



Denne guide er oprindeligt
udgivet på Eksperten.dk

Animation af PNG-24 - Del 1

I denne serie vises, hvordan en animation med et PNG24 billede laves. I Del 1 vises den komprimerede JS-kode og brugen af den i et HTML dokument. I Del 2, hvordan du scripter Photoshop til at lave billedet. I Del 3 gennemgås og forklares koden.

Skrevet den **24. mar 2011** af **olebole** i kategorien **Programmering / JavaScript** | ★★★★★

Denne del af guiden er ikke færdig

Den hadede animerede gif

Vi, som begyndte at kode dengang, JavaScript rollover'en var state of the art, husker endnu den infantile glæde, der bredte sig ved synet af en animeret gif på en kedelig Times New Roman side. Det varede dog ikke længe, før hveranden amatørside havde 20-30 dårlige animationer, som alle var hugget fra andre sider med en anden baggrundsfarve - hvorfor animationerne fremstod med en grim, hakket kant i en underlig farve.

Nu er den her igen

Efter en længere og ganske tiltrængt pause i skammekrogen, kom disse små animationer igen på banen, da Ajax så dagens lys i slutningen af 90'erne. Selv MS har indført den velkendte, roterende ring som pause-cursor i Windows. "It's just so Web 2.0!" *o)

Den nye teknik

Et af problemerne med animerede giffer er dog, at de ikke understøtter semitransparens - og derfor blander dårligt med vekslende baggrunde. Et godt alternativ ville derfor være png24, hvis formatet altså understøttede animation. Det gør det som bekendt ikke, men jeg opdagede for nylig en løsning, der bliver anvendt i MS' Sidebar Gadgets. Når en af MS' Sidebar Gadgets henter data med Ajax, vises nemlig en animeret png24-version af den velkendte pause-cursor.

Hemmeligheden består i at bruge et png24 billede, hvor hver enkelt frame i animationen ligger under hinanden. Er animationen f.eks. 16x16 pixels - og består den af 18 frames - er billedet således 16px bredt og 288px højt. Et grafisk eksempel siger som regel mere end 1000 ord, så prøv at kikke i den Weather Gadget, der følger med Windows 7. Den ligger typisk her:

```
C:\Program Files\Windows Sidebar\Gadgets\Weather.Gadget
```

Dyk ned i mappen "images" og find billedet "activity16v.png". Det er præcist opbygget, som jeg forklarede ovenfor. Animationens første frame er det øverste billede, mens den sidste frame udgøres af det nederste billede.

Png-billedet bruges som baggrundsbillede i et lille div på 16x16px, hvorved kun det øverste billede er synligt. Hvis vi med JavaScript flytter baggrundsbilledet 16px op, vil anden frame vises og så fremdeles. Det faktum at baggrundsbilleder tiler - dvs gentages i det uendelige både horisontalt og vertikalt - kan udnyttes til at animere/loope billedet i det uendelige. For når vi har flyttet billedet så langt op, at det ikke er længere (288px), begynder det jo bare forfra igen.

Prøv det selv

Hvis du vil prøve denne måde at animere på, så kopier koden nedenfor over i et tomt dokument og gemdet som **pnganimator.js**:

```

/*
=====
                PNGAnimator
                (C) Copyright: Ole Clausen, 2011
                Free for use if you leave the lines above untouched
=====
Constructor:
    PNGAnimator(
        path:String
        width:Number
        height:Number
        framesPerSecond:Number [Optional - default:33]
    )
Methods:
    show(
        parentElement:String or HTMLElement
        startFrame:Number [Optional - default:0]
        endFrame:Number [Optional - default:Infinity]
    )

    hide()
=====
Usage:
// Instantiate the animation object
var oSpin = new PNGAnimator("path/to/myanim.png", 16, 16);

// Show in the element 'myDiv'. Start the animation
// -- on the first frame and let it loop infinitely.
oSpin.show("myDiv");

// Stop and hide the animation
oSpin.hide();
*/
(function(){var A=function(sP,nW,nH,fPs){var
c={position:"absolute",width:px(nW),height:px(nH),top:"50%",left:"50%",margin:"- "+px(nH/2)+" -
"+px(nW/2),zIndex:10000,backgroundImage:"url("+sP+")",backgroundPositionY:0};this.h=nH;this.fR=ty
peof fPs=="number"?1000/fPs:33;this.v=false;this.s=document.createElement("div");for(var x in
c)this.s.style[x]=c[x]};var p=A.prototype;p.show=function(P,sF,eF){this._kT();this.y=typeof
sF=="number"?-sF*this.h:0;this.s.style.backgroundPosition="0 "+px(this.y);this.eP=typeof
eF=="number"?-eF*this.h:-Infinity;this.ct=typeof
P=="string"?document.getElementById(P):P;this.ct.appendChild(this.s);this.v=true;this._a();p.hide=fun
ction(){this._kT();if(this.v)this.ct.removeChild(this.s);this.v=false};p._kT=function(){if(this.t){clearInterv
al(this.t);this.t=null}};p._a=function(){var
c=this.s.style,me=this,nP;this.t=setInterval(function(){me.y=me.y-me.h;if(me.y<me.eP)return
me._kT();c.backgroundPosition="0 "+px(me.y)},me.fR)};function px(s){return
s+"px"};window.PNGAnimator=A})();

