Datavetenskap

Mathilda Gustafsson

Elin Rydström

ELMA 1.0 Testgenerator

Flash-produkt för mätning av stress

Examensarbete, C-nivå

2002:14

ELMA 1.0 Testgenerator

Flash-produkt för mätning av stress

Mathilda Gustafsson

Elin Rydström

Denna rapport är skriven som en del av det arbete som krävs för att erhålla en kandidatexamen i datavetenskap. Allt material i denna rapport, vilket inte är vårt eget, har blivit tydligt identifierat och inget material är inkluderat som tidigare använts för erhållande av annan examen.

Mathilda Gustafsson

Elin Rydström

Godkänd, 2002-06-13

Handledare och examinator: Martin Blom

Sammanfattning

Denna rapport beskriver konstruktionen och utformningen av en testgenerator gjort på uppdrag av Psykologiinstitutionen vid Karlstad universitet. Målet med arbetet har varit att göra ett oberoende test som skall mäta huruvida olika personer påverkas negativt av störningar då de genomgår ett test. Testet skall mäta hur pass stressade personerna blir och om de presterar ett sämre resultat när de utför ett test framför en dator som kan krångla, till skillnad från om de skulle sitta framför ett test på papper.

För att göra denna testgenerator har vi använt oss av Flash 5, som är ett multimediaverktyg med vilket främst filmer och animeringar av olika slag görs. Tanken bakom detta arbete är att skapa en produkt med vilken vår kund lätt ska kunna bygga upp egenutformade test i form av Flash-filmer. Dessa test skall generera ett resultat som skickas från den aktuella Flash-filmen till en databas. Flash-filmen läggs ut på Internet för att testpersonerna skall kunna komma åt testet på ett smidigt och bekvämt sätt.

ELMA 1.0 Test Generator

Flash product for measuring the level of experienced stress

Abstract

In this report we have described the modelling and the configuration of a testgenerator. This project was assigned to us by the department of Psychology at Karlstad University. The purpose of this project was to make an independent test that serves the purpose of measuring the level of stress that a person that endures the test experiences. The test should further evaluate whether or not the testperson that experiences problems while doing the test achieves as good results on the test as the ones that do the test without experiencing problems. In other words the test is supposed to tell our client whether or not a person's results on the test are affected in a negative way when under stress.

To build this testgenerator we have been using the Macromedia program Flash 5, which is a multimedia tool that is mostly used to make movies.
The goal was to create a product that you can easily design different tests with. The tests are also supposed to give some kind of feedback to the tester, so all the answers from the tests are sent from the Flash-movie to a database where the results are stored and further evaluated by the tester. The Flash-movie is published on a webpage so that the testpersons can access the test easily.

Förord

Vi vill tacka alla personer som hjälpt oss med vårt examensarbete, men ett speciellt stort tack till Anders Jonsson på Malmö universitet samt Martin Sandström, Anders Wikström och Pontus Östlund på forumet Communen. Eftersom vi inte hade någon att diskutera våra tankar och problem med angående Flash, var deras kunskaper helt avgörande för utgången av vårt projekt. Vi lade in programmeringsfrågor på forumet och de gav oss många bra tips och lösningar på nästan alla våra problem.

Innehållsförteckning

1 Introduktion 1
1.1 Bakgrund 1
1.2 Syfte och mål 2
1.3 Avgränsning 3
1.4 Problem 3
1.5 Rapportens struktur 4
2 Förutsättningar 4
3 Översikt av Flash 5
3.1 Introduktion 6
3.2 Arbetsflöde 6
3.3 Verktygslåda 7
3.4 Snabbmenyer 7
3.5 Arbetsmiljö 8
3.5.1 Scenen och tidslinjen
3.5.2 Symboler och förekomster
3.5.3 Symboler och interaktiva filmer
3.5.4 Fönstret Bibliotek
3.5.5 Filmutforskaren
3.5.6 Paneler
3.6 Bilder 13
3.7 Vektor- och bitmappsgrafik 14
3.8 Animeringar 15
3.9 Filmer och egenskaper 15
4 Interaktiva filmer 16
4.1 Översikt 16
4.2 ActionScript 17
4.3 Koppla åtgärder till objekt 18
4.4 Mushändelser 19
4.5 Bildrutor och åtgärder 20

4.6 Åtgärder för navigering och interaktivitet 20
4.7 Hoppa till en bildruta eller scen 21
4.8 Spela upp och stoppa filmer 21
4.9 Hoppa till en annan URL 21
4.10 Läsa in en bild 21

5 Konstruktionslösning 22

6 Implementering 23
6.1 Databaskoppling 23
6.2 Mätning av tid 27
6.3 E-postprogram 28
6.4 Popup-ruta 29
6.5 Preloader 30
6.6 Formatering av text 30
6.7 Svarsalternativ 31
6.8 Slumpning 31
6.9 Muspekare 32
6.10 Fönster 32
6.11 Design 33
6.12 Manual 33

7 Erfarenheter och rekommendationer 34

8 Slutsatser 35

Referenser 37

A Tidtabell 39

Figurförteckning

Figur 1.1: Produktidé....2
Figur 3.1: Verktygslåda....7
Figur 3.2: Snabbmeny....7
Figur 3.3: Scen....8
Figur 3.4: Tidslinje....9
Figur 3.5: Bibliotek - Scener....9
Figur 3.6: Symbolredigering....10
Figur 3.7: Bibliotek - Symboler....11
Figur 3.8: Filmutforskare....12
Figur 3.9: Panel – färg....13
Figur 3.10: Vektorgrafik....14
Figur 3.11: Bitmappsgrafik....14
Figur 4.1: Objektåtgärder....17
Figur 4.2: Bildruteåtgärd 1....18
Figur 4.3: Bildruteåtgärd 2....19
Figur 4.4: Tidslinje - frame....20
Figur 6.1: Beskrivande bild av databasintegrering med Flash....24
Figur 6.2: Funktion för ASP-integrering....26
Figur 6.3: Kontrollpanelen – publicering av databas 1....26
Figur 6.4: Kontrollpanelen – publicering av databas 2....27
Figur 6.5: Mailprogram....29

1 Introduktion

Vi har utfört vårt examensarbete åt psykologiämnet vid Karlstads universitet. Vår uppgift var att utforma en slags "verktygslåda". Erik Wästlund, vår kontaktperson på psykologiinstitutionen, skulle med hjälp av denna tillverka egna tester för att kontrollera hur olika personer påverkas av stress i samband med datoranvändande.

1.1 Bakgrund

Användandet av datorer ökar kraftigt. Varje vecka utvecklas nya program och därigenom ökar även datorernas användningsområde. I nuläget använder de allra flesta kontorsanställda en dator i sitt dagliga arbete. Detta medför att de med eller mot sin egen vilja kommer i kontakt med en hel del olika program. Många företag är uppkopplade mot Internet och på dessa kontor har personalen oftast någon form av e-postprogram installerade på sina datorer.

Datorer i största allmänhet kan öka stressen avsevärt hos så väl en van som ovan användare. Finns det dessutom ett e-postprogram på datorn som automatiskt känner av när användaren har fått ny post och meddelar honom/henne detta, kan det både uppfattas som ett störande och stressande moment om användaren är upptagen med något annat då meddelandet om ny post kommer upp på skärmen.

Vi har ofta funderat på hur mycket stress en dator som krånglar kan framkalla hos en människa och när det fanns ett examensarbete om just detta, blev vi genast mycket intresserade. Vi bestämde tillsammans med vår kund att vi med hjälp av programmeringsspråket Java skulle göra en produkt för att utröna hur människan påverkas av stress vid arbete med datorer.
Med hjälp av denna produkt skall vår kund kunna utföra tester som visar hur personer som utsätts för stressande moment påverkas när de arbetar vid sin dator i jämförelse med personer som inte utsätts för några störningar. Testpersonerna som stressas ska läsa samma text och svara på samma frågor som personerna som inte utsätts för störningar och resultaten skall sedan jämföras.

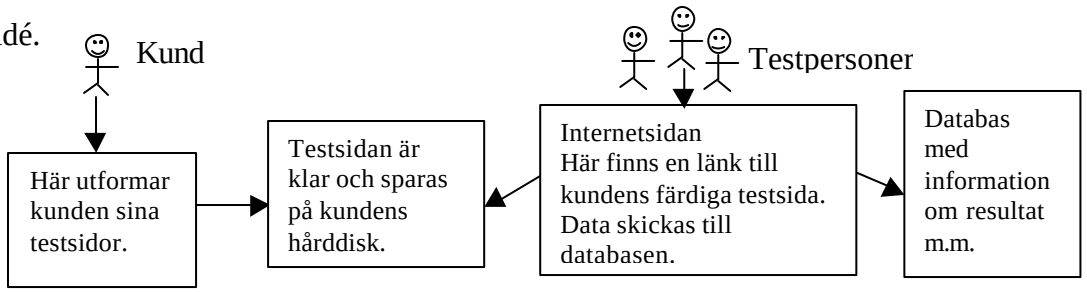
Efter att ha läst några böcker om Java och undersökt möjligheterna att göra en sådan produkt i detta språk och även hittat information om andra språk, valde vi att inte använda Java då andra programmeringsspråk verkade vara bättre lämpade för denna uppgift. Vi bestämde då i samråd med vår handledare, att vi skulle använda oss av multimediaverktyget Flash istället. Detta eftersom vi hade hittat ett liknande test på en webbsida och denna var gjord i Flash.

1.2 Syfte och mål

Syftet med detta examensarbete är att skapa en produkt som skall kunna utföra olika stresstålighetstester på en viss utvald grupp människor.

Vid examensarbetets början var målet att skapa en användarvänlig produkt där vår kund kan lägga till vissa fördefinierade funktioner, som tillsammans skall bilda ett test som liknar högskoleproven. Vi har dock under arbetets gång kommit fram till att det inte fanns någon anledning att göra en sådan "verktygslåda" i Flash. Flash är en verktygslåda i sig själv och vi tyckte att det var onödigt att göra en verktygslåda i denna redan utmärkt fungerande verktygslåda. Istället valde vi att lära ut hur vår kund kan bygga egna test i Flash.

Målet med vårt examensarbete blev att tillverka en produkt som vi skall använda som mall när vi lär ut hur egna sidor i Flash görs. Denna Flash-produkt skall förutom fin design av många sidor och formulär även innehålla en del störningsmoment i form av ett e-postprogram, muspekarförändringar, timglas, tidtagningslist och speciell utformning av testtexterna. I uppgiften ingick även att göra ett simulerat e-postprogram och en databas. Databasen används till att lagra och hämta information om testen. När testpersonerna utför testet skall bland annat information om angivet svarsalternativ skickas till denna databas. Informationen sparas som en Excel-fil för att vår kund sedan skall kunna läsa in denna direkt i ett statistikprogram och därifrån på ett enkelt sätt kunna sammanställa och tolka informationen. Figur 1.1 visar vår produktidé.



Figur 1.1: Produktidé

1.3 Avgränsning

Allt eftersom tiden gick och vi lärde oss mer om Flash, insåg vi dess begränsningar. Vi förstod att vissa saker som vi först trott var möjliga att genomföra var helt omöjliga i Flash.

Vår produkt är uppbyggt av dessa delar:

- Databas.
- Simulerat e-postprogram.
- Tidsfunktion.
- Inloggning med lösenord.
- Rullister till långa texter.
- Olika frågeutseenden och textutseenden.
- Popup-ruta för meddelanden.
- Stressladdare (visar under hur lång tid testet har pågått).
- Preloader, det vill säga filmladdare.
- Timglas.
- "Säkerhet" – fönster utan minimera/maximera/stängknapparna.

Vår produkt kommer tyvärr inte att bestå av följande delar:

- Riktig slumpning.
- Verktygslåda för produktion av nya test.

Anledningen till att vi valde bort slumpningen var att olika Flash-filmer som körs på olika datorer inte känner av varandra. Det vill säga, om en variabel blir "satt" till ett värde i en film som körs på en dator blir denna variabel inte "satt" i en annan film som körs på en annan dator. Det blir därför svårt att se hur många som kommit in till de olika testen. En av anledningarna till att vi valde att inte göra en verktygslåda var att Flash, vilket vi tidigare nämnt, är en verktygslåda i sig själv.

