodningen UTF-8!

iso-8859-1 er en af mange ISO tegnsæt som indeholder europæiske/central europæiske tegn som danske bogstaver og franske eller spanske tegn! Denne kodning – også kaldet 'Latin-1' – forstås af i praksis alle XML parsere.

Hvis man **vil** anvende den helt generelle UTF kodning – og det kan være en **fordel** hvis dokumentet skal kunne læses også i Japan eller i Tibet! – kan man indsætte **referencer** på denne måde:

```
<?xml version='1.0' encoding='UTF-8'?>
<data>
Dette er en &#230;gte dansk tekst som indeholder flere lokale danske bogstaver som &#198; og
&#216;.
</data>
```



Dette kan virke noget besværligt – men man kan anvende en tekstbehandling med søg og erstat til at indsætte sådanne referencer på en nem måde!

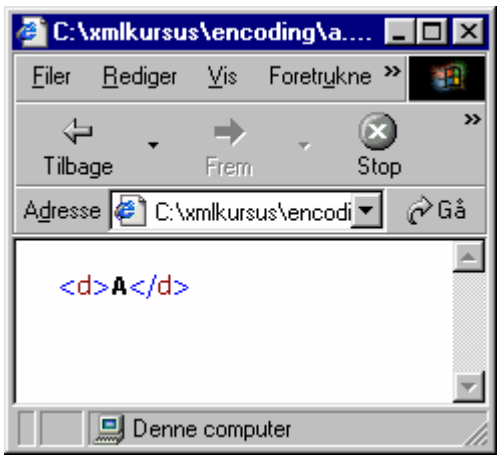
Man kan ikke indsætte entiteter som ø (altså tegnet 'ø') med mindre man selv har defineret en entitet i XML dokumentet med denne definition! Man kan bruge ø i HTML – fordi HTML og XHTML automatisk definerer sådanne entiteter! Men det gør XML ikke!

Man kan have problemer med UTF formaterne fordi nogle metoder f. eks. i MSXML automatisk koder i UTF-16 – selv om vi har bedt om noget andet! I så fald får vi en fejl melding i browseren – at der er skiftet kodning! Hvis et XML dokument først er blevet gemt med en bestemt kodning kan man ikke hen ad vejen ændre denne kodning!

Et eksempel: Vi skriver denne yderst simple tekst og gemmer den som a.xml:

<d>A</d>

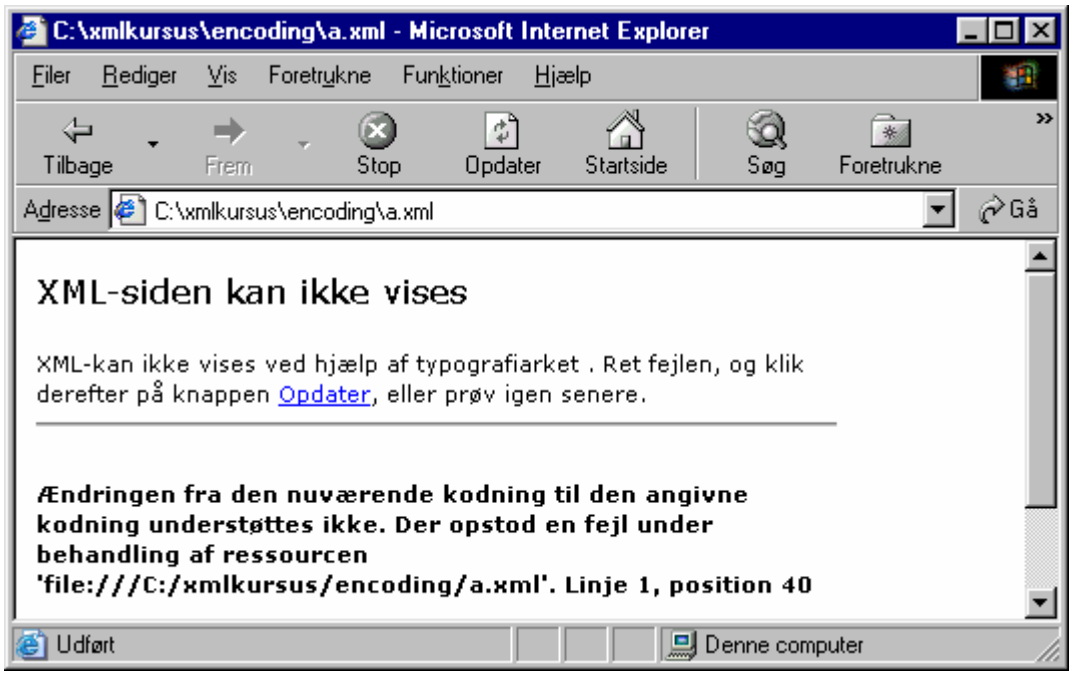
Dette XML dokument vises OK i enhver browser!



