

B.B.Mo`minov, N.I.Kamalova, V.R.Rajabova



FLASH TEXNOLOGIYASI & ACTOINSCRIPT



Buxoro - 2010

Hozirgi kunda asosiy muammolardan biri animatsion obyektlar yordamida dinamik ko'rgazmali qurollar yaratish. Macromedia ishlab chiqaruvchisining Flash seriyali mahsulotlari bunday muammolarni yechishda juda qulay va keng imkoniyatlar yaratadi. Ushbu qo'llanma **Macromedia Flash** muhitida ishlash va animatsiya, video – rolik, web- sahifa, interfaolli, ovozli lavhalarni yaratish hamda **Actionscript** dasturlash tili orqali dasturlashtirilgan, boshqariladigan animatsiya, video – rolik, web – sahifalarni yaratishni o'rganishga bag'ishlangan. Qo'llanmada vektorli grafika, animatsion obyektlar haqidagi ham ma'lumot berib o'tilgan.

Ushbu uslubiy qo'llanmadan oily ta'lim muassasalarida tahsil olayotgan talabalar, litsey va kasb hunar bilim yurti o'quvchilari o'zlarining malakaviy bitiruv ishlari, kurs ishlari taqdimotlari uchun foydalanishlari mumkin. Ayniqsa turli xil imkoniyatga ega bo'lgan elektron darslik yaratishni istagan har bir kasb egasi uchun mo'ljallangan.

Mualliflar

Mo'minov Bahodir Boltaevich – BuxDU amaliy matematika va axborot texnologiyalari kafedrasida o'qituvchisi, *ped. f. n.*

Kamalova Nilufar Ilxomovna – BuxDU fizika – matematika fakulteti “Amaliy matematika va informatika” ta'lim yo'nalishining 4 – kurs talabasi

Rajabova Vazira Rajabovna – BuxDU fizika – matematika fakulteti “Amaliy matematika va informatika” ta'lim yo'nalishining 3– kurs talabasi

Taqrizchilar

Boltayev Telmon – BuxDU amaliy matematika va axborot texnologiyalari kafedrasida katta o'qituvchi

Aslonov Ulugbek – BVPKQT va MOI informatika o'qituvchisi

Ushbu qo'llanma - Buxoro davlat universitetining

- amaliy matematika va axborot texnologiyalari kafedrasida (22/02/2010, 7 – soni bayonnoma);
- fizika – matematika fakultetining ilmiy kengashida (23/02/2010, 5 – sonli bayonnoma) muhokama qilingan va nashrga tavsiya berilgan.

Mundarija

Flash texnologiyasiga kirish	4
Flash dasturi bilan ishlash asoslari	5
Flashning uskunalar paneli	7
Flashning ish maydoni	9
Vaqt diagrammasi	10
Xususiyatlar inspektori	11
Foydalanuvchi ishini tashkil etish vositalari	12
MACROMEDIYA FLASH da animatsiya yaratish	14
Interfaollik	16
Actionscript dasturlash tilining imkoniyatlari haqida dastlabki ma'lumot va Actions paneli bilan ishlash.	18
Actionscriptda o`zgaruvchilar va o`zgaruvchilarning tipi.	20
Trace funksiyasi, rolikga izoh yozish va oddiy arifmetik operatorlar	21
Flashda matematik hisoblashlarni olib borish, Math obyektining metodlari va xususiyatlari	23
Munosabat operatorlari	24
Tarmoqlanish operatori	24
Tanlash operatori SWITCH-CASE	26
Takrorlash operatorlari	27
While operatori	28
Do while operatori	28
Tugma holatlarini qayta ishlash	29
O`tish operatori va kadrlarni belgilash	32
Quyunchani boshqa hayvonlar uyidan uyigacha eltib qo`yish	34
Roliklar va videokiliplar xususiyati	35
Kiritish madonlarini hosil qilish. if, if...else shartli operatorlarini rolikda qo`llanilishi	38
Dinamik maydonlarning asosiy xossalari	39
Oddiy hisoblagichni tayyorlash	41
Scroll funksiyasi	42
Klip holatlarini qayta ishlash haqida umumiy ma'lumotlar	43
enterFrame holati	44
getTimer() funksiyasi.	45
mouseMovie vasifasi	47
Dastur yordamida flash simvollarini nusxasini hosil qilish	53
Sichqoncha ko`rsatgichi o`rniga sehirli tayoqcha hosil qilish	57

Flash texnologiyasiga kirish

Flash texnologiyasiga - **ShockWave Flash (SWF)** formatli vektorli grafikdan foydalanishga asoslangan texnologiyadir. Bu format eng samarali grafik formatlardan bo'lmada, **SWF** formati foydalanuvchilarga grafik imkoniyatlari cheklanmagan grafiklar bilan ishlovchi vositalar va natijani Web- brouzerlarda, kerakli muxarrirlarda foydalanish imkoniyatlari mavjud. **Flash** texnologiyasining imkoniyatlardan yana biri - bu uning moslashuvchanligidir, ya'ni bu format barcha platformalarda (MacOS tizimli Macintosh kompyuterlari yoki Windows tizimli kompyuterlarida) ishlatilishi mumkin. Yana bir qulay imkoniyati uning yordamida yaratilgan tasvirlar nafaqat animatsiyali bo'lishi, balki interfaol elementlar va tovush bilan boyitilishi hamda dasturlash orqali boshqarilishi mumkin.

Flash texnologiyasining moslashuvchanlik va interfaol multimediya dasturlar yaratish imkoniyati ko'pchilik Web-dizaynerlar o'rtasidagi bahslarga sabab bo'lib, uni mashhurligini oshishiga imkoniyat berdi. Shuning uchun bu texnologiyaning yaratilishi bilan bir vaqtda **Macromedia** kompaniyasi tomonidan ikki asosiy web-brouzerlari, **Internet Explorer** va **Netscape Communicator**lar uchun elementlar **Plug-In** yaratildi. Bu esa, o'z navbatida Flash texnologiyasini Internetda yana ham keng tarqalishiga olib keldi. Natijada ushbu web-brouzerlar yaratuvchilari **swf** formatini o'z dasturlarini asosiy formatlar bazasiga qabul qildi. Bunday usulni boshqa yirik dasturiy ta'minot yaratuvchilar (masalan, Adobe firmasi) ham qo'llay boshladi. Macromedia kompaniyasi **swf** formatini juda oddiy va qulay uskunalar bilan ta'minlaganligi bu formatdan ko'p muxlislarning foydalanishiga olib keldi. Shuni aytish kerakki, hozirgi vaqtda ushbu uskunalarni bir qancha to'liq to'plamlari ham mavjud. Ushbu uskunalarni bir turi **Macromedia Director Shockwave Studio** - multimediya taqdimotlarni yaratish, **Macromedia FreeHand** va **Macromedia Fireworks** - grafik tasvirlar muharriri, **Macromedia Authorware** va **Macromedia CourseBuilder** - interfaol o'rgatuvchi kurslarni yaratish muharriri va boshqalarni misol qilib olish mumkin. Web-sahifa yaratuvchilar orasida eng ko'p ishlatiladigani bu **Macromedia Flash** dasturidir, chunki ushbu dastur ixtiyoriy Web-sahifaga mashhurlik olib keluvchi banner va animatsiya, interfaol lavhalar yaratish imkonini beradi. Balki shuning uchundir **swf** formatini oddiy qilib Flash deb atalish odatga kirib kolgan.

Flash texnologiyalar tarkibining elementlari:

- vektorli grafika;
- animatsiyani bir qancha usullarda ishlash;
- interfeysda interfaol elementlarini yaratish;
- sinxron ovoz qo'shish;
- HTML formati va boshqa internetda foydalaniladigan barcha formatlarga o'tkazishni ta'minlash;
- mustaqil platformali;
- Flash-roliklarni avto rejimda ham, Web - brouzer yordamida ham ko'rish imkoniyati mavjud;
- vizual uskunalari mavjudligi Flash-rolik yaratuvchilarini ko'plab murakkab amallardan xalos etadi, shuningdek Flash-texnologiyalarning texnik asneklarni o'rganishni talab etmaydi.