1.4 Problem

Detta examensarbete har varit svårt att strukturera på ett lämpligt sätt, eftersom vi inte har haft någon handledning inom Flash-programmering. Implementeringsavsnittet, senare i denna uppsats, tar upp de problem vi haft.

1.5 Rapportens struktur

Rapporten är skriven för personer som har ungefär samma bakgrundskunskaper som vi, det vill säga personer som har en högskoleutbildning i datavetenskap eller motsvarande.

Grundstrukturen på denna rapport följer den ordning som vi arbetat i under detta examensarbete, det vill säga med början vid bakgrund och studerandet av Flash, till slutprodukten. För att inte glömma bort vad vi har gjort och därmed förlora viktig information, har vi kontinuerligt dokumenterat allt vi gjort.

Resten av rapporten har följande indelning. Kapitel 2 beskriver förutsättningar och krav. Kapitel 3 presenterar nödvändiga grunder i Flash som behövs för att förstå resten av uppsatsen. I Kapitel 4 har vi gått in på djupet och förklarat mer om hur man gör interaktiva filmer i Flash. I Kapitel 5 beskriver vi vår konstruktionslösning i korthet. I Kapitel 6 utvecklar vi det som vi beskrivit i föregående kapitel och detaljerar varje dels beskrivning. Kapitel 7 tar upp våra tankar om vad vi lärt oss under projektets gång. I Kapitel 8 presenteras slutligen våra slutsatser och en beskrivning av tidsupplägget tas upp.

2 Förutsättningar

Detta examensarbete består av en 10-poängs uppsats som skrivs på C-nivå. Uppsatsen skrivs på halvfart och beräknad tid är därför 20 timmar per vecka. Vår handledare på datavetenskap heter, som vi tidigare nämnt, Martin Blom och vår handledare på psykologiinstitutionen heter, som vi också tidigare nämnt, Erik Wästlund. Under tiden för examensarbetet har regelbundna möten med båda våra handledare genomförts.

När vi bestämde oss för detta examensarbete trodde vi att vi skulle kunna använda oss av programmeringsspråket Java. En av anledningarna till att vi tyckte Java lät bra, var att vi hade lite förkunskaper om hur man programmerar i detta språk. Tyvärr märkte vi efter två veckors inläsning av Java, att vi inom examensarbetets tidsram inte skulle kunna skapa den produkt som vår uppdragsgivare beställt. Vi fick då hjälp av vår handledare att komma på en lösning om hur vi skulle kunna göra den efterfrågade produkten inom tidsramen. Han berättade att det finns ett multimediaverktyg som heter Flash och att det verkade vara anpassat för att göra just sådana här tester. Vi hade aldrig hört talas om Flash tidigare, men vi tyckte att det som Martin

berättade verkade bra. Vi började genast läsa in oss på Flash. Tyvärr fanns det bara böcker om de mest grundläggande sakerna i Flash. När vi läst dessa och fortfarande inte hittat det som vi egentligen letade efter, nämligen information om hur man programmerar i Flash, började vi leta efter detta på Internet. Där fann vi en hel del information och snälla personer som ville dela med sig av sina kunskaper.

Eftersom utvecklingen av vår produkt sker i Macromedia Flash 5.0 i Windows-miljö, har tillgången till en bra arbetsplats varit begränsad. På Karlstads universitet finns detta program enbart installerat på datorerna i Multimediaprogrammets datorsalar och dit kunde vi inte få tillgång. Därför har vi till en början suttit hemma vid våra egna datorer, men i mitten av april fick vi äntligen en arbetsplats på skolan.

Tillgång till litteratur har förutom universitetets bibliotek och Karlstads stadsbibliotek erbjudits av Informatiks eget lilla bibliotek samt av Martin Blom.

3 Översikt av Flash

Eftersom konkurrensen om besökare på webbsidor idag är mycket hård blir det allt viktigare att använda rörliga effekter och detaljer som fångar besökarens uppmärksamhet och som gör denne aktiv. Flash är ett program som har utvecklats för att skapa snygga och effektfulla animationer på ett lätt och smidigt sätt.

Dessa animeringar kan köras fristående eller bäddas in i en webbsida. Det är det senare alternativet som har gjort Flash så populärt. Undersökningar visar att 80-90 procent av alla som besöker Internet har stöd för Flash-animeringar, men med nya uppdateringar och versioner av webbläsare borde de öka [2].

Gränssnittet i utvecklingsverktyget liknar Photoshops med många paneler med olika inställningar.

För att göra en animering i Flash, skapar man ett antal bilder (olika sidor) vilka tillsammans bildar en så kallad scen. Varje sida i scenen utgör en del av animeringen. Man bygger upp varje sida med bilder och grafik som man antingen gör själv eller importerar. Slutligen låter man Flash ”limma ihop” alla sidor till en fungerande animering [2].

I detta kapitel kommer vi, för att öka läsarens förståelse för vårt arbete, att presentera grunderna i Flash.

3.1 Introduktion

Flash-filmer är grafik och animeringar för webbplatser. De består huvudsakligen av vektorgrafik, men de kan även innehålla importerade bitmappar och ljud. Vektor- och bitmappsgrafik skrivs det mer om i kapitel 3.7. Flash-filmer kan innehålla funktioner för interaktivitet så att användarna kan skicka indata och skapa ickelinjära filmer som kan fungera med andra webbprogram. Många webbdesigners använder Flash för att skapa navigeringskontroller, animerade logotyper, längre animeringar med synkroniserat ljud och till och med fullständiga webbplatser med multimediefunktioner. Flash-filmerna är komprimerad vektorgrafik, vilket gör att inläsningen går snabbt [8].

Antagligen har Du både sett och använt Flash-filmer på olika webbplatser, till exempel Disney [51], The Simpsons [52] och Coca-Cola [53]. Miljontals Internetanvändare har fått Flash Player levererad med datorn, webbläsaren eller systemprogramvaran och många andra har hämtat den från Macromedias webbplats. Flash Player läggs på den lokala hårddisken och används för uppspelning av filmer i webbläsare och fristående program. Att se en Flash-film i Flash Player är ungefär samma sak som att titta på en videofilm på en videobandspelare fast med större interaktivitet. Flash Player är den enhet som används för att visa de filmer som man skapar i redigeringsprogrammet i Flash [8].

3.2 Arbetsflöde

När man arbetar i Flash skapar man filmer genom att skapa eller importera grafik, ordna grafiken på scenen och animera den med tidslinjen. Man kan göra filmen interaktiv genom att använda åtgärder som gör att filmen svarar på händelser på ett bestämt sätt [8].

När man är klar med animeringen kan man spara den på många olika sätt. Man kan till exempel spara den som en vanlig Flash-fil. Då sparar programmet filen i två format, ett arbets- och redigerarformat, .fla och ett färdigt filmformat, .swf. Men man kan även göra om den till något annat filformat, till exempel html. Om man sparar filen som .html kan man titta på den i en webbläsare, medan man måste ha en speciell Flash Player om den är sparad som .swf [1].

3.3 Verktygslåda

Med verktygen i verktygslådan kan man teckna, måla, välja och ändra bilder, samt ändra vyn på scenen. Verktygslådan är indelad i fyra sektioner, vilket kan ses till höger i figur 3.1:

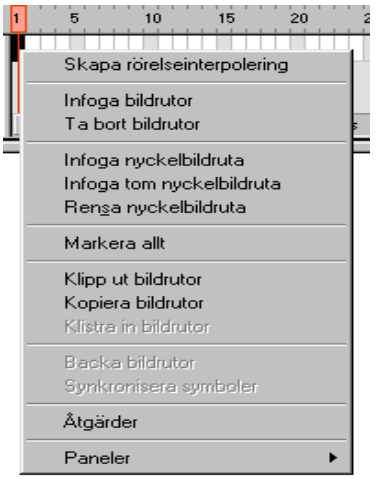
- Sektionen *Verktyg* innehåller verktyg för att teckna, måla och välja.
- Sektionen *Visa* innehåller verktyg för att zooma och panorera i programfönstret.
- Sektionen *Färger* innehåller verktygsalternativ för konturlinjefärger eller fyllningsfärger.
- Sektionen *Alternativ* innehåller verktygsalternativ för de valda verktygen som påverkar verktygen vid målning och redigering [8].

Figur 3.1: Verktygslåda

3.4 Snabbmenyer

Snabbmenyer innehåller kommandon som är relevanta för det som är markerat i fönstret. Om man till exempel markerar en bild i tidslinjen, innehåller snabbmenyn kommandon för att skapa, ta bort och ändra bilder och nyckelbilder. En nyckelbild är den specifika bild som är kopplad till en bildruta. För att öppna en snabbmeny högerklickar man (i Windows) på ett objekt i tidslinjen, i biblioteksfönstret eller på scenen.

Figur 3.2 till höger visar en snabbmeny för en markerad bildruta [8].

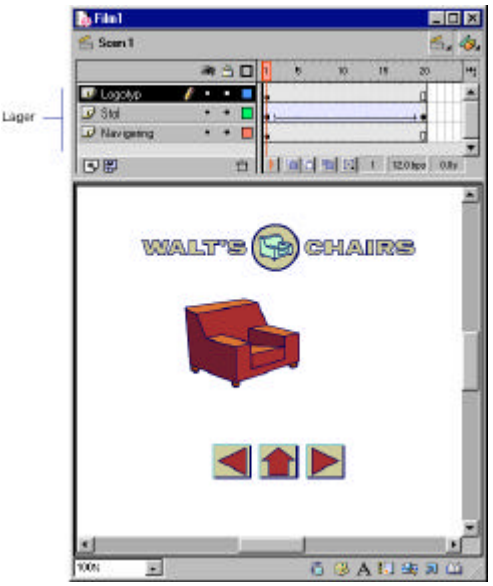


Figur 3.1: Snabbmeny

3.5 Arbetsmiljö

När man skapar och redigerar filmer arbetar man vanligtvis i de här huvudområdena:

- Scenen är det område där filmen spelas upp.
  Figur 3.3 visar en scen.
- Tidslinjen, där grafik animeras i tidsföljd.
- Symboler, återanvändningsbara mediaobjekt för en film.
- Fönstret Bibliotek, där symboler ordnas.
- Filmutforskaren, som ger en översikt över en film och dess struktur.
- Flytande, dockningsbara paneler som gör att man kan ändra olika element i filmen och konfigurera redigeringsmiljön i Flash så att den passar sitt arbetsflöde.



Figur 3.1: Scen

3.5.1 Scenen och tidslinjen

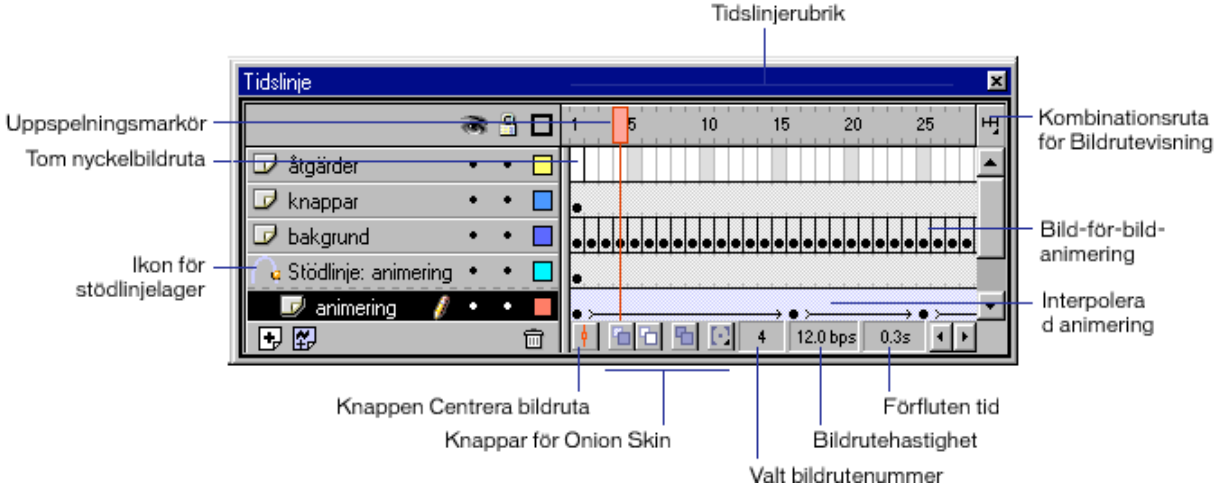
Liksom i andra filmer delas Flash-filmer upp i bildrutor. På scenen sätter man ihop innehållet i enskilda bildrutor i filmen genom att rita bilder direkt på scenen eller använda importerad grafik [8].