Vi tilføjer nu en encoding således:

<?xml version='1.0' encoding='utf-16'?>
<d>A</d>

Vi får nu en fejl:



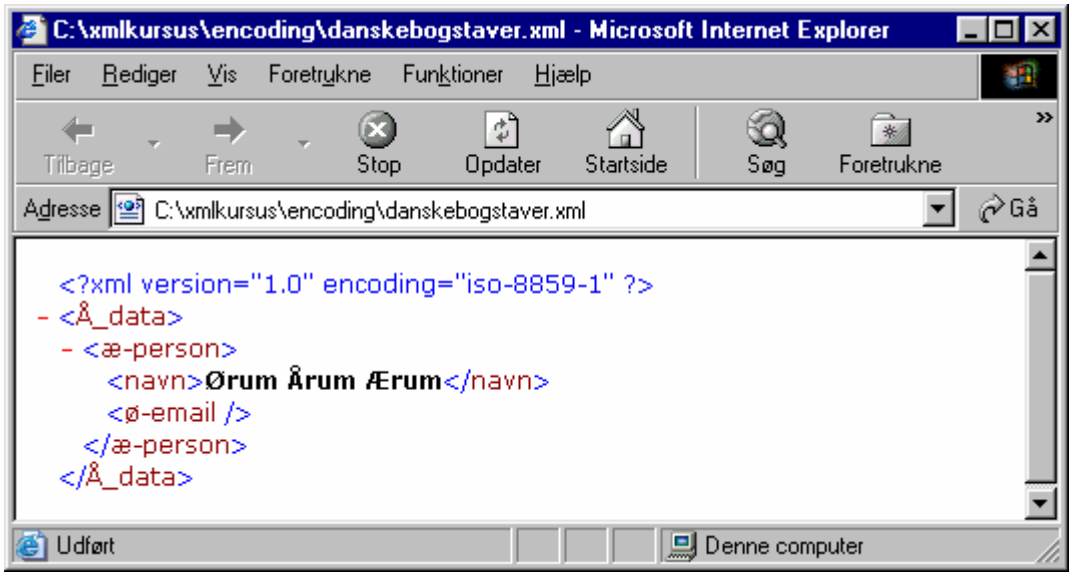
Den angivne kodning er UTF-16 og den nuværende kodning er i følge Internet Explorer UTF-8!

Hvis vi ændrer vores kodning til:

<?xml version='1.0' encoding='utf-8'?>

vises dokumentet uden problemer! XML dokumenter bliver altså gemt automatisk i UTF-8 eller opfattet sådan - hvis intet andet angives!

Det væsentligste er altså at sætte en passende encoding – så opstår ingen problemer:



Som det kan ses er 'æ' osv. OK både i tekstnoden og i definitionen af et element!

Hvis den tekst behandling man arbejder med ikke kan klare 90.000 forskellige tegn kan man altså altid – sikkert – indsætte fremmede tegn med tegn referencer:

궜

XML kan takle alle disse tegn - selv om computerens tekstbehandling eller browser ikke kan!

Man kan vælge forskellige **skrifttyper** i en tekst behandling. Dette er **irrelevant** og har intet at gøre med XML dokumenters **encoding**.