Hozirgi vaqtda Web-sahifalarni yaratishda birinchi o`rinlardan birini **rastrli grafika** egallaydi. Rastrli formatlardan GIF (**Graphics Interchange Format** - ma'lumotlar almashuvi uchun grafik format), JPEG (**Join Photographic Experts Group** – tasvir bo'yicha mutaxassislar birlashgan guruhi) va PNG (**Portable Network Graphics**- ko'chirma grafik format) va boshqa formatlarni keltirish mumkin. Rastrli grafikani ishlatishda tasvir nuqtalar majmuasi (piksellar – inglizca pixels) dan iborat bo'ladi. Bu nuqtalar bir - biri bilan bog'liq bo'lmaganligi uchun ushbu nuqtalarni har birga rangi va koordinatasi berilishi kerak. Oddiy holda, agar ikki xil rangli tasvir ishlatilsa (masalan, oq-qora), u holda har bir pikselni ta'riflash uchun bitta ikkili razryad (0- qora, 1- oq) ta'riflash etarli bo'ladi. $256 = 2^8$. Juda ham murakkab fotorealistik rangli tasvirlar 1 pikselga 24 razryad talab qiladi. Natijada rastr tasvirli fayllar o'lchami tasvirni rang chuqurligi o'sgani sari oshib boradi. Rastrli tasvirlarni yana bir kamchiligi shundan iboratki, tasvir sifati piksel o'lchamiga bog'liq, u esa o'z navbatida monitorni imkoniyati bilan belgilanadi. Shuning uchun bir xil rasm turli monitorlarda har xil ko'rinishga ega bo'lishi mumkin. Rastrli tasvir o'lchamini o'zgartirish juda ham murakkab ishdir. Chunki bunday tasvirni kattalashtirish piksellar sonini o'sishiga olib keladi. Kompyuter grafikasi sohasidagi mutaxassislar tomonidan juda murakkab rastrli tasvirlar piksellar «ko' naytirish» yoki “o'chirish (agar tavirni kichraytirish kerak bo'lsa)” algoritmlari ishlab chiqilgan, lekin ular doim ham ushbu masalani oqilona bajara olmaydi.

Web-sahifalarni yaratishda birinchi o`rinlardan birini **vektorli grafika** ham egallaydi. Bu tasvirni rasmdagi joylashuvi matematik formulalar bilan berilgan egri chiziqlar majmuasi yordamida namoyish etish usulidir. Masalan, istalgan doirani tasvirlash uchun uch-to'rt raqam kerak bo'ladi: radius, markaz koordinatalari va chiziq qalinligi. Shuning uchun, vektorli grafika rastrli grafikaga nisbatan bir qancha afzalliklarga ega:

- vektorli tasvirlarni belgilovchi matematik formulalar kompyuter xotirasida rastrli tasvir piksellariga qaraganda kamroq joy egallaydi;
- tasvir (yoki uning ayrim qismlarini) sifatini yuqotmasdan chegaralanmagan kattalashtirish imkoniyati mavjudligi;
- tasvirni bir platformadan ikkinchisiga ko'chirishning qulayligi.

Albatta, vektorli tasvirlarni o'z kamchiliklari ham mavjud. Masalan, fotorealistik tasvirni vektorli formatda namoyish qilish murakkabroq. Flash yaratuvchilari bunga echimni tonishgan. Flash yordamida Web – sahifalar tuzishda siz nafaqat vektorli balki rastrli tasvirlarni ishlatishingiz ham mumkin.

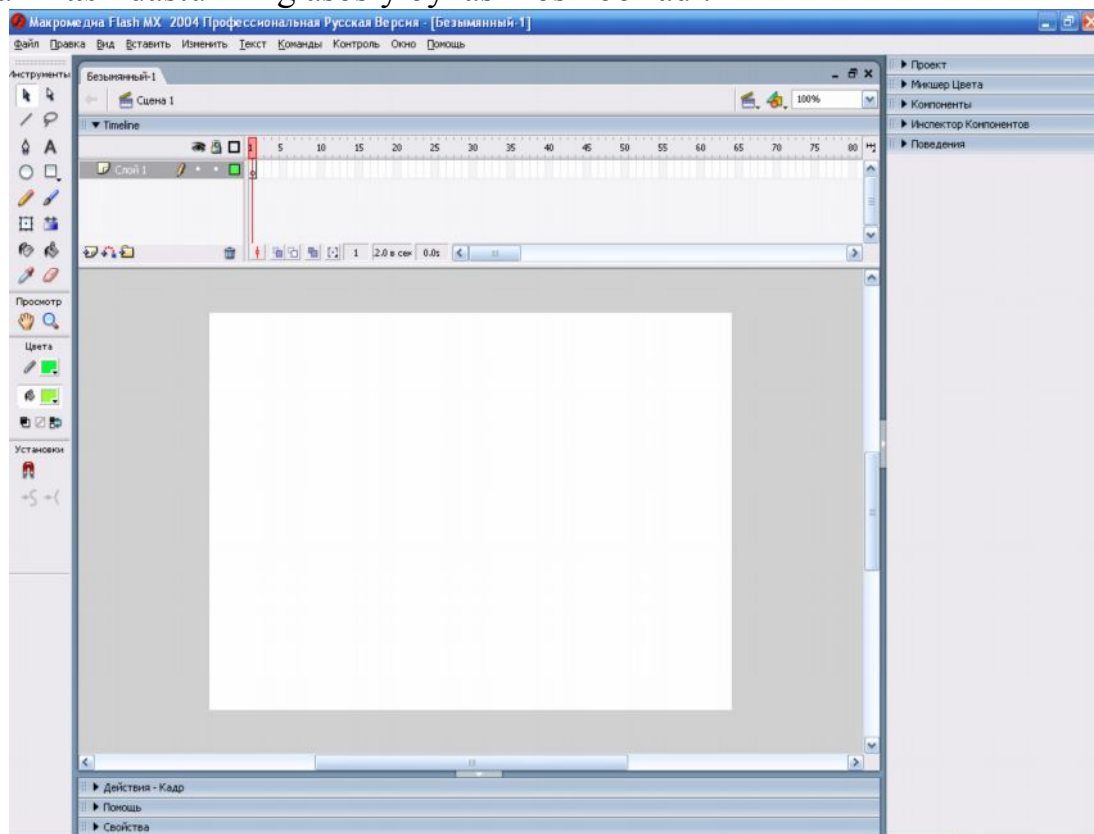
Flash dasturi bilan ishlash asoslari

Flash dasturida ishlashni o`rganish uchun avvalo uning interfeysi bilan tanishish lozim. Agar Macromedia firmasi tomonidan yaratilgan biror-bir dasturiy ta'minotlar (masalan, Dreamweaver muxarriri) bilan tanish bo'lsangiz Flashni interfeysini o'zlashtirish unchalik murakkab bo'lmaydi. Agar Macromedia tomonidan yaratilgan uskunalarni o'zlashtirishni aynan Flash dasturidan boshlasangiz, u holda uning menyulari, uskunalari va ob'ekt xususiyatlar paneli va boshqalarning ko'rinishi Windowsning amaliy dasturlarinikidan farq qiladi. Masalan, aksariyat Windowsdagi

amaliy dasturlarda (ofis majmuasi) uskunalar paneli tarkibini boshqarish buyruqlari “View” menyusiga kiradi. Flash dasturida esa bu buyruqlar “Window” menyusida joylashtirilgan. Tahrirlanayotgan obyekt xususiyatlar panelining ko`rinishi ham Flash dasturiga xosdir. Lekin bu turdagi yangiliklar unchalik ham qiyinlik tug`dirmaydi.

Muharrirning foydalanuvchi interfeysini tashkil etilishi umumiy holda Windows dasturlar kabi standart ko`rinishga egadir, oynaning yuqori qismida sarlavha, menyular to`plami, asosiy buyruqlar joylashtirilgan uskunalar paneli va boshqalar. Muharrirda yangi vaqt diagrammaning ko`rinishi mavjud.

Muxarrirni ilk bor ishga tushirganda asosiy oyna ustidan qo`shimcha muloqot oynasi naydo bo`ladi. Bu oyna sizga ish boshlashni o`zinigiz xohishingiz bo`yicha sozlash imkonini beradi. Bu muloqot oynasida 3 ta **Open a Recent Item** (oxirgi flash faylni ochish), **Create NEW** (yangi fayl yaratish), **Create form Template** (shablondan foydalanib fayl yaratish) bo`limlari bor. Bu muloqot oynasidan keraklisini tanlab olish mumkin. Aksiriyat holatlarda **Create NEW -> Flash document** bandi tanlanadi va ekranga **Flash** dasturining asosiy oynasi hosil bo`ladi.



1-рasm. Flash dasturining asosiy interfeysi

Flash dasturining asosiy oynasi menyular, uskunalar, obekt xususiyatlar va qo`shimchalar, ishchi maydon kabi 4 ta asosiy sohalardan iborat.

Flash dasturining menyusi 10 bo`limdan iborat bo`lib, ular quyidagi jadvalga keltirilgan vazifalarni bajaradi.

1- jadval

Flash dasturining menyu bo`limlarining vazifalari

T.r.	Nomi	vazifasi
1	Файл	Yangi Flash hujjat yaratish, ochish, yopish, xotiraga olish usullari, tasvirlarni import va eskport usullari, publikatsiya, sahifaning xususiyatlari kabi ishlarni amalga oshiradi.