Scenen är där man sätter ihop enskilda bildrutor till en film. I tidslinjen samordnar man animeringarna och sätter samman grafiken i olika lager. På tidslinjen visas varje bildruta i filmen [8].

På tidslinjen koordinerar man animeringarna och sätter ihop dem i olika lager.
Lager fungerar som en stapel med genomskinliga OH-ark, som håller isär olika grafiska element så att man kan kombinera olika element till en sammanhängande bild [8].

Använda tidslinjen

Med tidslinjen ordnas och styrs innehållet i en film över tiden i lager och bildrutor. De viktigaste delarna i tidslinjen är lager, bildrutor och uppspelningsmarkören.



Figur 3.1: Tidslinje

De lager som finns i en film visas i en kolumn till vänster om tidslinjen, se figur 3.4. De bildrutor som finns i varje lager visas på en rad till höger om lagernamnet. I tidslinjerubriken överst i tidslinjen anges bildrutenumren. Uppspelningsmarkören anger den aktuella bildrutan som visas på scenen.

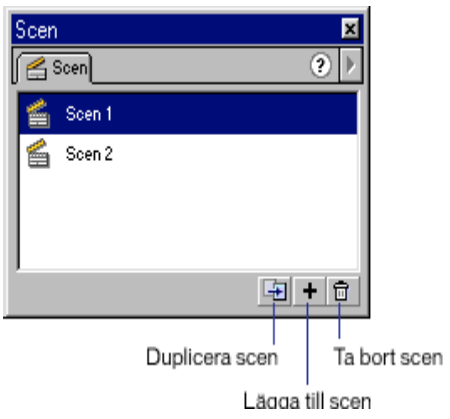
Tidslinjens status visas längst ned i tidslinjen och innehåller det aktuella bildrutenumret, den aktuella bildrutehastigheten och hur lång tid som gått sedan filmen startades.

Man kan infoga, ta bort, välja och flytta bildrutor i tidslinjen. Man kan även dra bildrutor till en ny plats på samma lager eller till ett annat lager [8].

Använda scener

Om man vill ordna en film tematiskt kan man använda scener. Man kan till exempel använda separata scener för inledning, inläsning av ett meddelande och sluttexter.

När man publicerar en Flash-film som innehåller fler än en scen spelas scenerna i .swf-filen upp i den ordning som de visas i panelen Scen i .fla-filen. Figur 3.5 visar hur panelen Scen ser ut.



Figur 3.2: Bibliotek - Scener

Man kan lägga till, ta bort, duplicera, byta namn och ändra ordningen på scener.
Om man vill avbryta eller pausa en film efter varje scen, eller låta användaren navigera i filmen på ett ickelinjärt sätt, kan man använda åtgärder [8]. Åtgärder är de händelser man kopplar till ett objekt och dessa åtgärder bestämmer vad som ska hända då man till exempel klickar på ett objekt.

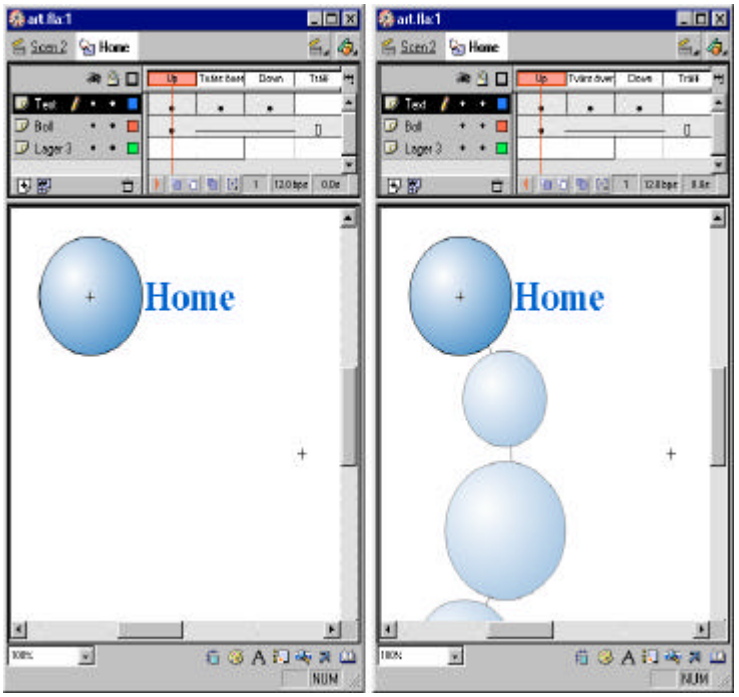
3.5.2 Symboler och förekomster

Symboler är återanvändningsbara element som man kan använda i en film. Symboler kan vara grafik, knappar, filmklipp, ljudfiler eller teckensnitt. När man skapar en symbol sparas den i biblioteket. När man placerar en symbol på scenen skapar man en förekomst av symbolen.

Symboler reducerar filstorleken eftersom en symbol endast sparas en gång i filen oavsett hur många förekomster det finns av den. Det kan vara bra att använda symboler, animerade eller statiska, för alla element som förekommer fler än en gång i en film. Man kan ändra egenskaperna för en förekomst utan att huvudsymbolen ändras och man kan redigera själva huvudsymbolen så att alla förekomster ändras.

Man kan redigera symboler direkt på scenen. Andra element på scenen visas nedtonade. Man kan också redigera en symbol i ett separat fönster. När man redigerar en symbol visas endast tidslinjen för den symbol man redigerar.

Figur 3.6 visar ett "redigeringsförlopp" av en symbol. Vänster bild visar hur det ser ut när man redigerar en fristående symbol och höger bild när man redigera en symbol i sitt sammanhang i filmen [8].



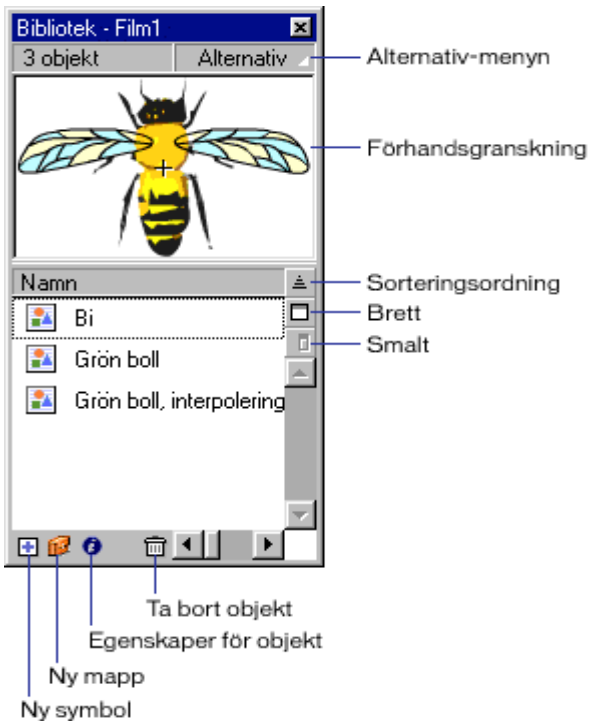
Figur 3.1: Symbolredigering

3.5.3 Symboler och interaktiva filmer

Symboler ingår även när man skapar interaktiva filmer. Man kan använda förekomster av symboler för att skapa interaktivitet i en film. Man kan till exempel skapa en knappsymbol som ändras beroende på vad användaren gör med musen och placera en förekomst av en symbol på scenen. Man använder en annan typ av symbol, som kallas filmklipp, för att skapa avancerade interaktiva filmer [8].

3.5.4 Fönstret Bibliotek

I biblioteksfönstret, se figur 3.7, lagras och ordnas symboler som har skapats i Flash, samt importerade filer som exempelvis ljudfiler, bitmappsgrafik och QuickTime-filmer. I biblioteksfönstret kan man organisera biblioteksobjekt i mappar, se hur många gånger ett objekt används i en film och sortera objekt efter typ [8].



Använda biblioteket

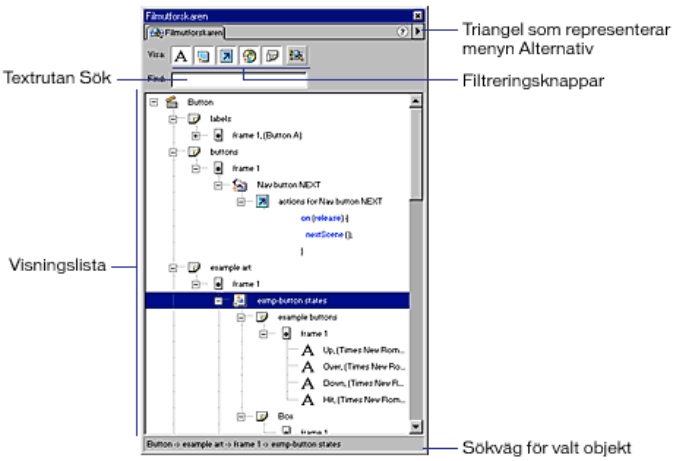
Ett bibliotek i en Flash-film innehåller symboler, inklusive de som skapats i Flash och de som importerats till Flash, vilket gör att man kan visa och ordna dessa filer medan man arbetar. Biblioteksfönstret innehåller en bläddringslista med namnen på alla objekt i biblioteket. En ikon bredvid ett objektnamn i biblioteket anger objektets filtyp.

Figur 3.1: Bibliotek - Symboler

När man markerar ett objekt i biblioteksfönstret visas en miniatyrbild för förhandsgranskning av objektet överst i biblioteksfönstret. Om det markerade objektet är animerat eller är en ljudfil, kan man förhandsgranska det med Flash Playern.

Man kan skapa permanenta bibliotek i Flash-programmet som är tillgängliga när man startar Flash. Flash innehåller även flera inbyggda bibliotek som innehåller knappar, grafik, filmklipp och ljud som man kan lägga till i de egna Flash-filmerna [8].

3.5.5 Filmutforskaren



Filmutforskaren, se figur 3.8, innehåller funktioner för att visa och ordna innehållet i en film och markera element i filmen som skall ändras. Den innehåller flera funktioner för att effektivisera arbetsflödet när man skapar filmer [8].

Figur 3.1: Filmutforskare

Använda Filmutforskaren

Filmutforskaren kan till exempel användas för att göra följande:

- Söka efter ett element i en film med hjälp av namnet.
- Visa egenskapspanelen för ett markerat element som skall ändras.
- Lära Flash-användaren mer om strukturen i en film som skapats av en annan person.
- Hitta alla förekomster av en viss symbol eller åtgärd.
- Ersätta alla förekomster av ett visst teckensnitt med ett annat.
- Skriva ut den navigerbara visningslistan som visas i Filmutforskaren.

Filmutforskaren innehåller en visningslista, en lista med innehållet i filmen ordnat i ett navigerbart hierarkiskt träd. Man kan filtrera vilka kategorier av objekt i filmen som skall visas i Filmutforskaren. Man kan visa de valda kategorierna som filmelement (scener), symboldefinitioner eller både och [8].