Skift af encoding giver engang imellem problemer. Et eksempel ville være denne fil:

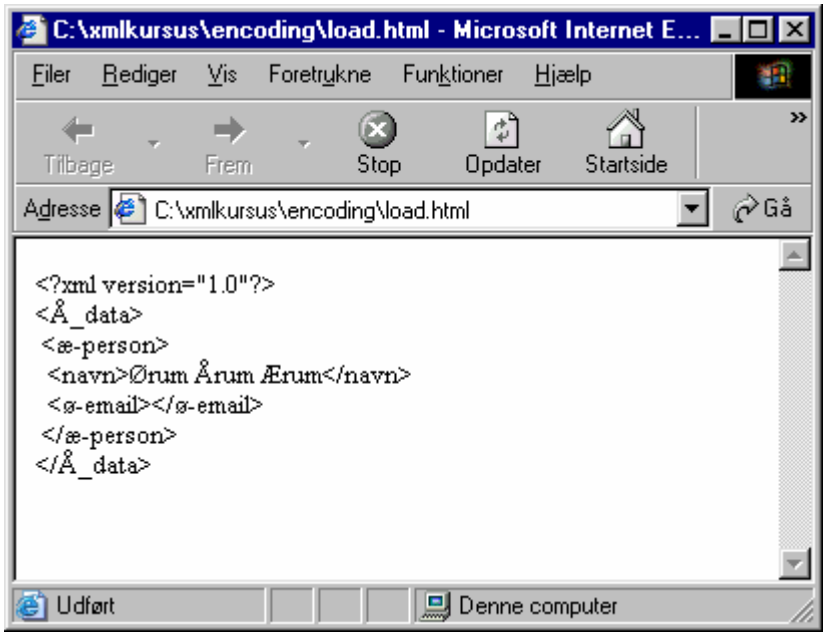
```
<?xml version="1.0" encoding="iso-8859-1"?>

<Å_data>
<æ-person>
<navn>Ørum Årum Ærum</navn>
<ø-email></ø-email>
</æ-person>

</Å_data>
```

Dette dokument vises OK f. eks. i en browser. Men hvis vi **transformerer** eller bearbejder det i script kode – med et Microsoft XSLT stylesheet - bliver resultatet **altid** kodet til UTF-16:

```
<body>
<div id="div"</div>
</body>
<script>
load();
function load(){
var doc=new ActiveXObject("MSXML2.DOMDocument.4.0");
doc.load("danskebogstaver.xml");
```

```
div.innerText=doc.xml;
}
</script>
```

Når vi loader vores dokument og henter værdien doc.xml bliver resultatet et **nyt** dokument kodet i UTF-16 og vores encoding bliver **droppet** af parseren!

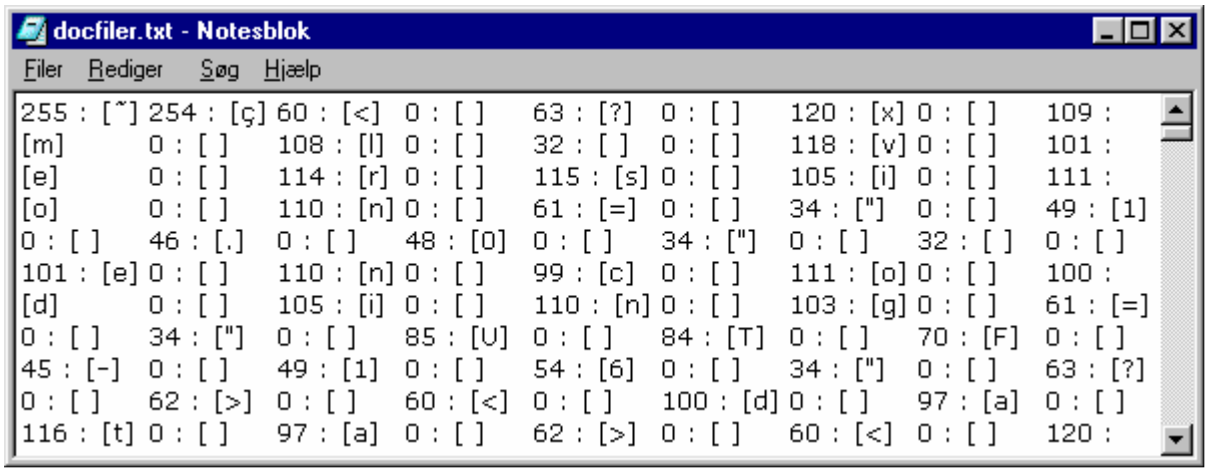


Hvis vi gemmer doc.xml i et nyt dokument dansk.xml og vil åbne det får vi at vide at dokumentet indeholder ugyldige tegn!:

Vi vil senere vende tilbage til spørgsmål og problemer med encoding.