```

Gem **pnganimator.js** i en mappe sammen med en kopi af billedet, jeg nævnte ovenfor.

Ved siden af disse to filer gemmer du en HTML fil med følgende indhold:

```

<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<title>Untitled Document</title>
<style type="text/css">
#myDiv {
    position: relative;
    width: 400px;
    height: 300px;
    background: #c00;
}
</style>
<script type="text/javascript" src="pnganimator.js"></script>
<script type="text/javascript">
var oSpin = new PNGAnimator("activity16v.png", 16, 16);
</script>
</head>
<body>

<div id="myDiv"></div>

<p><button onclick="oSpin.show('myDiv')">Vis spinner</button>
  <button onclick="oSpin.hide()">Gem spinner</button></p>

</body>
</html>

```

Load HTML dokumentet i en browser og prøv de to knapper.

En lille smule om brugen

Læg mærke til, at constructoren tager tre obligatoriske og et valgfrit argument:

- *) **Path** - En streng, der angiver stien til billedet
- *) **Width** - Et tal, der angiver den enkelte frames bredde
- *) **Height** - Et tal, der angiver den enkelte frames højde
- *) **FramesPerSecond** - Et tal, der angiver, hvormange frames pr. sekund, der animeres med

Læg også mærke til, at metoden **show** tager et obligatorisk og to valgfri argumenter:

- *) **ParentElement** - Det element, animationen skal oprettes i. Kan være en streng (ID) eller en reference til DOMElementet.
- *) **StartFrame** - Et tal, der angiver hvilken frame, animationen skal begynde med.
- *) **EndFrame** - Et tal, der angiver den frame, hvor animationen skal stoppe. Er dette argument sat, looper animationen *ikke*.

Kommentar af T0M (nedlagt brugerprofil) d. 24. mar 2011 | 1

Hvis nogen sku' savne bemeldte PNG-fil ligger der en her:

www.afterhourscomputing.net/software/microsoft/Win7/MS%20Windows%207%20Stuff/Windows%20Sidebar/Gadgets/Currency.Gadget/images/activity16v.png

Det er snedig kode du dér har forfattet - yderst brugbar!

Jeg blev nødt til at "de-komprimere" (for at lære, ikk')
- men savnede en større "spinner" - i det lodrette format : (

Google's image-seacrh kunne ikke finde nogen til mig, men har tilgængæld masser af "sprites", men deres "frames" ligger i 2 dimensioner (ikke bare lodret)..

SÅ, jeg forsøgte at modificere dit lib, så det kunne håndtere dette.. SUCCESS!

Demo her: www.ak247.dk/spriteAnimator.html

Jeg har som sagt "fiflet" gevaldigt med koden!
- man kan nu sætte både hastighed, opacity og z-index med show() metoden.

Håber det er OK ?

Kommentar af T0M (nedlagt brugerprofil) d. 24. mar 2011 | 2

Urimeligt flotte "sprites" her: mrbubblewand.wordpress.com/tag/animation

Kommentar af andreas13_fam d. 29. mar 2011 | 3

Godt at se at olebole er vendt tilbage. Fin guide så længe det bliver forklaret senere. Jeg glæder mig da til at se hvorfor z-index: 10000 er nødvendigt.

Man bør desuden også nævne APNG formatet selvom understøttelsen ikke er god kan den faktisk det efterspurgte.

<http://en.wikipedia.org/wiki/APNG>

Kommentar af jakobdo d. 30. mar 2011 | 4

Fin artikel. :o)