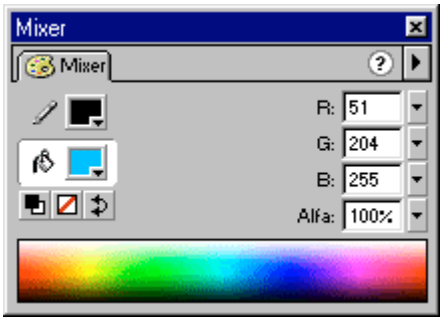
3.5.6 Paneler

Om man vill visa, ordna och ändra element i en Flash-film kan man använda flytande paneler som innehåller kommandon och alternativ som tillhör respektive typ av element. Med paneler kan man ändra symboler, förekomster, färger, text, bildrutor och andra element.

Man kan använda paneler för att anpassa Flash-gränssnittet genom att visa de paneler som man behöver för en viss uppgift och dölja andra paneler [8].

Använda paneler

Med hjälp av flytande paneler kan man visa, ordna och ändra element i en film. De alternativ som finns i panelerna styr egenskaperna för de valda elementen.
Figur 3.9 visar *Mixer* panelen. Denna används när man arbetar med färger.



Figur 3.1: Panel – färg

Med panelerna i Flash kan man arbeta med objekt, färger, text, förekomster, bildrutor, scener och hela filmer. Man kan till exempel använda panelen *Tecken* för att välja teckensnittsattribut och panelen Bildruta för att ange bildruteetiketter och välja interpoleringsalternativ. Om man vill visa en fullständig lista över tillgängliga paneler i Flash väljer man Fönster > Paneler.

3.6 Bilder

Flash innehåller olika metoder för att skapa originalbilder och importera bilder från andra program. Man kan skapa objekt med rit- och målningsverktygen och ändra attributen för befintliga objekt. Man kan även importera vektor- och bitmappsgrafik från andra program och ändra den importerade grafiken i Flash [8].

3.7 Vektor- och bitmappsgrafik

Grafik kan visas på datorn i antingen vektor- eller bitmappsformat. Det underlättar arbetet om man känner till skillnaden mellan de två formaten. I Flash kan man skapa och animera komprimerad vektorgrafik. Man kan också importera och arbeta med vektor- och bitmappsgrafik som har skapats i andra program [8].

Vektorgrafik

Vektorgrafik beskriver bilder med linjer och kurvor, så kallade vektorer, som även innehåller egenskaper för färg och placering. I en bild av ett löv, se figur 3.10, beskrivs till exempel lövets konturer av punkter som binds samman av linjer. Lövets färg bestäms av konturens färg och färgen på det område som innesluts av konturen.

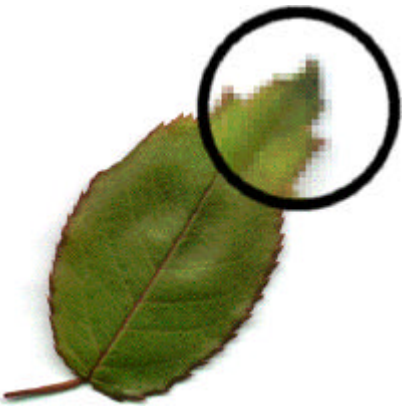
När man redigerar vektorgrafik ändrar man egenskaperna för de linjer och kurvor som utgör bilden. Man kan flytta och ändra storlek, form och färg på vektorgrafiken utan att ändra bildkvaliteten. Vektorgrafiken är inte beroende av upplösning, vilket gör att den kan visas på skärmar med olika upplösning utan att bildkvaliteten ändras [8].

Figur 3.1: Vektorgrafik

Bitmappsgrafik

Bitmappsgrafik beskriver bilder med hjälp av färgade punkter, så kallade. bildpunkter (kallas även "pixlar") som ordnats i ett rutnät. Bilden av lövet till höger, se figur 3.11, kan till exempel beskrivas av den specifika placeringen och färgen på varje bildpunkt i rutnätet, vilket skapar en bild på ett mosaikliknande sätt.

När man redigerar bitmappsgrafik ändrar man bildpunkter istället för linjer och kurvor. Bitmappsgrafiken är beroende av upplösning eftersom de data som beskriver bilden ligger inom ett rutnät med fast storlek.



Figur 3.2: Bitmappsgrafik

Det gör att bildkvaliteten kan ändras om man redigerar bitmappsgrafiken. Det är speciellt storleksändringar som kan få kvaliteten att försämras. Till exempel kan bildelementens kanter bli taggiga och ojämna eftersom bildpunkterna omdistribueras i rutnätet. Om bitmappsgrafik visas på en skärm med lägre upplösning än själva bilden, försämras också bildkvaliteten [8].

3.8 Animeringar

Med hjälp av Flash kan man animera objekt så att de verkar röra sig över scenen och/eller ändra form, storlek, färg, kapacitet, rotation och övriga egenskaper. Man kan skapa bildruta-för-bildruta-animeringar. Då skapar man en separat bild för varje bildruta. Men man kan också skapa interpolerade animeringar, där man gör den första och den sista bildrutan för en animering och instruerar Flash att skapa alla bildrutor däremellan [8].

3.9 Filmer och egenskaper

Varje gång man öppnar Flash skapas en ny fil. Man kan skapa flera nya filmer under tiden man arbetar. Om man vill ange storlek, bildrutehastighet, bakgrundsfärg och övriga egenskaper för en ny film använder man dialogrutan ”Egenskaper för film” [8].

I Flash kan man skapa interaktiva filmer där användarna kan använda tangentbordet eller musen för att hoppa till olika delar i filmen, flytta objekt, ange information i formulär och utföra många andra operationer. Man kan skapa interaktiva filmer genom att skapa åtgärder med hjälp av ActionScript [8]. ActionScript beskrivs mer ingående i kapitel 4.2.

Interaktiva filmer beskriver vi utförligare i nästa kapitel.

4 Interaktiva filmer

I detta kapitel kommer vi att berätta om hur man skapar interaktiva filmer i Flash. Det är just detta som vårt examensarbete är uppbyggt på och vi vill att läsaren lättare skall förstå våra två kapitel ”Konstruktionslösning” (kapitel 5) och ”Implementering” (kapitel 6).

4.1 Översikt

I en enkel animering spelas scener och bildrutor i en film upp i ordningsföljd. I en interaktiv film använder användaren tangentbordet, musen eller bådadera för att hoppa till olika delar av filmen, flytta objekt, ange information i formulär och utföra många andra interaktiva åtgärder [8].

Man skapar interaktiva filmer genom att ange åtgärder, eller grupper av instruktioner i ett ActionScript som körs när en viss händelse inträffar. Händelser som utlöser en åtgärd kan antingen vara när uppspelningsmarkören når en viss bildruta, när användaren klickar på en viss knapp eller trycker på vissa tangenter på tangentbordet [8].

Att användaren kan navigera i en film genom att hoppa till en scen, bildruta eller URL, eller starta/stoppa en annan film, eller skriva ut bildrutor från Flash-filmen, är bara en del av den interaktivitet som man kan skapa [8].

I panelen Åtgärder anger man åtgärder för en knapp, ett filmklipp eller en bildruta. Genom att använda kontrollerna i panelen Åtgärder i Normalläge kan man infoga åtgärder utan att skriva ActionScript. Det är även möjligt att skriva egna script i ActionScript. Instruktionerna kan vara i form av en enskild åtgärd, till exempel att instruera en film att avbryta uppspelningen, eller en serie åtgärder som att först utvärdera ett villkor och sedan utföra en åtgärd. Många åtgärder kräver ganska liten erfarenhet av programmering medan andra åtgärder kräver viss kunskap om programmeringsspråk och är avsedda för avancerad utveckling [8].

4.2 ActionScript

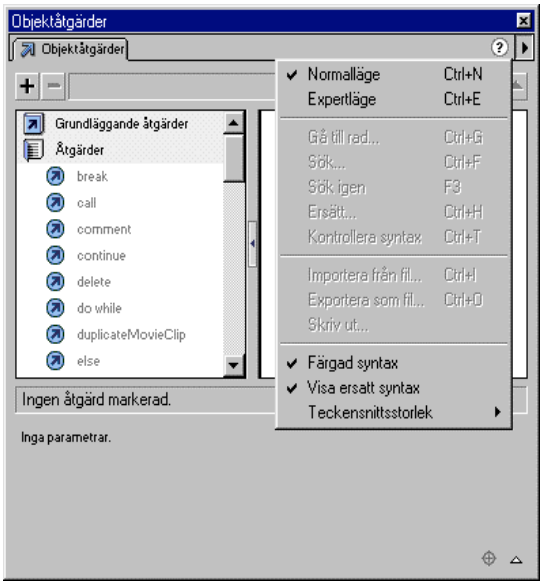
Med hjälp av ActionScript, scriptspråket i Flash, kan man lägga till interaktivitet i en film. På samma sätt som JavaScript är ActionScript ett objektorienterat scriptspråk. I objektorienterade script ordnar man information genom att dela upp den i grupper som kallas klasser. Man kan skapa flera förekomster av en klass och använda i ett script. Förekomsterna kallas objekt. Man kan antingen använda de fördefinierade klasserna som finns i ActionScript eller så kan man skapa egna [8].

När man skapar en klass definierar man alla egenskaper (kännetecken) och metoder (beteenden) för varje objekt som skapas, precis som man definierar verkliga ting. Till exempel kan en person ha egenskaper som kön, längd och hårfärg och beteenden som att prata, gå och kasta. I det här exemplet är "person" en klass och varje enskild person är ett objekt eller en förekomst av den klassen [8].

Objekt i ActionScript kan innehålla information och kan visas grafiskt som filmklipp på scenen.

Använda panelen Åtgärder

Från panelen Åtgärder kan man skapa och redigera åtgärder för objekt eller bildrutor genom att använda två olika redigeringslägen. Man kan välja redan skrivna åtgärder från listan i verktygslådan, dra- och släpp-åtgärder samt använda knappar för att ta bort eller omstrukturera åtgärder. I Normalläge kan man skriva åtgärder med hjälp av parameterfält (argument) som frågar efter de rätta argumenten. I Expertläge kan vana ActionScript-användare redigera scripten med en textredigerare på samma sätt som JavaScript [8].