2	Правка	Flash hujjatdagi tahrirlash ishlarini amalga oshirish (bekor qilish, qaytarish, nusxalash, qirqish, qo'yish usullari, qidirish usullari va boshqalar)
3	Вид	Flash hujjatning ko'rinishini o'zgartirish amallarini bajaradi (masalan, o'lchamini kattalashtirish yoki kichiklashtirish, chegaralash va boshqalar)
4	Вставить	Flash hujjatga qo'yiladigan obyektlar qo'yish (yangi belgi – rolik, tugma, tasvir va boshqalari)
5	Изменить	Flash hujjatdagi obyektlarni tahrirlash, o'zgartirish amallarini bajaradi.
6	Текст	Flash hujjatdagi matnlarning yozuvi, o'lchami, stili va boshqa amallarni bajarish uchun mo'ljallangan.
7	Команды	Flash hujjat uchun turli buyruqlarni bajartirish va natijalar olish uchun mo'ljallangan.
8	Контроль	Flash hujjatni boshqarish amallari uchun mo'ljallangan.
9	Окно	Flash interfeysining ko'rinishi ustida amallarni bajarish uchun mo'ljallangan.
10	Помощь	Flash hujjat bo'yicha yordam tizimi.

Flashning uskunalar paneli

Flash dasturining uskunalar paneli tasvirlarni yaratish va tahrirlash kabi amallarni bajarishga yordam beradi. Uning tarkibiga quyidagi jadvalda keltirilgan uskunalar kiradi. Tahrirlash uskunolari paneli asosiy ekranni chan tomonida joylashadi. Unda asosan grafik obyektlar yaratish va tahrirlash uchun mo'ljallangan uskunalar joylashtirilgan. Bu uskunalarining ko'chiligi foydalanuvchiga grafik muxarrirlari (masalan, Paint) bilan ishlash jarayonidan tanishdir. Foydalanuvchiga qulaylik tug'dirish uchun uskunalar paneli to'rt qismga bo'lingan.

✚ **Tools (Инструменты)** – bu qismda aniq uskunalar joylashtirilgan. Bu uskunalarini tanlash va chizish uskunolari tashkil etadi.

✚ **View (Просмотр)** – ish stolidagi tasvirni ko'rishni boshqarish vositalari joylashtirilgan. View ikki uskunadan iborat: **Hand Tool**- bu uskuna tanlanganda maxsus rejim yoqiladi, bu rejimda sichqoncha yordamida ish maydonini turli yo'nalish bo'yicha siljitish mumkin; **Zoom Tool** – bu uskuna tanlanganda ish maydonidagi tasvirni tez masshtablash rejimi yoqiladi. Bu rejim yoqilganda Options maydonida ikki qo'shimcha uskunasi naydo bo'ladi, bu uskunalar masshtablash yo'nalishini (kattalashtirish yoki kichiklashtirish) o'zgartirish imkonini beradi.

✚ **Colors (Цвета)** – bu erdagi uskunalar chegara va maydon asosining rangini bir-biridan alohida o'zgartirish imkonini beradi.

✚ **Options (Установки)** – tanlangan uskuna uchun qo'shimcha parametrlarni o'rnatish elementlari joylashtirilgan. Qo'shimcha parametrlari bo'lmagan uskunalar uchun Options maydoni bo'sh qoladi.

Flashni avvalgi variantlariga nisbatan, chizish uskunalarini ayrim parametrlarini o'rnatish xususiyatlar inspektori paneli yordamida ham amalga oshirish mumkin. Har

bir uskunani sozlashni o`ziga xos tomonlari keyingi mavzularda ko`rib o`tiladi. Agar zaruriyat tug`ilsa uskunalar panelini oynadan «uzish» va istalgan joyga joylashtirish mumkin.

2-jadval

Flash dasturining uskunalari

T.r.	Rasmi	Nomi	Tugmasi	Vazifasi
1		Выделение	V	Obyektlarni belgilash uchun ishlatiladi, lekin selection tool dan farqli tomoni shundaki, selection tool da rasmning biror bir qismini alohida ajratib olish mumkin, lekin subselection tool asbobini tanlab rasmning biror qismini ajratib ko`rsatsak rasm butunligicha belgilanadi, ya`ni uning ma`lum qismini ajratib ko`rsatib bo`lmaydi;
2		Субвыделение	A	Obyektlarni belgilash va ularni bir joydan boshqa joyga ko`chirish uchun ishlatiladi. Bundan tashqari biror rasmni chetki konturini yoki ichki bo`yalgan qismini belgilab olib uning rangini o`zgartish mumkin;
3		Линия	N	Chiziq chizish;
4		Лассо	L	selection tool kabi obyektlarni belgilash uchun ishlatiladi, lekin selection tool dan farqli ravishda obyektни belgilash uchun shu obyekt ustiga sichqonchani bir marta chertish emas, balki uni to`lalligicha aylana ichiga olib, so`ngra sichqonchani tugmadini qo`yib yuborish kerak;
5		Ручка	P	turli xil chiziqlarni birlashtirib chizish uchun ishlatiladi;
6		Текст	T	Matn yozish;
7		Овал	O	Oval chizish;
8		Прямоугольник	R	To`rt (ko`pburchak) burchak chizish;
9		Карандаш	Y	Qalam;
10		Кисть	B	mo`yqalam asosan rasmning biror qismini ranglash, yoki qalin chiziqlar chizish uchun mo`ljallangan;
11		Свободная трансформация	Q	obyektlarni belgilab, bir joydan boshqa joyga o`tkazish, o`lchamini o`zgartirish, markaziy nuqta atrofida aylantirish uchun ishlatiladi;
12		Трансформация заливки	F	Fon gradiyentasini o`zgartirish;

T.r.	Rasmi	Nomi	Tugmasi	Vazifasi
13		Чернильница	S	Siyoxdonni ishlatish;
14		Paint Bucket Tool	K	chegaralangan sohani ranglash uchun ishlatiladi;
15		Пипетка	I	biror rangdan nusxa olish;
16		Ластик	E	o`chirg`ich;
17		Рука	H	qo`l;
18		Масштаб	M,Z	Masshtab;
19		Цвет штриха	-	Obyekt chegarasini rangi;
20		Выделение	-	Obyekt fonni (ostiki fon) rangi;
21		Субвыделение	-	Obyekt oq-qora rangda;
22		Линия	-	Obyektda rang yo`q;
23		Лассо	-	Obyekt rangni o`zgartirish;

Flashning ish maydoni

Ish maydon interfeysning markaziy qismida joylashgan. Ish maydonda obyektlarni tahrirlashning barcha amallarni bajariladi. Ish maydonining yuqorisida uskunalar paneli joylashtirilgan. Bu panelda quyidagi interfeys elementlari joylashgan (2- rasm, chapdan o`ngga):



2-rasm. Oynalarni boshqarish

- Yo`nalishli tugma - butun sahnani tahrirlash rejimiga qaytish imkonini beradi va faqat agar siz alohida elementni tahrirlash rejimida bo`lsangiz ishlaydi;
- tahrirlanayotgan sahna (simvol joylashtirilgan sahnani) nomi yozilgan matnli maydon; roliklarin sahnalariga avtomatik ravishda Сцена (Scene) nomi va sahna tartib nomeri beriladi;
- tahrirlanayotgan simvol nomi yozilgan matn maydoni. Agar simvol o`z ichida boshqa simvollardan tashkil tongan bo`lsa u holda ichma-ich joylashgan simvolni tahrirlashga o`tilsa panelda simvollar nomidan iborat zanjir hosil bo`ladi;
- sahnani tanlash tugmasi bosilganda rolik sahnalaridan iborat ro`yxatli menyu ochiladi;
- simvolni tanlash tugmasi bosilganda rolik simvollaridan iborat ro`yxatli menyu ochiladi.
- ish maydonining masshtabini tanlash ro`yxati, bu ro`yxatdagi ko`rsatkichlarni o`zgartirish mumkin.

Ish maydonining xohlagan joyida sichqonchani o`ng tugmasi bosilganda lokal menyu ochiladi, bu menyu ish maydoni va turli rolik parametrlarini o`zgartirish uchun asosiy buyruqlardan iborat.