Unicode eller UTF-16 dokumenter er specielle ved at de altid eller normalt har en **'BOM'** – **byte order mark** – som de første bytes i filen **før** <?xml...?>! Dette BOM indsættes **automatisk** – det skrives aldrig direkte! - og derfor kan det af og til give nogle problemer!

Hvis vi ser på en Unicode tekst starter den således – her er den læst binært d.v.s. vi får de enkelte bytes og deres værdi:

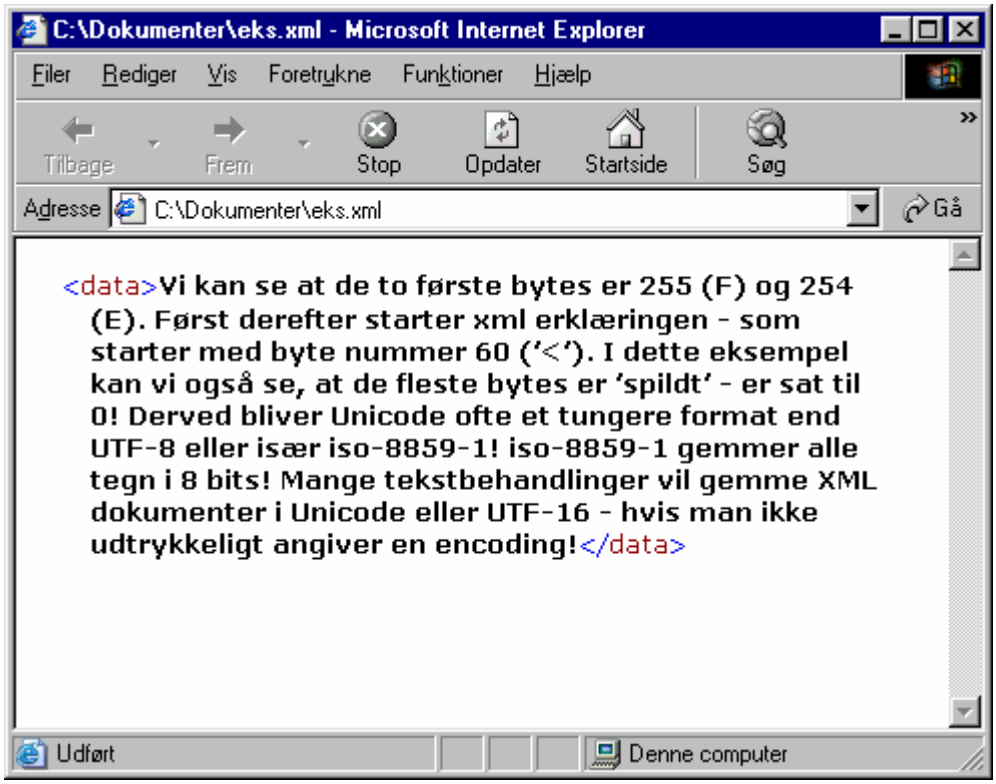


Vi kan se at de to første bytes er **255** (F) og **254** (E). Først **derefter** starter xml erklæringen – som starter med byte nummer 60 ('<').

I dette eksempel kan vi også se, at de fleste bytes er 'spildt' – er sat til 0! Derved bliver Unicode ofte et **tungere** format end UTF-8 eller især iso-8859-1! iso-8859-1 gemmer alle tegn i 8 bits!

Mange **tekstbehandlinger** vil gemme XML dokumenter i Unicode eller UTF-16 – hvis man ikke udtrykkeligt angiver en encoding!

I **Wordpad** i Windows kan man gemme XML dokumenter i **Unicode** formatet – dvs. at de kan skrives uændret – eller som normalt - med f. eks. danske tegn og gemmes som Unicode dvs. UTF 16 direkte!:



Her har vi dog escapet tegnet '<' som ellers ville være et ugyldigt tegn!