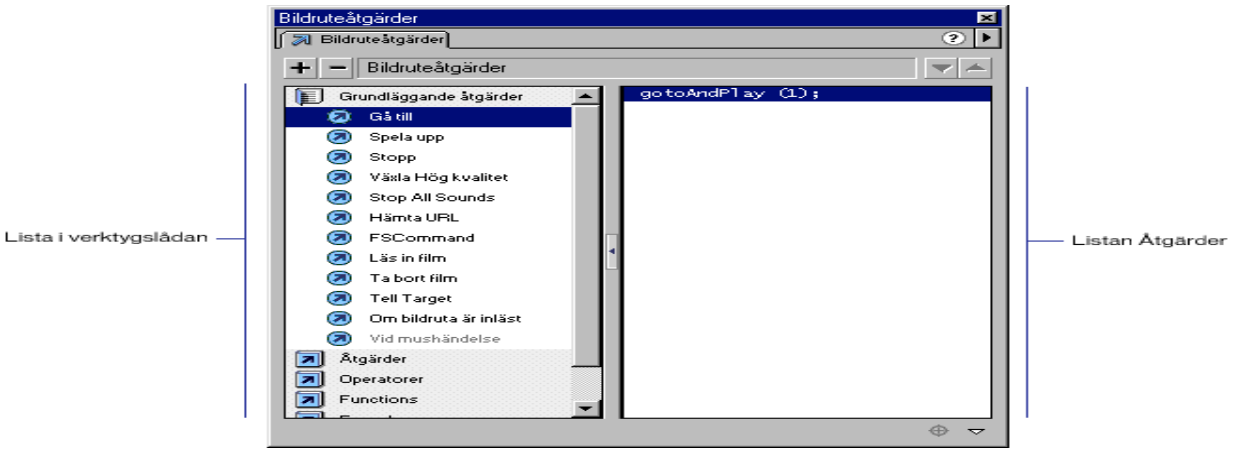


Figur 4.1: Objektåtgärder

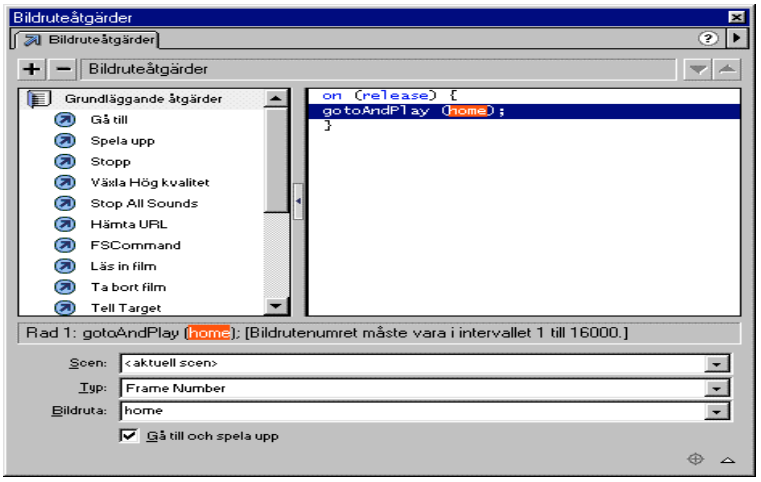
4.3 Koppla åtgärder till objekt

Man kan koppla en åtgärd till en knapp eller ett filmklipp så att åtgärden utförs när användaren klickar på knappen, för pekaren över den, när filmklippet läses in eller när det når fram till en särskild bildruta. Man kopplar åtgärden till en förekomst av knappen eller filmklippet. Andra förekomster av symbolen påverkas inte. När man kopplar en åtgärd till en knapp eller ett filmklipp tilldelas automatiskt en specialåtgärd som kallas för hanterare, åtgärden ”OnMouseEvent” (vid mushändelse) för knappar eller åtgärden ”OnClipEvent” (vid klipphändelse) för filmklipp. En hanterare hanterar en händelse på ett visst sätt och innehåller grupper med ActionScript-programsatser som körs när en särskild händelse inträffar. Varje hanterare börjar med ordet `on` eller `onClipEvent` följt av en händelse som hanteraren reagerar på [8].

Händelser är åtgärder som inträffar medan en film spelas upp, till exempel när ett filmklipp läses in, när uppspelningsmarkören kommer fram till en bildruta eller när användaren trycker på en tangent på tangentbordet. Följande instruktioner beskriver hur man anger åtgärder för objekt genom att använda panelen Åtgärder i Normalläge [8].
Figur 4.2 nedan visar åtgärden gotoAndPlay(1); som betyder att filmen skall gå till bildruta 1 och spela upp den. I figur 4.3 på nästa sida skall samma sak hända, men bara om man släpper upp musen efter det att man tryckt på en knapp.



Figur 4.1: Bildruteåtgärd 1



Figur 4.2: Bildruteåtgärd 2

4.4 Mushändelser

När man kopplar en åtgärd till en knapp kopplas även en mushändelseåtgärd till knappen för att hantera åtgärden. Varje hanterare börjar med ordet `on` följt av en händelse som hanteraren svarar på. Exempel:

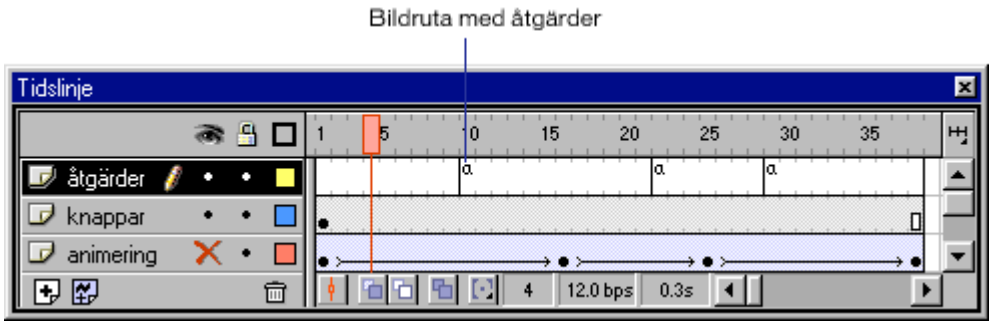
```
on (release)
on (keyPress "<Space>")
on (rollOver)
```

Parametern release anger att användaren har tryckt på och släppt upp musknappen, parametern keyPress ”<Space>” anger att användaren har tryckt ned mellanslagstangenten, medan parametern rollOver anger att användaren har muspekaren över objektet.

Man kan ange vilka mushändelser som skall utlösa en knappåtgärd med hjälp av panelen Åtgärder [8].

4.5 Bildrutor och åtgärder

Om man vill få en film att göra något speciellt när den når en nyckelbildruta, kopplar man en bildruteåtgärd till nyckelbildrutan. Man kan till exempel skapa en slinga i en film genom att lägga till en bildåtgärd i bildruta 20 som anger "gå till bild 10 och spela upp".
Det kan vara bra att placera alla bildruteåtgärderna i ett lager så att det är lättare att spåra dem. Bildrutor med åtgärder har ett litet a i tidslinjen, se figur 4.4 nedan [8].



Figur 4.1: Tidslinje - frame

4.6 Åtgärder för navigering och interaktivitet

Med hjälp av kategorin ”Enkla åtgärder” i panelen Åtgärder kan man kontrollera navigering och användarinteraktivitet i en film genom att välja åtgärder och låta Flash skriva ett ActionScript åt en. De grundläggande åtgärderna innefattar följande:

- Hoppa till en bildruta eller en scen med hjälp av åtgärden `goto`.
- Spela upp och stoppa filmer med åtgärderna `play` och `stop`.
- Hoppa till en annan URL med hjälp av åtgärden `getURL`.
- Läsa in och ta bort ytterligare filmer med åtgärderna `loadMovie` och `unloadMovie`.
- Kontrollera andra filmer och filmklipp med åtgärden `TellTarget`.
- Kontrollera om en bildruta har lästs in med åtgärden `ifFrameLoaded`.
- Koppla en mushändelse eller en tangent som utlöser åtgärden med åtgärden `on(mouseEvent)` [8].

4.7 Hoppa till en bildruta eller scen

Om man vill hoppa till en särskild bildruta eller scen i en film använder man åtgärden `goto`. När filmen hoppar till en bildruta kan man spela upp filmen från den nya bildrutan (standard) eller stanna vid den. Filmen kan även hoppa till en scen och spela upp en angiven bildruta eller den första bildrutan i nästa eller föregående scen [8].

4.8 Spela upp och stoppa filmer

Om inget annat anges spelar en startad film igenom alla bildrutor i tidslinjen. Man kan stoppa eller starta en film med särskilda intervall med hjälp av åtgärderna `play` och `stop`. Man kan till exempel stoppa en film i slutet av en scen innan den fortsätter till nästa scen. När filmen har stoppats måste den startas igen med hjälp av åtgärden `play`.
Åtgärderna `play` och `stop` är de åtgärder som används mest för att kontrollera filmklipp med knappar, eller för att kontrollera huvudtidslinjen [8].

4.9 Hoppa till en annan URL

Om man vill läsa in ett dokument från en viss URL till ett specifikt webbläsarfönster, eller överföra variabler till ett annat program på en definierad URL, använder man åtgärden `getURL`. För att se att denna åtgärd fungerar krävs att den begärda filen finns på den angivna platsen och att absoluta URL:er har en nätverksanslutning till exempel http://www.minserver.com/ [8].

4.10 Läsa in en bild

Om man vill skapa en förinläsare för att hindra vissa åtgärder från att utlösas innan det nödvändiga innehållet har lästs in av användaren kan man använda åtgärden `ifFrameLoaded`. En förinläsare är en enkel animering som spelas upp medan resten av filmen läses in. Åtgärden `ifFrameLoaded` är användbar för att bekräfta att en stor fil (till exempel en bitmapp eller en ljudfil) är inläst. Man kan även använda egenskapen `_framesloaded` (i åtgärden if) för att kontrollera om innehållet i en viss bildruta är tillgängligt lokalt.

Genom att antingen använda åtgärden eller egenskapen kan man starta uppspelningen av en enkel animering medan resten av filmen läses in till en lokal dator. Båda kontrollerar om innehållet i en särskild bildruta är tillgängligt lokalt. Normalt brukar åtgärden `ifFrameLoaded` användas som en bildruteåtgärd, men den kan även användas som en knappåtgärd [8].

5 Konstruktionslösning

Vi inledde vårt arbete genom att tillsammans med vår kund utforma en skiss av produkten. Efter intensiva diskussioner med kunden om olika scenarion, tankar och förväntningar fick vi en bra skiss som vi var noga med att förklara för oss själva och kund flera gånger för att vara säkra på att inga onödiga missförstånd skulle uppstå.

När vi gjort detta fortsatte vi med att undersöka Flash som program, hur dess arbetsmiljö är upplagd och hur den verkade fungera. Det var dock svårt att bara testa sig fram, så vi tog hjälp av en lektion på Internet.
Lektionen var upplagd så att en företagsrepresentant hade filmat in sig själv i ett Flash-programs bakgrund där han hela tiden berättade om olika applikationer och funktioner medan han samtidigt visade var i programmet man hittade alla saker. Han berättade om olika tillvägagångssätt man kan använda för att göra olika händelser i Flash. Problemet var att han bara tog upp de mest grundläggande teknikerna i Flash. Eftersom Flash i huvudsak är ett verktyg för att skapa filmer så fokuserade lektionen på animeringar av olika slag. Databaser anses exempelvis vara för komplicerat och tas därför inte upp. Det som lektionen gav var kunskap om de olika biblioteken och var det mesta fanns.

Till en början verkade det som att vi inte behövde programmera någonting alls själva, eftersom Flash har en hel del inbyggda funktioner. Lektionen tog inte heller upp någon egen programmering, men då vi arbetat med Flash ett tag upptäckte vi att det fanns if-satser, while-loopar, for-loopar och andra vanliga programspråkskonstruktioner. Sättet att skriva kod på är ganska likt C++, vilket var en fördel för oss då vi under vår tid som ”programmerare” huvudsakligen har programmerat i detta programspråk.

Det fanns från vår kunds sida vissa grundläggande krav på hur den färdiga produkten skulle fungera. De grundläggande kraven var att vi skulle göra två olika test, ett som innehöll stressande moment och ett som fungerade felfritt under testet. I båda fallen skall information om utförda test läggas in i en databas och därefter kunna läsas in i ett statistikprogram. När en person kommer till webbsidan där testet ligger, skall han/hon slumpmässigt loggas in på något av de två testalternativen. Från det att testpersonen loggar in för att starta testet skall tidtagning ske. Vi skulle även göra ett simulerat e-postprogram som skulle meddela testpersonen när denne fått brev. Hur vi gick tillväga beskrivs utförligare i nästa kapitel.

6 Implementering

När vi kände oss lite säkrare i Flash-miljön och trodde oss veta hur vår kund ville att produkten skulle fungera gjorde vi en layout till det simulerade e-postprogrammet. Vi gjorde även en temporär lösning till hur det skulle fungera vid läsning av e-posten och så vidare.

6.1 Databaskoppling

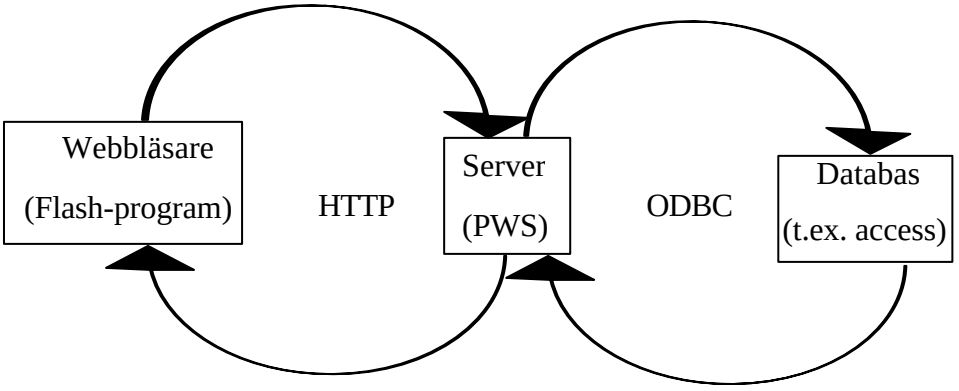
Vi ansåg att det allra viktigaste i detta examensarbete var att först och främst göra en fungerande koppling mellan Flash och en databas. Detta eftersom vi hade minst aning om hur man skulle lösa databasproblemet. Därför avbröt vi vårt arbete med e-postprogrammet och startade arbetet med databasen istället.