Scene (sahna) buyrug'ini tanlanganda nomi Scene nomli panel naydo bo'ladi, bu panel rolik sahnalari bilan ishlash uchun mo'ljallangan. Uning yordamida quyidagi amallarni bajarish mumkin:

- roliklar sahnalari ro'yxatini ko'rish; sahnalarni ro'yxatdagi joylashuvi rolikdagi ketma- ketlikka to'g'ri keladi;
- rolikni xohlagan sahnasiga o'tish;
- tanlangan sahnani nusxasini olish. (panelni pastki qismidagi tugma bosiladi, nusxa ro'yxatga sahnaning haqiqiysidan so'ng joylashtiriladi);
- yangi sahna qo'shish (sahna ro'yxatda tanlangan sahnadan keyin joylashadi);
- tanlangan sahnani o'chirish va boshqa amallar;

Ish maydoning markaziy qismida montaj stoli (Stage) joylashgan. Montaj stolining o'lchami va rangi rolikni namoyish vaqtidagi ekran o'lchami va rangiga to'g'ri keladi. Bir vaqtda montaj stolida faqat bir sahna joylashishi mumkin.

Flash ko'p hujjatli interfeys - MDI (Multi Document Interface) texnologiyasidan foydalaniladi. Bunday ish modeli bir asosiy oyna ichida bir nechta fayllar - qo'shimcha oynalarni tahrirlash uchun ochish imkoni borligini bildiradi. MDI bir qancha qulayliklarni yaratib beradiki, ulardagi rolikni bir – biriga, boshqasiga sichqoncha yordamida (ya'ni drag-and-drop - «olib o't va qoldir»ni qo'llash) elementlarni ko'chirib o'tish mumkin. Ob'ektni boshqa bir rolikga ko'chirib o'tilganda yoki nusxasi olinganda, bu ob'ektni barcha xususiyatlari saqlanib qolinadi.

Vaqt diagrammasi

Ish maydonining yuqori qismida vaqt diagrammasining (Timeline) paneli joylashgan. Ish stoli kabi vaqt diagrammasi ham rolikning aniq bir sahnasiga tegishli bo'ladi. Vaqt diagrammasi sahnadagi kadrlar joylashuvining tartibi, obyektlar holatini o'zgarish ketma-ketligi va boshqa amallarni bajarish uchun mo'ljallangan. Vaqt diagrammasi animatsiyani yaratishda va rolikni interfaol elementlar holatini tavsiflaydigan asosiy vositadir. Vaqt diagrammasi paneli murakkab tuzilishga ega bo'lib, ko'p boshkaruv elementlarini o'z ichiga oladi. Bundan tashqari, vaqt diagrammasini tashqi ko'rinishi o'rnatilgan parametrlarga qarab juda keng diapazonda o'zgarishi mumkin.

Vaqt diagrammasida quyidagi elementlar mavjud:

- rolikni ayni vaqtdagi sahnasining kadrlarini ta'rifi;
- kadrlar shkalasidan iborat vaqt diagrammasi, “o'qish kalagi” va sahnani har bir kavati uchun vaqt diagrammasi;
- vaqt diagrammasida kadrlar namoyish etish formatini tanlash menyusi;
- vaqt diagrammasi oynasini holat satri. bu satrda rolikni bir necha parametrlari to'g'risidagi ma'lumot va ish stolidagi animatsiya kadrlarini boshqarish tugmalari.

Vaqt diagrammasini ko'rinishi kadrlarni tanlangan formatiga ham bog'liq bo'ladi. Bu formatlarni tanlash **Frame View** menyusi yordamida amalga oshiriladi. Bu menyu buyruqlari vaqt diagrammasining barcha qatorlariga barovar ta'luqlidir. Menyu punktlari to'rtta guruhga bo'linadi.

Birinchi guruhdagi buyruqlar kadr yacheykasi razmerini gorizontal yo'nalishda o'rnatilishini bildiradi, quyidagi variantlar mavjud:

- **Tiny** (juda kichik);
- **Small** (kichik);
- **Normal** (oddiy);
- **Medium** (o`rta);
- **Large** (katta).

Large formatini tovush parametrlarini ko`rish uchun ishlatiladi.

Ikkinchi guruhga bitta buyruq kiradi - **Short** (qisqa). Bu buyruq kadr yacheykasi razmerini vertikal yo`nalish bo`yicha kichiklashtirishni bildiradi.

Uchinchi guruhga ham bitta buyruq kiradi - **Tinted Frames** (bo`yalgan kadrlar). Ushbu buyruq tanlanganda animatsiyani turli xillari uchun kadrlarning rangli indikatsiyasi ishlatiladi.

To`rtinchi guruh ikki buyruqdan iborat:

- **Preview** (avvaldan ko`rish – предварительный просмотр) - bu buyruq qo`llanilganda vaqt diagrammasida animatsiyani asosiy kadrlarni kichik ko`rinishi hosil bo`ladi;

- **Preview In Context** (предварительный просмотр в контексте) - bu buyruqni agar har bir asosiy kadrda ikki va undan ortiq obyektlar o`zgarib tursa ishlatilishi maqsadga muvofiq, chunki bu buyruq vaqt diagrammasida kadr yacheykasiga sahnani barcha joyini tonshiriladi, lekin bu holda kadr tasviri **Preview** buyrug`i ishlatilgandagidan ham kichikroq bo`ladi.

Xususiyatlar inspektori

Ish maydonning pastki qismida xususiyatlar inspektori paneli joylashgan. Agar ish maydonida biror-bir obyekt tanlanmagan bo`lsa yoki obyektlar umuman bo`lmasa, u holda xususiyatlar inspektori faqat rolikning umumiy xossalari ko`rsatadi. Birob-bir obyekt tanlanganda xususiyatlar inspektori formati avtomatik ravishda o`zgaradi. Xususiyatlar inspektori paneli interfeysdan ajratib olish va uni ixtiyoriy qismida joylashtirish mumkin.



3-rasm. Xususiyatlar inspektori paneli

Bu holda xususiyatlar inspektori paneli quyidagi elementlardan tashkil tongan bo`ladi:

- **Document** (hujjat) matn maydoni, bu maydonda tahrirlanayotgan fayl nomi aks ettiriladi;

- **Size** (o`lcham) tugmasi, bu tugma ish stolini (Flash-rolikni ko`rish uchun sahna o`lchami) joriy o`lchamni ko`rsatadi. Bu tugma bosilganda qo`shimcha **Document Properties** (hujjat xossalari) muloqot oynasi ochiladi, uning yordamida ish stoli o`lchamini o`zgartirish va boshqa qo`shimcha parametrlarni o`rnatish mumkin;

- **Publish** (Публиковать) tugmasi, bu tugma bosilganda rolikni avval o`rnatilgan parametrlari bilan birga publikatsiyasi (namoyishi) amalga oshiriladi. Rolikni publikatsiyasi Flash-rolikni **FLA**-formatidan **SWF**-formatiga konvertatsiya qilishdan iboratdir, bundan tashqari avtomatik ravishda rolikdan iborat Web-saxifa (HTML-fayl)

yaratiladi. Tugmada nublikatsiyani asosiy parametrlaridan biri- Flash-Player varianti aks ettiriladi;

- **Background** (Fon) tugmasi, bu tugma bosilganda rolik foni (ya'ni, ish stoli rangi)ni tanash uchun rang majmuasi oynasi ochiladi;

- **Frame Rate** (Kadrlar tezligi) matn maydoni. Bu maydon yordamida animatsiyani kadrlari almashish chastotasini o'rnatilishi mumkin (chastota sekund/kadr - frame per second, fps bilan o'lchanadi);

- so'roq belgisi [?] tasviri tushirilgan tugma, HTML-xujjat formatidagi kontekst ma'lumotnomani chaqirish uchun xizmat qiladi;

- yulduzcha tasviri tushirilgan tugma, bu tugma bosilganda **Accessibility** (**Dostunnost**) paneli ochiladi, bu panel rolikning barcha elementlariga qo'shimcha parametrlarni o'rnatish uchun xizmat qiladi. Masalan, ovoz o'rniga ekranda paydo bo'luvchi tegishli yozuvlar va boshqalar.

Foydalanuvchi ishini tashkil etish vositalari

Flash tarkibiga kiruvchi foydalanuvchi ishini tashkil etish vositalari ushbu dastur bilan ishlashni o'rganish tezlashtirish, vaqtini kamaytirish, shuningdek ishlash qulayligini va ish unumdorligini oshirishga mo'ljallangan. Bu maqsadlarga erishish uchun yordam tizimi kerakdir. Flashning bu tizim bir necha xilda tayyorlangan:

- HTML formatidagi interfaol ma'lumot tizimi;
- HTML-sahifalar ko'rinishidagi interfaol kitob;
- 10 qismli Flash-rolik ko'rinishidagi interfaol darslar;
- Web orqali tezkor texnik ta'minot;
- Flash-roliklar misollar to'plami (bu misollar nafaqat Flashning imkoniyatlarini ko'rsatadi balki yangi roliklar yaratish uchun asos bo'lib ham hisoblanadi).