Denna koppling, som vi från början trodde skulle vara relativt enkel, visade sig vara betydligt mer komplicerad. Problemet är att man måste ha en server med något script på, mellan Flash-programmet och databasen. Exempel på script är CGI (Common Gateway Interface), PHP (Personal Home Page) eller ASP (Active Server Pages). En Flash-film kan inte kommunicera direkt med en databas. Däremot kan Flash-filmer ”prata” med en serverapplikation (mellanvara). Mellanvaran kan fråga databasen och skicka data fram och tillbaka mellan databasen och Flash-programmet. Det finns många olika serverapplikationer att välja mellan. De vanligaste är CGI, ASP, PHP, ColdFusion och Tango [9].

Vi valde att använda oss av en PWS (Private Web Server) eftersom vi endast kunde sitta hemma och testa på vår hemdator och detta program finns på Windows98-skivan som vi hade tillgång till. Sedan valde vi att använda ASP-script eftersom psykologiinstitutionens server ligger i operativsystemet Windows och ASP är den teknik som lämpar sig bäst för detta.

När man har lagt in en viss funktion som hänvisar till ett ASP-script på en frame i Flash-filmen, skickar Flash-programmet en förfrågan till webbservern. Om det är en ASP-fil och det finns scriptkod innanför avgränsarna <% %>, körs scriptkoden först och därefter sänds sidan till Flash som Html-kod. I de fall där scriptet innehåller ett anrop till en databas, skickar servern en SQL-fråga via ODBC (OpenDataBaseConnectivity) till databasen. Frågespråket som används är alltså SQL (Structured Query Language), vilket är det vanligaste språket för att hantera data i relationsdatabaser. Vi har bara gjort en enkel databas som består av en tabell med kolumner motsvarande varje fråga i testet.

Databasen förser sedan webbservern/PWS med det data som begärdes och detta skickas tillbaka till klienten/Flash-programmet som ren Html-kod efter det att ASP-scriptet har körts [10]. Figur 6.1 nedan visar i enkla drag hur Flash-filmen kommunicerar med databasen.



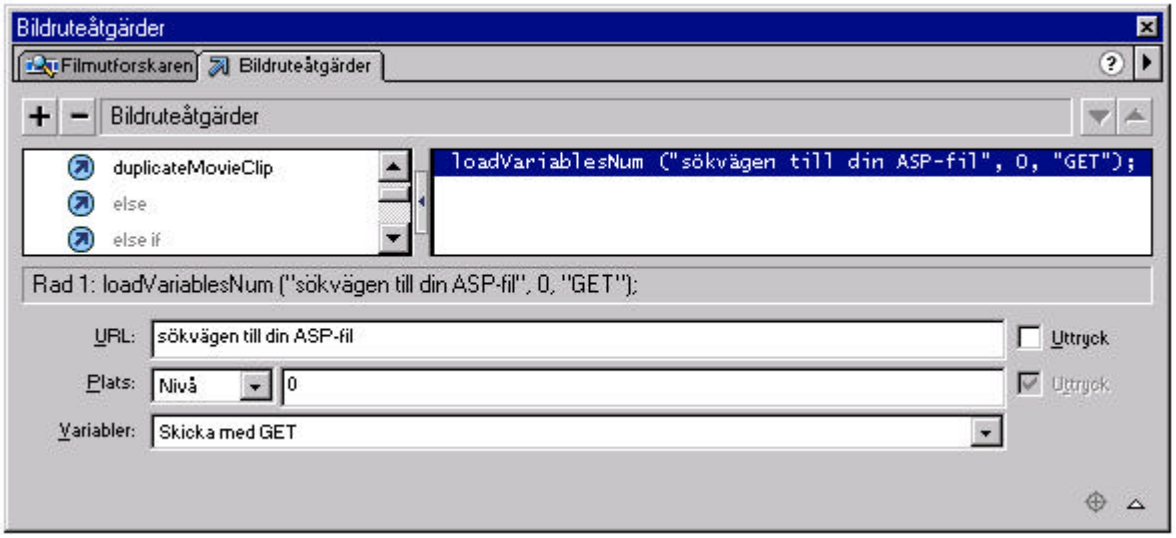
Figur 6.1: Beskrivande bild av databasintegrering med Flash

Från början var det svårt att sätta sig in i själva tanken med hela integreringen och sedan lära sig ASP-script, med tanke på att vi inte ens kunde Flash när vi inledde projektet. Vi lärde oss ASP och fick allting att integrera felfritt mellan Flash och en databas. För att få kunskaper om ASP-script fick vi söka en hel del på Internet och logga in på olika forum. Vi har haft mycket stor nytta av speciellt ett av forumen, nämligen Communen [55], men även Eforum [54].

Vi kan med säkerhet säga att vi inte hade klarat av detta exjobb om det inte vore för all hjälp vi fått från dessa forum.

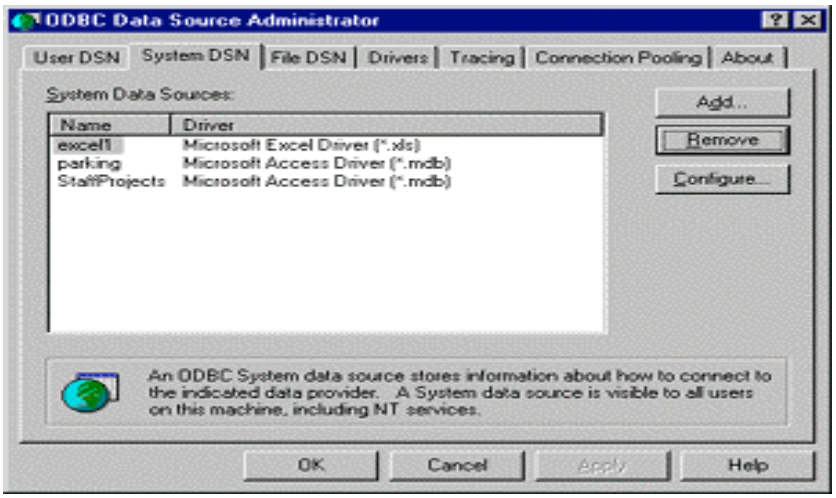
Som alltid när man är oerfaren inom ett område är det fördelaktigt att börja från grunden. För att kunna gå in på de mer komplicerade bitarna i ett programspråk, bör man ha en ganska bra grund att bygga från. Vi började med att testa lite grundläggande ASP-script som vi bara skrev in i en Html-fil. för att se vad scriptet gjorde. Vi gjorde script som skrev ut tiden, script som tog hand om text som skrivs in i ett formulär med mera. Testet med formuläret gjorde vi noga eftersom det var ungefär samma princip som det vi skulle utnyttja i Flash-filmen.

En testperson skall kunna mata in något i ett formulär i Flash och detta skall fångas upp i en variabel som kan skickas iväg till databasen. Efter mycket testning av script av olika karaktär började vi med att undersöka hur Flash integrerar med ASP-filer. Detta gjorde vi genom att titta på några exempel som fanns på Internet och genom att läsa på den inbyggda hjälpen i Flash. Det finns en inbyggd funktion i Flash som heter `loadVariablesNum()` som tar in tre argument och vi använde denna funktion för att kommunicera med ASP-filen. Funktionens syntax ser ut så här: `loadVariablesNum (url, plats, [variabler])`, vilket visas i figur 6.2 på nästa sida. Argumentet `url` är en absolut eller relativ webbadress till variablerna. Värden för webbadressen måste vara i samma underdomän som filmen när den öppnas med en webbläsare. Argumentet `plats` är en nivå eller ett mål för variablerna. I Flash Player tilldelas filmfiler ett nummer som bestäms av i vilken ordning de blev inlästa. Den första filmen laddas till bottennivån (nivå 0). Inuti åtgärden `loadMovie` måste man ange ett nivånummer för varje efterföljande film. Argumentet `plats` är valfritt. `Variabler` är också ett valfritt argument som anger en metod för sändning av variabler. Om det inte finns några variabler utelämnar man argumentet. I annat fall anger man om variabler ska laddas med en GET- eller POST -metod. GET bifogar variablerna i slutet av en webbadress och används när ett litet antal variabler är aktuella. POST skickar variablerna i en separat Http-rubrik och används för långa variabelsträngar. Funktionen läser data från en extern fil, till exempel en textfil eller text som genererats av ett CGI-script, ASP eller PHP och anger värdena för variabler i en film eller ett filmklipp. Åtgärden kan också användas för uppdatering av variabler i den aktiva filmen med nya värden.



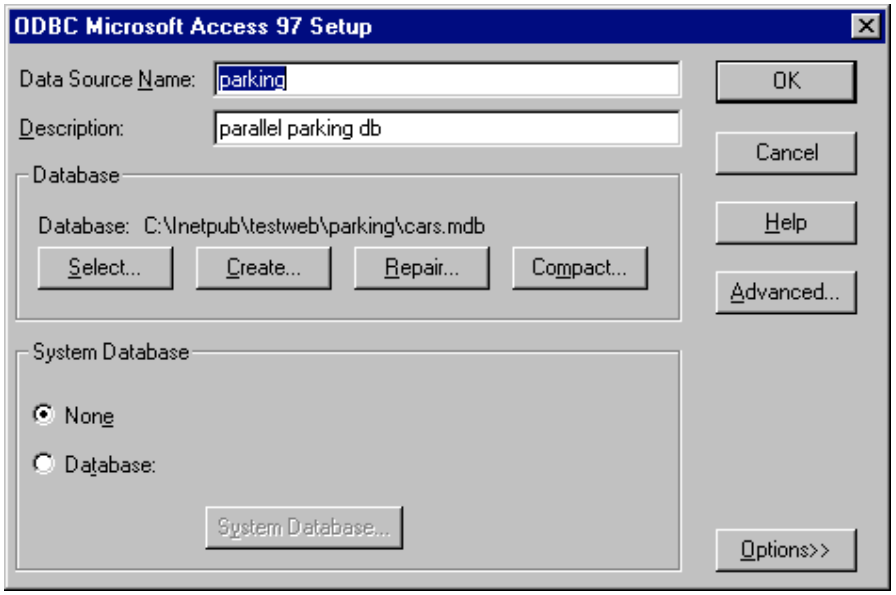
Figur 6.2: Funktion för ASP-integrering

Inte nog med detta, ASP behöver även ha kontakt med en databas. För att lära oss om detta läste vi en ASP/databaskopplingslektion på Internet. Där fanns en detaljerad beskrivning över hur man publicerar den databas som skall användas och hur man får databasen att kunna kommunicera med ASP-scriptet. För att få kopplingen att fungera så gick vi in i kontrollpanelen och dubbelklickade på ODBC-ikonen. Då kom vi till ett fönster där vi klickade på DSN (Data Source Name)-fliken och i den fliken klickade vi på add/lägg till. Figur 6.3 visar fönstrets utseende.



Figur 6.3: Kontrollpanelen – publicering av databas 1

Sedan fick vi välja vilken slags databas vi skulle använda, i vårt fall rörde det sig om en databas implementerad i MS Access. I fältet där DSN skall fyllas i skrev vi in det namn vi valt till vår databas, tryckte Select och skrev sedan in sökvägen till databasfilen på datorn [25]. Figur 6.4 visar miljön där vi valde vår databas.



Figur 6.4: Kontrollpanelen – publicering av databas 2

Efter några småjusteringar fick vi hela integreringen att fungera som den skulle. När detta stora steg var klart började vi att arbeta på resten av de delar som skulle finnas med i slutprodukten. Hur vi till exempel skulle göra det simulerade e-postprogrammet, lösa tidsinställningen och utforma alla delars utseenden. Detta trodde vi också skulle vara relativt enkelt och inte alltför jobbigt, men det visade sig att det blev mycket komplikationer på vägen till den slutliga och fulländade versionen.

6.2 Mätning av tid

Det första som vi angrep var tidsfrågan, kunden vill mäta tiden det tar för en testperson att göra hela testet. Detta behövs givetvis för att kunna jämföra om en testperson som utsätts för störningar tar längre tid på sig att slutföra testet än en testperson som inte får några störningar alls.

Först började vi se efter om det fanns någon räknare inbyggd i Flash som automatiskt kunde hålla koll på hur länge en film används. Det finns en funktion som heter `getTimer()` som returnerar antalet millisekunder som har förflutit sedan filmen började spelas upp, men denna passade inte eftersom vi hoppar mellan olika filmklipp från själva huvudfilmen. Vår slutliga

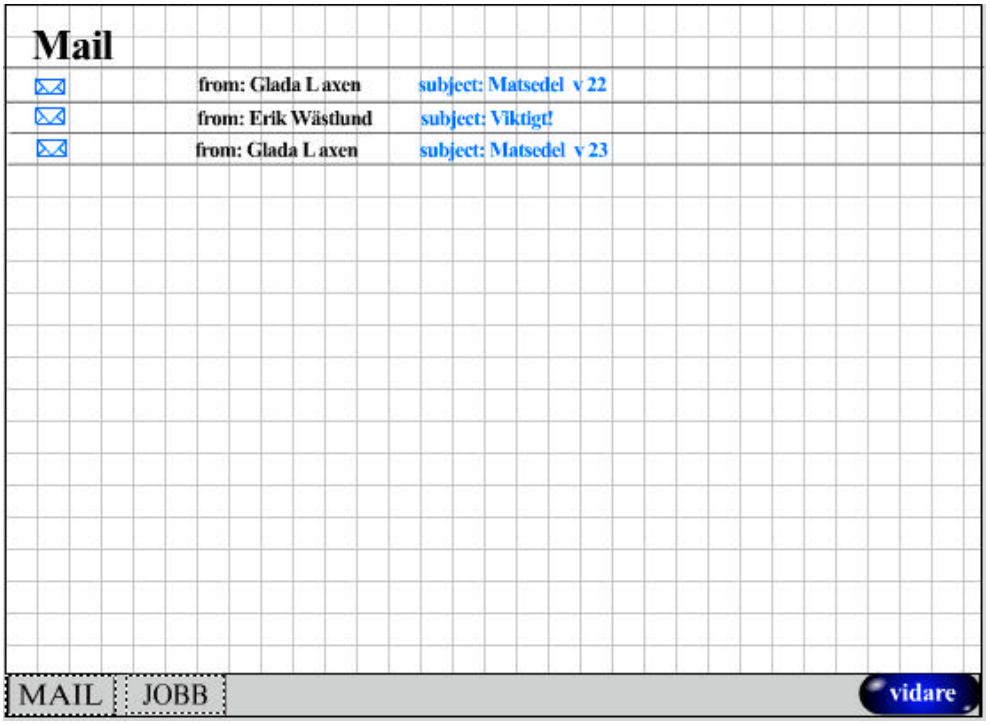
lösning på detta problem blev att vi hämtar tiden då en testperson startar filmen och lägger den i databasen. När sedan testpersonen klickar på avsluta-knappen och är färdig med testet, hämtar vi tiden igen och kan på så sätt räkna ut hur lång tid testpersonen behövde för att göra testet. För att hämta tiden satte vi ett objekt till `newDate()` och hämtade timmarna, minuterna och sekunderna på objektet med `get`-funktioner.

Den andra tidsfrågan vi hade var att kunden vill att en testperson skall kunna se en viss fråga och kunna svara på denna under en begränsad tid och skall sedan komma vidare i testet. Detta var inte särskilt svårt, men det fanns ett krav till, nämligen att en stressande uppladdare skall visa en list som fylls i mer och mer ju mindre tid testpersonen har kvar. Vi löste först problemet med den stressande uppladdaren med en funktion som beroende på tiden gick till olika frames i filmklippet och spelade upp dessa. En sak som inte var bra med detta var att då man kommer tillbaka till filmen där klippet ligger, så körs klippet från början. Det vill säga. laddaren laddar upp till så lite tid som testpersonen har kvar nu och hoppar sedan ner till noll (som om att testpersonen har hela tiden kvar) för att senare hoppa upp till så lite tid som testpersonen har kvar senare. Vi hittade dock till slut ett ”smartclip” (en färdig Flash-film) som gjorde ungefär samma sak som vi ville att den skulle göra så vi tog dess princip och använde det istället.

6.3 E-postprogram

Det vi tog tag i härnäst var det simulerade e-postprogrammet som kunden vill skall störa testpersonerna under testets gång. Detta blev mycket krångligt, då vi ville att det skulle se ut precis som vilket e-postprogram som helst. Testpersonen skall kunna se om han/hon har läst ett brev eller inte. Han/hon skall kunna läsa ett brev vid vilken tidpunkt som helst och även hur många gånger som helst. Det som krånglade till allt var att när man gör en helt vanlig knapp så kan den byta utseende medan man klickar på den, men den går tillbaka till sitt ursprungliga utseende när man kommer in på den frame där knappen ligger igen. Detta kunde vi dock lösa genom att göra ett filmklipp med två frames i vilka det ligger två knappar med olika utseende. Problemet som återstod nu var att det inte gick att sätta åtgärder på knapparna i filmklippet så att man kunde komma vidare till en annan frame i huvudfilmen. Det visade sig dock gå bra. Anledningen till att vi först fick problem med detta var att vi inte hade tillräckliga kunskaper om hierarkin i Flash-filmerna. Det var nu som vi lärde oss lite mer om `_root` vilken är ”huvudtidslinjen” i Flash. [26]. Genom att anropa funktionen `gotoAndPlay(frame)` med _root kunde vi sätta de rätta åtgärderna på knapparna i

filmklippet. Men det var inte bara det som behövdes, för att få välja en frame som ligger i en annan scen, måste man döpa den frame och hänvisa direkt till frame-namnet. Det var hur vi gick till väga för att bygga upp e-postprogrammet som ni kan se i figur 6.5.



Figur 6.1: Mailprogram

6.4 Popup-ruta

För att testpersonen skall veta att han/hon har fått brev önskade kunden att vi skulle göra en popup-ruta som informerar testpersonen om att denne har olästa brev. Denna popup skall vara tidsinställd för en viss frame eller en viss fråga. När testpersonen kommit till en ny frame skall han/hon kanske ha sett denna frame i en minut innan rutan dyker upp. Ett annat scenario kan vara att testpersonen har svarat på en fråga och tre sekunder efter det dyker rutan upp. Detta kunde vi välja att lösa på tre olika sätt. Endera genom att göra ett riktigt fönster med en fördröjning på, att ”fejka” ett fönster med en fördröjning, eller att göra två likadana frames med samma bildrutor i, fast en fördröjning på den första framen och lägga in en klickbar ruta i den andra framen. Vi valde det andra alternativet, att ”fejka” ett fönster med en fördröjning. Detta eftersom vi tyckte att detta alternativ kändes smidigast. För att göra den ”fejkade” rutan läste vi först en lektion om hur man gör en ”fejkad” ruta som man kan stänga och öppna på Internet [27]. Därefter lärde vi oss att det finns en åtgärd i Flash som kallas `_visible` som kan göra ett filmklipp synligt eller osynligt för filmtittaren. Givetvis insåg vi att det var helt

idealiskt att nyttja denna åtgärd. För att simulera att testpersonen får en popup-ruta på skärmen så gjorde vi ett filmklipp på vilket vi satte åtgärderna `_visible` = 1 när en viss tid har förflutit i den frame filmklippet ligger i. Detta innebär att om programmeraren till exempel har satt tiden till fem sekunder kommer rutan att bli synlig då testpersonen varit i den framen i fem sekunder. För att sedan simulera en stängning av fönstret satte vi åtgärderna `_visible` = 0 på en knapp i filmklippet. När man trycker på knappen ”stängs” rutan, det vill säga den blir inte längre synlig för användaren.

6.5 Preloader

För att inte testresultatet för en testperson skall påverkas av hur bra dator eller uppkoppling han/hon har, önskade kunden att hela filmen (testet) skall laddas ner innan tidtagningen sätts igång. För att lösa detta laddar vi in hela filmen innan testpersonen kommer vidare i testet. Detta visar vi även med en så kallad preloader, vilket är en laddare som visar hur många frames (hur många kB av filmen) som har laddats. För att kunna konstruera denna preloader fick vi än en gång ta hjälp av Internet. [28]

6.6 Formatering av text

Vår kund hade som önskemål att texterna i testet skulle kunna vara utformade på två olika sätt, länkad eller inbäddad i en rullist. Att länka texten var inget problem. Det var bara att klistra in texten på en frame, lägga in en knapp med åtgärd att gå till nästa frame och lägga in mer text. Att göra en fungerande rullist med formaterad text inuti var dock lite mer komplicerat. Vi fick först undersöka hur vi skulle gå till väga för att göra en rullist, sedan hur vi skulle lägga in texten i rullisten och därefter även hur vi skulle göra för att få texten att delas in i stycken med mera. För att kunna utforma en rullist fick vi ta en lektion på Internet igen [29]. Vi fick fel uppfattning och trodde inte att det bara var att skriva in vilken text som helst i rullisten. Vi trodde att texten måste formateras manuellt och att man som programmerare exempelvis måste lägga in radbrytningar som ”\n”. Till slut hittade vi dock en enkel lösning: det var bara att välja radbrytning i textalternativpanelen.

6.7 Svarsalternativ

En av de viktigare delarna i detta projekt är de olika svarsalternativen testpersonen kan få då han/hon skall svara på en fråga. Testpersonen kan svara genom att skriva in en helt egen text, dra en knapp mellan ett bestämt intervall och därigenom visa hur han/hon värderar frågan, eller kryssa i en checkbox.

Principen med checkbox-alternativet är den samma som med e-postknappen, om testpersonen klickar på knappen (som då skall ligga i ett filmklipp) så byter den utseende till att vara en ruta med ett kryss i. För att göra en textruta i vilken testpersonen kan fylla i vad han/hon vill så gör man en textruta, väljer att sätta denna till indatatext och ange begränsningar på antal tecken eller inte, beroende vad rutan ska användas till.

Att göra en dragknapp är dock mer komplicerat. Varje läge på draglisten, i vilken testpersonen kan släppa knappen, måste ha ett värde som skall skickas till databasen som svar på frågan. Därför måste vi sätta en variabel som förändras i förhållande till var i x- eller y- led (beroende på om dragknappen skall gå horisontellt eller vertikalt) knappen befinner sig. För att förstå hur vi skulle kunna skapa en sådan konstruktion, fick vi titta på ett exempel med volymreglage och utgå ifrån detta när vi arbetade fram vår egen dragknapp med inbyggd variabelsättning. [30]

6.8 Slumpning

Vår kund önskar testa olika personer av olika kön med detta test. Något han är mycket intresserad av är hur stor skillnad det blir på resultatet mellan en testperson som blir störd med e-post med mera i jämförelse med en testperson som slipper dessa störningar. För att göra detta skulle testpersonerna slumpas in till endera testfilmen med störningar eller till testfilmen utan. Vi har dock inte kunnat lösa detta problem på ett idealiskt sätt. Man kan slumpa in personer till filmer hit och dit, men Flash har inget minne av filmer som körs på olika datorer så därför skulle antalet inslumpade personer till ett test inte kunna regleras. Detta medför att vi inte kan se till att det blir en jämn fördelning av test med störningar och utan. För att lösa detta problem så kommer två olika lösenord att delas ut till testpersonerna, varav ett skickar testpersonen till ett test med störningar och det andra skickar testpersonen till ett test utan störningar.