Flash yaratuvchilarning barcha yaratgan qulayliklariga qaramay, hamma foydalanuvchilar talablarini qondirish mushkul albatta. Foydalanuvchining tayyorgarlik darajasi, maqsadli rolikning o'ziga xos tomonlari va boshqalari o'z ta'sirini ko'rsatadi. Shuning uchun Flash yaratuvchilari foydalanuvchilar muxarrirning o'zi uchun qulay bo'lgan ko'rinishini o'rnatish imkonini yaratishgan. Juda ko'p foydalaniladigan parametrlarga quyidagilarni kiritish mumkin:

- ekranda joylashgan uskunarlar panellari tarkibi va formatini o'zgartirish;
- ish stolidagi tasvirlarni namoyish etish o'lchamlarini o'zgartirish;
- tugmalarni belgilash.

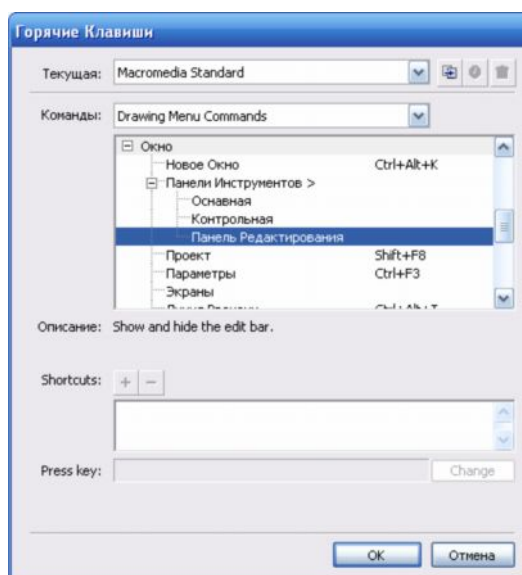
Uskunarlar panelini o'zgartirish. Flash texnologiyasida 3 xil konfiguratsiya ishlab chiqarilgan: dizayner (Designer) uchun, yaratuvchi (Developer - разработчик) uchun va universal konfiguratsiya (General). Agar har bir konfiguratsiya uchun qisqa ma'lumotni olmoqchi bo'lsangiz "**Help**" menyusidan **Welcome** (Xush kelibsiz) buyrug'ini tanlang. Bunda Flash yordamida yaratilgan muloqot oyna ekranda naydo bo'ladi. U uch bo'libdan tashkil tongan menyudan iborat, bu bo'limlar tegishli konfiguratsiyani sozlash uchun mo'ljallangan. Konfiguratsiyani sozlash rejimiga o'tish uchun quyidagilarni ketma – ketlikni bajarish lozim:

Asosiy oynani **Window** menyusida **Panel Sets** (panellar to'plami) buyrug'ini tanlang.

Ochilgan menyuda kerakli variantni tanlang, universal konfiguratsiyaga **Default Layout** punkti tegishlidir; **Designer** va **Developer** variantlari uchun uchtdan modifikatsiya nazarda tutilgan.

Barcha panellar ro'yxati asosiy oynaning **Window** bo'limida joylashgan. Panellarni o'z xohishingiz bo'yicha joylashtirganingizdan so'ng, tanlangan konfiguratsiyani keyingi ish seanslarida foydalanish uchun saqlab qo'yishingiz mumkin. Konfiguratsiyani saqlash uchun quyidagilarni bajarish kerak **Window** menyusida **Save Panel Layout** (panel konfiguratsiyasini saqlash) buyrug'ini tanlaniladi. Foydalanuvchi panellar konfiguratsiyasi haqidagi ma'lumot **Flash\First Ruu\Panel Sets** papkasida berilgan nom ostida saqlanadi. Shuning uchun konfiguratsiya nomi siz ishlayotgan tizimning talablariga to'g'ri kelishi kerak.

Tugmalarni belgilash. Grafik foydalanuvchi interfeyslari keng tarqalishi bilan dasturlardagi amalga oshiriladigan funktsiyalarni ko'pi «sichqoncha»li bajarilishga o'tib borilmoqda. Lekin amaliyot shuni ko'rsatmoqdaki, ba'zi bir ko'p ishlatiladigan buyruqlarni bajarish uchun "tugmalar (горячие клавиши, **shortcut keys** yoki **Hotkeys**)larni ishlatish qulayroqdir. Flashda foydalanuvchiga shaxsiy Hotkeyslar kombinatsiyalarini belgilash imkoniyati yaratilgan. Bunday belgilash maxsus muloqot oyna orqali amalga oshiriladi, bu oyna **Edit** menyusida tarkibiga kiritilgan **Keyboard Shortcuts...** buyrug'i orqali chaqiriladi. Bu oyna interfeysni quyidagi elementlaridan tashkil tongan.



4-rasm. Keyboard Shortcuts... muloqot oynasi

Current Set (joriy to'plam) ro'yxati oldindan belgilangan **Shortcuts**lar kombinatsiyasidan birini tanlash imkonini beradi, bu kombinatsiyalar quyidagi grafik muharrirlarda ishlovchilar uchun tanishdir:

- **Macromedia Fireworks 4;**
- **Macromedia FreeHand 10;**
- **Adobe Illustrator 10;**
- **Adobe Photoshop 6.**

Bundan tashqari **Macromedia Standard** va **Flash 5** variantlari nazarda tutilgan. Commands (buyruqlar) ro'yxati belgilashning qaysi bir buyruqlar to'plami uchun bajariladi, bunday to'plamlar to'rtta:

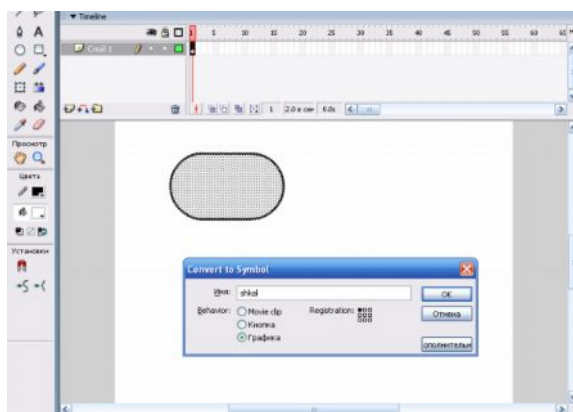
- **Drawing Menu Commands** (chizish menyusida buyruqlari);

- **Drawing Tools** (chizish uskunolari);
- **Test Movie Menu Commands** (roligni tekshirish menyusi buyruqlari);
- **Actions Panel Commands** (Actionscript muxarriri buyruqlari).

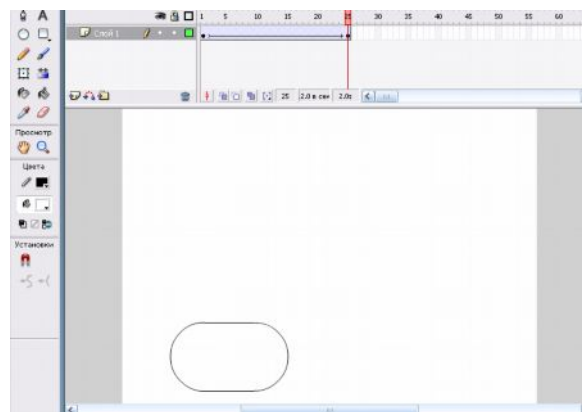
Tanlangan variantdan quyida joylashgan ro'yxatda namoyon bo'ladigan buyruqlar to'plami bog'liqdir. Agar **Drawing Menu Commands** varianti tanlangan bo'lsa, u holda ro'yxatda **Flash** asosiy oynasi menyusini barcha buyruqlari (**File**, **Edit** va boshkalar) keltirilgan. Biror - bor menyuga o'tish uchun, menyuning chan qismida joylashgan [x] belgisini tanlash kerak.

MACROMEDIYA FLASH da animatsiya yaratish.

Macromedia Flash dasturida animatsiya ikki usulda yaratiladi: kadrli (покадровое создание) va avtomatik (автоматическое создание промежуточных кадров). Avtomatik animatsiya shakllar geometriyasini uzgarishi (**shape tweening**) yoki boshqaruv kadrlar o'zgarishi (**motion tweening**) asosidagi animatsiya turlarga bo'linadi. Boshqaruv kadrlar o'zgarishi (**motion tweening**) asosidagi yaratilgan animatsiya. Shu turdagi animatsiyani yaratish uchun biz bitta boshqaruv kadrni yaratamiz va unga belgi qo'shamiz. Masalan, boshqaruv kadrda biror bir shakl (aylana) chiziladi va u **grafik tasvir** belgi turiga «Вставит» menyusida «Преобразовать в символ (**Convert to Symbol**)» buyrig'i yordamida (F8 tugmasi) o'tkaziladi. Yoki «Вставит» menyusida «Новый символ (**New symbol**)» buyrig'ini (Ctrl+F8 tugmachalar majmuasi) tanlab yangi belgi yaratamiz va belgilar kutubxonasi yordamida uni boshqaruv kadrda qo'shamiz. Obe'ktning chegarasini rangini tanlash uchun  uskunasi, asosni ranglash uchun  uskunasi,  - chegara va asos rangini almashtirish, o'zaro rang almashish  uskunalaridan foydalanamiz.