6.9 Muspekare

En simulering av att testpersonens dator arbetar är också en del av testet. Detta löste vi genom att använda funktionen `mouseHide()`, vilken gömmer den vanliga pilen. För att få det att se ut som om datorn arbetar, gjorde vi ett timglas som vi ersatte pilen med. Om testpersonen flyttar musen är det alltså timglaset som rör sig och ingen pil. För att sedan göra musen obrukbar, det vill säga se till att det inte går att klicka på något, gjorde vi en frame med en fördröjning på där det helt enkelt inte finns något som testpersonen kan klicka på för att komma vidare.

6.10 Fönster

Eftersom det är mycket viktigt att testpersonerna i detta test inte får olika förutsättningar när de skall göra testet så är det viktigt att alla har lika stor text att läsa. Testpersonen skall med andra ord inte påverkas av vilken skärmstorlek han/hon har. Därför satte vi en förutbestämd storlek på vår film genom att ställa in höjd och bredd i publiceringsinställningarna. Detta hjälper dock inte helt och hållet eftersom testpersoner med större skärmar kan maximera sitt fönster och därmed få större text. Detta löste vi genom att gå in i Html-koden och sätta resizeable till noll. Nu återstod bara ett problem med fönstret och det var att testpersonen kunde stänga ner fönstret hur som helst och när som helst. Det vill säga. testpersonen kunde stänga fönstret på ”krysset” och göra andra saker på sin dator, för att sedan återgå till sitt test. Därför tog vi bort fönstrets ram. För att testpersonen inte skulle kunna klicka på något annat på sitt skrivbord så behövdes detta gömmas. Vår kund kom då på att vi kunde ha ett svart helskärmsfyllt fönster utan stängningsfunktion bakom filmfönstret. När testpersonen går in för att göra testet, öppnas två fönster, först ett svart maximerat fönster och sedan ett storleksangivet filmfönster. Anledningen till att vi tog bort ramen på det fönster som skall spela upp filmen, är att vi vill göra det svårt för testpersonerna att hoppa ur systemet så att de som gör testet fullföljer detta på en gång. Det skulle inte vara bra om en testperson fick en längre sluttid än någon annan för att de gjort andra saker under testets gång.

6.11 Design

Att utforma alla delar till detta test tog lång tid. Vi gjorde först ett mycket enkelt utseende på alla delar för att inte lägga ner onödig tid på design när vi inte ens visste hur delarna skulle komma att fungera. På grund av detta blev det en ganska intensiv period i slutet av examensarbetet då vi satte oss ner och gjorde ett enhetligt och passande utseende på testet. Vi tog hjälp av våra kunskaper från kursen systemkonstruktion, då vi lärde oss om lämpliga färger och former för ögats behag hos människan.

6.12 Manual

Den sista delen att utforma i vår produkt var att ge vår kund kunskap om hur han kan använda Flash för att göra de utseenden och kluriheter han vill generera i de olika testen. Till detta behövde vi göra en manual och även personligen visa honom hur allt fungerar. Givetvis kommer det att bli en del frågetecken i början och därför kommer vi att erbjuda vår support så gott det går under den närmaste tiden. Då vi har skrivit så pass ingående om hur man arbetar och gör test i Flash gjorde vi ingen enskild manual. Vi beslutade att vår kund kunde använda detta arbete som en manual istället.

7 Erfarenheter och rekommendationer

Att konstruera vår slutgiltiga produkt i Flash krävde mer informationssökande och undersökande än vad vi först trodde. Att hitta information om hur varje del av produkten skulle fungera var mycket tidskrävande. Vi kom dock snart underfund med hur vi intensivt och effektivt kunde hitta det vi sökte på Internet. Att söka direkt på forumen var ett mycket bra sätt eftersom andra personer ofta har haft samma problem som en själv.

Det är inte att rekommendera att göra ett verktyg/program i Flash. Flash är ett verktyg i sig själv och är främst till för att skapa film och snygga hemsidor med kluriga funktioner av olika slag. Det vi inte visste innan vi började med detta projekt var att det inte är särskilt mycket programmeringsmöjligheter i Flash. Detta har både sina för- och nackdelar. En av fördelarna är att man inte behöver sitta under lång tid för att programmera fram något grafiskt, till exempel en knapp, bara för att sedan behöva sitta ännu en lång stund för att programmera vad som skall hända när man klickar på knappen. I Flash behöver man bara tänka på det senare, nämligen vad som skall hända när man klickar på knappen. Att programmera knappens beteende är inte heller något tidskrävande arbete, det rör sig om ”kodsnuttar” på mellan tre och tio rader. En nackdel är att man inte kan anpassa variablerna i de olika framen till ”kodsnuttarna” som ligger i hela filmen. Det går till exempel inte att ha en statisk variabel utan att göra en ”ful” lösning.

Vår tidsplan har omarbetats några gånger. Vi märkte att vi från början var lite väl optimistiska då vi uppskattade hur lång tid de olika delarna skulle ta. Eftersom vi inte har så mycket erfarenhet av att uppskatta tidsåtgång, planlägga vårt eget arbete och sätta egna tidsbegränsningar för olika delmål, fungerade detta inte riktigt så bra som vi hade hoppats på. Vår första tidsplan gjorde vi då vi trodde att vi skulle programmera i Java. Nästa gjorde vi ett par veckor senare där vi justerat om lite och valt att använda oss av Flash. Den sista justeringen gjorde vi för tre veckor sedan, tidsplanerna ser något olika ut, se bilaga A.

Vi tycker att det är mycket bra att vi lärt oss Flash för framtiden. Mycket av dagens affärer och dagssysslor görs på Internet och därför behövs människor som kan göra bra utformade webbsidor åt till exempel företag. Med tanke på vilken genre vi kommer att tillhöra som dataingenjörer kan det därför vara bra att vi kan Flash när vi kommer ut i arbetslivet.

8 Slutsatser

Tyvärr kunde vi inte göra en exakt likadan lösning som vi och kunden från början hade tänkt oss. Tanken var att vi skulle göra en ”verktygslåda” åt kunden med vilken han skulle kunna plocka ihop olika delar till ett test och sedan publicera detta. Nu blev hans verktygslåda själva programmet Flash. Vi har inte heller gjort denna testgenerator säker med kontroller för oförsiktiga användare, då tiden inte räckte till för oss och kontroller inte var en högt prioriterad aspekt för kunden.

Målet med detta arbete var att göra ett oberoende test som skulle mäta huruvida olika personer påverkas negativt av störningar då de genomgår ett test. Testet skulle mäta hur pass stressade personerna blir och om de presterar ett sämre resultat när de utför ett test framför en dator som kan krångla, till skillnad från om de skulle sitta framför ett test på papper. Detta mål har vi nu nått.

Resultatet av detta examensarbete är en produkt med vilken vår kund kan utföra olika stresståhetstester på en viss utvald grupp människor.

Referenser

Böcker som vi använt oss av:

[1] Chapman N. Flash 5 Interactivity and Scripting. John Wiley & Sons, Ltd, 2001.

[2] Eidhagen K, Ek J, Wetterström A. Flash 5 Handboken. Pagina AB, 2000.

[3] Milburn K och Croteau J. Flash 4 Web Animation, f/x & Design. Coriolis, 2000.

[4] Deitel H.M, Deitel P.J, Nieto T.R. Internet & World Wide Web, How to Program. Second Edition. Prentice Hall, 2002.

[5] Bates Chris. Webprogramming Building Internet Applications. Wiley, 2001.

[6] Rosenzweig G. Special edition using Director 8. Indianapolis Ind, 2000.

[7] Deitel H.M, Deitel P.J. JAVA How to Program. Fourth Edition. Prentice Hall, 2001.

Kurser med vars hjälp vi lärt oss Flash:

[8] Macromedia Flash 5 – Hjälpmenyn.

Platser på Internet som vi besökt och där vi har hittat information som vi använt oss av:

[9] http://www.macromedia.com/support/flash/ts/documents/flash_database.htm 2002-05-25

[10] http://www.internetgrafik.nu/asp/db.htm 2002-05-25

[11] http://eforum.idg.se/viewmsg.asp?EntriesId=275749 2002-05-25

[12] http://hotwired.lycos.com/webmonkey/programming/asp/ 2002-05-25

[13] http://www.asp101.com/articles/flash/index.asp 2002-05-25

[14] http://www.litu.umu.se/stod/ 2002-05-25

[15] http://www.echoecho.com/se/flashbasics06.htm 2002-05-25

[16] http://www.gymnasium.sundsvall.se/avweb/mflash/databaser.htm 2002-05-25

[17] http://www.were-here.com/ 2002-05-25

[18] http://www.macromedia.com/se/software/ultradev/productinfo/features/automate.html 2002-05-25

[19] http://www.flash-db.com 2002-05-25

[20] http://www.tek-tips.com/gthreadminder.cfm/lev2/4/lev3/31/pid/250 2002-05-25

[21] http://eforum.idg.se/Threads.asp?list=b&forumid=181 2002-05-25

[22] http://www.tek-tips.com/gviewthread.cfm/lev2/4/lev3/31/pid/250/qid/223037 2002-05-25

[23] http://www24.brinkster.com/kunskapsbanker/snabbkurs/snabbkurs.htm 2002-05-25

[24] http://www.tek-tips.com/gviewthread.cfm/lev2/4/lev3/31/pid/250/qid/228389 2002-05-25

[25] http://hotwired.lycos.com/webmonkey/backend/databases/tutorials/tutorial3.html 2002-05-25

[26] http://www.communen.com/filer/paths/paths.php?p=2 2002-05-25

[27] http://www.communen.com/filer/flytt/lektioner/flytt/flyttis.html 2002-05-25

[28] http://www.communen.com/filer/procentpreloader01/procentpreloader01.html 2002-05-25

[29] http://tq.hkr.se/~erikdahlberg/pro/flash 2002-05-25

[30] http://www.communen.com/filer/pan_volume/pan_volume.html 2002-05-25

[31] http://www.asptutorial.info/script/aspcounter/ 2002-05-25

[32] http://hotwired.lycos.com/webmonkey/98/29/index2a.html 2002-05-25

[33] http://www.internetgrafik.nu/asp/index.htm 2002-05-25

[34] http://www.it-mattias.com/asp.html 2002-05-25

[35] http://www.aspsidan.nu/kurser/grunderna_asp.asp 2002-05-25

[36] http://designsbymark.com/ 2002-05-25

[37] http://www.developersdex.com/asp/ 2002-05-25

[38] http://www.macromedia.com/support/flash/interactivity/orderform/orderform04.html 2002-05-25

[39] http://webforums.macromedia.com/flash/categories.cfm?catid=194 2002-05-25

[40] http://www.lns.nu/flash2/ 2002-05-25

[41] http://www.macromedia.com/support/flash/sound/sound_player/sound_player07.html 2002-05-25

[42] http://internetworld.idg.se/webbskolan/skriptskolan/asp.asp 2002-05-25

[43] http://www.macromedia.com/support/flash/action_scripts/objects/date_object.html 2002-05-25

[44] http://hotwired.lycos.com/webmonkey/javascript/ 2002-05-25

[45] http://hotwired.lycos.com/webmonkey/99/13/index0a_page3.html?tw=backend 2002-05-25

[46] http://hotwired.lycos.com/webmonkey/99/13/index2a_page3.html?tw=backend 2002-05-25

[47] http://www.flash-db.com/PopUp/JavaScriptPopUp.php 2002-05-25

[48] http://www.communen.com/filer/pause/pause.html 2002-05-25

[49] http://www.actionscripts.org/tutorials/beginner/spawn_browser/index.shtml 2002-05-25

[50] http://www.communen.com/filer/flashjouren1/flashjouren1.html 2002-05-25

[51] http://disney.go.com/disneyinteractive/flash/index.html 2002-05-25

[52] http://www.thesimpsons.com/ 2002-05-25

[53] http://www.coca-cola.com/ 2002-05-25

[54] www.eforum.idg.se 2002-05-25

[55] www.communen.com. 2002-05-25

A Tidtabell

Bilagorna finns i bibliotekets arkiv