5 – rasm. Shaklni grafik tasvirga o'tkazish



6 – rasm. Shaklga harakat berish

Endi belgi joylashgan boshqaruv kadrni sichqonchani o'ng tomondagi tugma yordamida tanlab «**Create motion tweening**» nomli buyrug'ini tanlaniladi. Shu harakatlar natijasida boshqaruv kadr rangi ko'k rangga o'zgaradi. Endi sichqoncha bilan yangi kadrni tanlaniladi, (masalan 25-chi kadrni) va F6 yoki Вставка menyusida ключевой кадр (Insert keyframe) aktiv qatlamda keyingi boshqaruv kadrini yaratish buyrug'ini tanlaymiz. Natijada 25-chi kadrda ko'k rangli boshqaruv kadr hosil qilinadi va shu kadrgacha birinchi boshqaruv kadrda strelka hosil bo'ladi. Birinchi boshqarish kadrda ikkinchi boshqarish kadrgacha kadrlar ko'k rangda avtomatik xosil qilinadi.

Oxirgi xarakatimiz - bu ikkinchi boshqarish kadrda belgini o'zgartirish (cho'zish, aylantirish, kattalashtirish, kichkinalashtirish yoki kadrda joylanishini o'zgartirish). Endi klaviaturadagi Enter tugmasini bosamiz va yaratilgan animatsiyani ko'rishingiz mumkin. **Swf** formatga o'tkizib, ko'rish uchun [Ctrl] + [Enter] tugmachalar majmuasi bosiladi. Bu animatsiyadan foydalanish uchun «1-animatsiya.swf» fayli hosil bo'ladi.

Shakllar geometriyasini o'zgarishi «**shape tweening**» asosidagi yaratilgan animatsiyadir. Bu turdagi animatsiyani yaratish uchun boshqaruv kadrlar o'zgarishi «**motion tweening**» asosidagi yaratilgan animatsiya xosil qilinadi. Faqat endi oxirida ikkinchi qatlamdagi belgini butunlay o'chirib, uning o'rniga kvadrat chizamiz. Shu xarakatdan keyin kadrlar rangi o'rta rangga qaytadi. Keyin birinchi va ikkinchi boshqaruv kadrlardagi **grafik tasvir** belgini [Ctrl]+[B] («изменить» menyusidagi «разделит отдельно (Break appartack)» buyrug'i yordamida alohida shakllarga bo'lib chiqamiz.

Endi avval birinchi boshqaruv sichqoncha chap tugmasi bilan tanlab «Свойства (Properties)» yoki [Ctrl]+[F3] yoki Окно menyusining shu nomli buyrug'ini tanlaymiz. Natijada muloqot oynasi hosil qilinadi va unda **Tweening** sohasida **Motions** o'rniga Shape holatini tanlaymiz. Shu natijasida boshqaruv kadr rangi yashil rangga o'zgaradi. Endi ikinchi boshqaruv kadrni ham sichqoncha bilan tanlab animatsiya turini Motions dan Shapega o'zgartiramiz va oxirida klaviaturadagi [Enter] tugmasini bosamiz va biz yaratgan animatsiyani ko'rishimiz mumkin.

Yuqoridagi ikkala animatsiyalarda ham ikkita boshqarish kadrlar o'rtasidagi masofa kadrlarni kompyuter o'zi avtomatik yaratdi va animatsiya to'g'ri chiziq bo'yicha harakatlandi. Ammo agar bizga harakat biror bir traektoriya bo'yicha bajarilishi kerak bo'lsa, u holda bunday animatsiyalarni hosil qilish uchun qatlam ustida maxsus  **xarakat traektoriya qatlamini** yaratish kerak bo'ladi. Shu katlamda  qalam bilan kerakli traektoriya chizig'i chiziladi. Natijani ko'rish uchun [Enter] tugmasini bosamiz.

Shu bilan birga avtomatik animatsiyalarda foydalanish mumkin bo'lgan yana bir effekt mavjud - bu maska qatlami. Maska qatlami uchun asosiy qatlam ustida yangi bo'sh katlam yaratamiz. Shu qatlamni sichqonchaning chan tugmasi bilan bosib Maska - **Mask** buyrug'ini tanlaymiz. Natijada qatlam ko'k rangga o'zgaradi va ikkala qatlamlar  o'zgarishlardan ximoyalanadi. Maska qatlamdan shu himoyani uchirib, boshkaruv kadrda bir nechta to'rtburchaklar chizamiz. Keyin yana maska qatlamning himoyasini yoqib klaviaturadagi [Enter] tugmasini bosamiz va natijani ko'ramiz. Maska qatlamdagi boshqaruv kadrni **motion tweening** animatsiyalashtirilsa harakatlanish efekti yanada chiroyli bo'ladi.

Flashdagi animatsiya multi - rolikda ishlatiladigan obyektlarning xossalarini o'zgartirishga asoslangan. Masalan, obyektlar yo'qolishi yoki naydo bo'lishi, joylashuvini, ko'rinishini, rangi va boshqalarni o'zgartirishi mumkin. Flashda obyektlarni animatsiyalashni uchta turli usuli ko'zda tutilgan:

-  kadrma-kadr («klassik») animatsiya, bu usulda muallif bo'lgusi «multi - rolik»ni har bir kadrini o'zi yaratadi yoki boshqa joydan qabul qiladi va namoyish ketma-ketligini o'zi o'rnatadi;
-  avtomatik animatsiya (tweened-animatsiya), bu usulda muallif multi – rolik kadrlarini faqat birinchi va oxirgisini yaratadi, oraliq kadrlarni esa Flash

avtomatik tarzda o'zi yaratadi; tweened-animatsiyani ikki xil turi mavjud: obyektни ko'chirishga asoslangan animatsiya (motion animation) va obyektни transformatsiya (ko'rinishini o'zgartirish)siga asoslangan animatsiya (shape animation);

- ✚ senariyga asoslangan animatsiya; senariy - Flashning dasturlash tilida (bu til **Actionscript** deb nomlanadi) obyektни hatti- harakatini berilishidir. Bu tilni sintaksisi Web-xujjatlarda ishlatiladigan boshqa senariy tillariga (masalan, JavaScript va VBScript) o'xshab ketadi.

Ushbu usullarning har birining o'ziga yarasha afzalliklari va kamchiliklari bor. Masalan, **tweened**-animatsiya ikki afzallikga ega: birinchidan, muallif har bir kadrni alohida yaratish zaruratidan qutulgan; ikkinchidan, bunday multi – rolikni namoyish etish uchun Flashga faqat birinchi va so'nggi kadrni saqlash kifoya, bu esa uz navbatida bunday rolikni hajmini kam bo'lishini ta'minlaydi. Shu bilan birga, **tweened**-animatsiyani faqat obyekt xossalari bir tekisda o'zgaradigan sodda roliklar yaratish uchun ishlatish bo'ladi. **Actionscript** sahnalarda obyektни juda ham murakkab bo'lgan hatti- harakatini ifodalashi mumkin. Lekin buning uchun **Actionscript** tilini o'rganish kerak bo'ladi.

Interfaollik

Flash roliklarni namoyishi vaqtida saytga kirgan foydalanuvchi klaviatura yoki sichqonchani ishlatib rolikni turli fragmentlariga o'tishi, obyektlarni ko'chirishi, formaga ma'lumot kiritishi va boshqa ko'pgina amallarni bajarishi mumkin bo'lgan interfaol roliklarni yaratish imkonini beradi. Flash-roliklarni interfaolli **Actionscript** tilidagi buyruqlar ketma-ketligi ko'rinishidagi amallarni berish yo'li bilan amalga oshiriladi. Amal u bilan bog'liq bo'lgan hodisa yuz berganda bajariladi. Bunday hodisalardan masalan, rolikni biror-bir kadriga o'tish yoki foydalanuvchi tomoindan sahifadagi tugmachani bosishi bo'lishi mumkin. Tugmachalar (masalan, alohida boshqaruv paneli ko'rinishida birlashtirilgan), rolik fragmentlari yoki alohida kadrlari interfaol bo'lishi mumkin. Shu bilan birga ko'pgina standart amallar rolikni biror- bir elementiga **Actionscript**da dasturlashdan foydalanmay berilishi mumkin. Standart amallarga masalan, rolikni namoyishini boshlash va to'xtatish, ovozni yoqish yoki o'chirish, belgilangan URLga o'tish va boshqalar. Hammasi bo'lib standart amallarni unga yaqin turi mavjud.

Flash roliklarda tovushni ishlatishni bir necha usullarini taqdim etadi. Siz rolikni syujeti va foydalanuvchi bajargan amaldan qat'iy nazar to'xtamaydigan tovushni yaratishingiz mumkin. Alternativ variant animatsiya va tovushni sinxronlash. Bundan tashqari, Flashda tugmachalarga tovushni ulashingiz mumkin, bu esa tugmachalarni yanada interfaol qiladi. Nihoyat, tovush bilan ishlashni yana bir varianti- **Actionscript** senariylardan foydalanib tovushni boshqarishdir. Flashda tovush qo'shishni ikki xili mavjud: hodisaviy-boshqariladigan (event sound) va oqimli (stream sound). Flash tovush bilan ishlashda foydalanuvchlar uchun quyidagi imkoniyatlarni yaratadi:

- bo'lunuvchan tovush kutubxonalarini yaratish, ya'ni bir kutubxonadagi tovushni bir necha roliklarda foydalanish imkoniyatidir;
- tovush sifati va eksnort qilinayotgan rolikni optimal birlashuvini tanlash maqsadida tovushli ma'lumotlarni qisish ko'rsatkichlarini boshqarish; muallif

alohida tovush yoki rolikdagi barcha tovushlar uchun qisish ko'rsatkichlarini tanlashi mumkin.

Bundan tashqari tovushni o'zgartirish uchun turli qo'shimcha effektlardan foydalanish ham mumkin.

Roliklarni Flash yordamida yaratish va tahrirlash vaqtida foydalanuvchi FLA formatga ega fayllar bilan ishlaydi. Bu Flash muharririni ichki (shaxsiy) formatidir va u faqat Flashga «tushunarli» dir. Web-brouzer yordamida namoyish etish uchun mo'jallangan roliklar **SWF** formatiga o'kazib olinishi zarur. Yuqorida ta'kidlab o'tilganidek, ushbu formatni brouzerlar (Internet Explorer, Netscape va Opera)ni so'ngi versiyalarida namoyish etilishi mumkin yoki maxsus Flash-Player yordamida ham namoyish etilishi mumkin. Ushbu Flash-Player Flash muharriri tarkibiga mustaqil dastur sifatida kiradi, va **SWF**-fayllar Flash muharririsiz ham Flash- Playerda namoyish etilishi mumkin. Shu bilan birga rolikni barcha funksional va interfaol imkoniyatlar saqlanib qoladi. Flash-roliklarni «mustaqil» namoyish etilishi ikkinchi varianti - bu Universal Media- Player (Windows Media Player)dan foydalanishdir. Buning uchun Flash-rolik AVI formatiga eksnort qilinishi kerak. Lekin bu holda, rolikni hajmi o'n ayrim hollarda yuz barobar ortib ketishi mumkin. Flash - roliklarni AVI formatiga o'tkazilishi ularning interfaolligini yo'qolishiga olib keladi, shuning uchun ushbu variant keng tarqalmagandir. Flash-roliklarni namoyish etilishni yana bir varianti mavjuddir (ushbu variant ham inetraktivsiz ko'rinishda bo'ladi). Bu variant Flash-rolikni animatsiyalashgan GIF-formatga o'tkazishdan iboratdir. Bu holda, eksnortni ikki varianti mavjud: har bir fayli animatsiyani 1 kadriga mos keluvchi GIF-fayllar to'plami ko'rinishida yoki bir animatsiyalashgan fayl ko'rinishidadir. Eksnortning ikkinchi varianti afzalrokdir, chunki boshqa bir dastur (masalan, Ulead GIF Animator) yordamida kadrlarni «yig'ish»ga qaraganda bu variantda fayl hajmi kichikroq bo'ladi.

Yuqorida sanab o'tilgan Flash-roliklarni eksnorti ularning dinamikasini (u yoki bu darajada) saqlanishini ta'minlaydi. Lekin, zarur bo'lganda Flashdan oddiy grafik muxarriri sifatida ham foydalanish mumkin. Uning yordamida yaratilgan tasvirni biror bir grafik format (vektorli yoki rastarli)ga o'tkazilish mumkin. Shu bilan birga, Flash rolikga boshqa grafik formatlarda berilgan tasvirlarni qo'shish (ya'ni inmort qilish) imkoniyatiga ega. Quyidagi jadvalda Flash «tushunadigan» keng tarqalgan grafik formatlarni ro'yxati keltirilgan

3- jadval

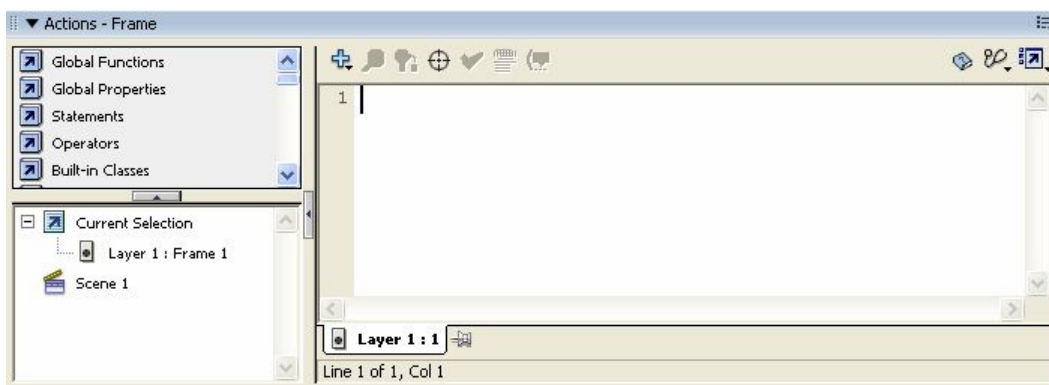
Flash «tushunadigan» grafik formatlarni ro'yxati

Fayl turi	Kengaytma	Eksport	Import
Adobe Illustrator (6.0 va undan yuqori versiyalar)	.eps, .ai	+	+
Bitmap	.bmp	+	+
Enhanced Windows Metafile	.emf	+	+
FreeHand	.ft7, .ft7, .fog, J8, ,fli9, .ft9	+	+
GIF	.gif	+	+
JPEG	.jpg	+	+
PNG	.png	+	+

Fayl turi	Kengaytma	Eksport	Import
Flash Player	.swf	+	+
Windows Metafile	.wmf	+	+
Agar kompyuterga QuickTime 4 yoki undan yuqori versiyasi o'rnatilgan bo'lsa			
PhotoShop	.psd	+	+
QuickTime Image	.qtif	+	+
Silicon Graphics	.sai	+	+
TIFF	.tiff	+	+

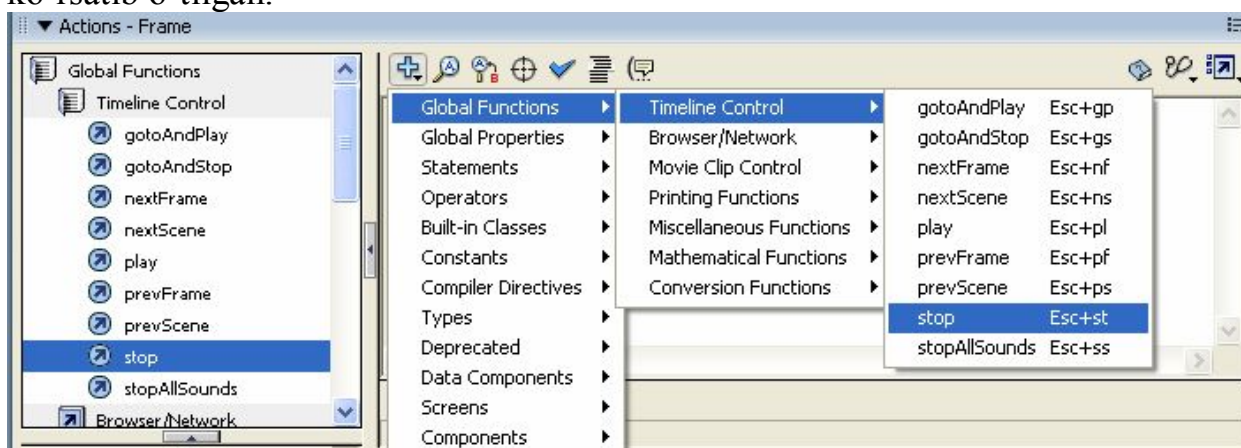
Actionscript dasturlash tilining imkoniyatlari haqida dastlabki ma'lumot va Actions paneli bilan ishlash.

Flash dasturi **Actionscript** dasturlash tiliga asoslangan bo'lib, bu til dasturlashtirishdan yiroq bo'lgan insonlar ham tushunishi uchun juda oddiy qilib yaratilgan. Flash dasturiga **Actionscript**dan foydalanmasdan turib ham oddiy roliklar, chiziqli animatsiyalar yaratish mumkin, lekin tuzilgan dastur samarali, mukammal va oson boshqarish mumkin bo'lishi uchun **Actionscript**dan foydalanishga to'g'ri keladi. Chindan ham interfaol dasturlar yaratish uchun **Actionscript** dasturlash tilini o'rganib olish kerak. **Actionscript** obyektga yo'naltirilgan dasturlash tili hisoblanadi. U sintaksis jihatidan "JavaScript" ga o'xshab ketadi. **Actionscript** boshqa dasturlash tillariga o'xshab o'zining maxsus sintaksisiga, rezervlashgan so'zlariga, operator va ma'lumotlarni o'zida saqlash uchun islatiladigan o'garuvchilariga ega. **Actionscript** o'zining obyekt va funksiyalariga ega va u foydalanuvchiga o'zning obyekt hamda funksiyalarni yaratishga imkon beradi. Flashda skript yozish uchun **Actionscript** tilining barcha imkoniyatlarini bilish shart emas. Oddiy buyruqlardan foydalanib ham scriptlar yaratish mumkin. **Actionscript** tilini o'rganishdan oldin, **Actions** (harakat) paneli bilan tanishib chiqishimiz kerak. Uni ochish uchun sarlavha satri ustiga sichqonchani bir marta chertish kifoya. Agar ekranda **Actions** paneli bo'lmasa [F9] tugmasini bosish orqali ham uni ishga tushirish mumkin. Bitta rolikda juda ko'p rolik bo'lishi mumkin. Dastavval **Actionscript**da roliklarni qayerlarda o'rnatish mumkinligini ko'rib chiqamiz. Roliklarni asosiy kadrlarga, tugmalarga va movie kilplarga joylashtirish mumkin. Grafik obyektlar va boshqalarga roliklarni joylashtirib bo'lmaydi. Hozirda rolikni qaysi obyektga kiritilayotganini bilish juda oson, buning **Actions** panelining sarlavha satridagi yozuviga e'tibor berish kerak. Masalan, tugmani belgilasak, **Actions** panelining sarlavha satridagi yozuvi "**Actions** - button" ga, agar **movie clip**ni tanlasak, "**Actions** - movie clip" ga o'zgaradi, hechnarsa belgilanmaganida "**Actions** - frame" ya'ni joriy kadr belgilangan bo'ladi.



7-rasm. Actions paneli

Kursor joylashgan oq rangli maydon skript kiritiladigan maydon bo'lib, uning yuqori chap qismida buyruqlar oynasi joylashgan. Bu oynadan skriptga joylashtirish mumkin bo'lgan turli xil harakatlar joylashgan. Flashni ishlayotgan foydalanuvchilar uchun oyna bilan ko'p ishlashni tavsiya etamiz. Dastur matnini buyruqlar oynasi va klaviatura orqali to'g'ridan - to'g'ri kiritish mumkin lekin, **Actions**scriptda bosh harflar va kichik harflarni o'rni almasha yoki tinish belgilari noto'g'ri ishlatilsa kiritilgan dastur matnini xato hisoblaniladi. Dasturda ishlash bo'yicha ko'nikma hosil bo'lmaguncha buyruqlarni yozishda shu oynadan masq qilsh maqsadga muvofiq. Buda har bir papkalar ichida faqatgina harakatlar emas, balki turli xil funksiyalar, o'zgarmlar, o'zgaruvchilar, operatorlar, metodlar, xususiyatlar, sikllar, izohlar va boshqalar joylashgan. Biror harakatni joylashtirish uchun papkalardan birini ochamiz, kerakli buyruqni topib, uni ustiga sichqoncha tugmasini bosib, rolik kiritiladigan maydon ustiga qo'yib yuboramiz, yoki shu buyruq ustiga sichqonchani ikki marta chertib qo'ymiz. Agar kerakli buyruq qaysi papkada joylashganligini unitib qo'yan bo'lsangiz, u holda "indeks" papkasidan foydalaning. Bu papkada Flashda ishlatiladigan barcha buyruqlar alfavit bo'yicha tartiblab qo'yilgan. Buyruqlar oynasini yopish uchun oynaning o'ng tomoni markazida joylashgan uchburchakchani bosish kerak. Agar buyruqlar oynasi yopib qo'yilgan bo'lsa, kerakli buyruqlar skript kiritiladigan maydon ustida joylashgan [plus] [+] tugmasi yordamida kiritish mumkin. Bu tugmada asboblarni oynasidagi barcha funksiyalarning nusxasi joylashtirilgan. Quyida "stop()" buyrug'ining ikki xil usulda asboblarni paneli va [plus] tugmasi yordamida joylashtirish ko'rsatib o'tilgan.



8-rasm. Stop() buyrug'ga murojaat

[Plus] tugmasidan keyin joylashgan tugmalarga izoh berib o'tamiz:

 **Find** - kiritilgan skriptdan biror soʻzni topish uchun ishlatiladi. Agar kiritilgan skript juda katta boʻlsa bu tugma qoʻl keladi.

 **Replace** – skriptdagi biror soʻzni boshqa soʻz bilan almashtirish uchun ishlatiladi.

 **Check syntax** – kiritilgan skriptning toʻgʻri yoki notoʻgʻriligini yaʼni sintaktik xatoga yoʻl qoʻyilgan yoki yoʻqligini tekshiradi.

 **Auto format** – skriptlarni yozilish jihatidan tartiblashtiradi va tinish belgilarini qoʻyadi.

 **Reference** – Help panelini ochadi. Debug options – kursor turgan satrga nazorat nuqtasini qoʻyadi, nazorat nuqtasini olib tashlaydi.

 **View options** – yordamida skriptga kiritiladigan satrlarni tartiblash (agar kiritilgan skriptga xatoga yoʻl qoʻyilsa, xato qaysi qatorda ekanligini aniqlash oson boʻladi) va tezkor maʼlumot kiritishga imkon beruvchi skriptlarni buyruqlar oynasida koʻrsatish yoki oʻchirib qoʻyish uchun ishlatiladi.

Actions oynasining chetki oʻng burchagida joylashga tugmada ham birqancha buyruqlar yozilgan ularni aksariyatini koʻrib oʻtdik. Qolganlari bilan tanishib chiqamiz:

1. “Импорт скрипта” – bundan oldin kiritilgan skriptni joylashtiradi.
2. “Экспорт скрипта” – kiritilgan skriptni as formatda kiritib saqlab qoʻyish uchun ishlatiladi. Keyinchalik buni oddiy usulda oʻqib olish mumkin.
3. “Печат” – kiritilgan skriptni chop etadi.
4. “Параметры автоформата” – skriptni toʻgʻirlashda qanday tartibda toʻgʻirlashi koʻrsatib oʻtiladi.

5. “Настройки” – **Actions** skriptni parametrlarini sozlash uchun ishlatiladi. Bu boʻlimdan foydalanib, skriptni qaysi usulda, rangda, shriftda kiritilishini, turli izohlarni koʻrinish yoki koʻrinmasligini va boshqa xususiyatlarni koʻrsatib oʻtish mumkin.

Actionscriptda oʻzgaruvchilar va oʻzgaruvchilarning tipi.

Flashda rolik yaratayotgan vaqtda maʼlumotlarni saqlash va ularni oʻzgartirishga toʻgʻri keladi. Bunda oʻzgaruvchi tushunchasidan foydalaniladi. Oʻzgaruvchi bu – xotiraning nomlangan qismi boʻlib, turli xildagi maʼlumotlarni saqlash uchun ishlatiladi. Endi biror oʻzgaruvchini yaratib koʻraylik (oʻzgaruvchini yaratish bu, uni eʼlon qilish demakdir). Buning uchun buyruqlar oynasidan **statements** papkasini tanlang, undan soʻng **variables** papkasini va bu papkada joylashgan **var** soʻzi ustiga sichqonchani ikki marta chertib qoʻying. Kiritish maydoniga quyidagi yozuv paydo boʻladi:

► var ;

Var soʻzini klaviatura yordamida ham kiritish mumkin, lekin **Actions** skriptni yaxshi oʻzlashtirgunga qadar buyruqlarni **Plus** tugmasi yoki buyruqlar oynasi yordamida kiritish afzal. Nuqta verguldan, oʻzgaruvchining nomini yoziladi. Masalan **x** oʻzgaruvchisini eʼlon qilamiz:

► var x;

Oʻzgaruvchining nomiga birqancha talablar qoʻyilgan. Ular faqat harflar, sonlar va ajratib koʻrsatish belgisi [“_”]dan iborat boʻlishi kerak; probel, difis va tinish belgilarini ishlatish mumkin emas. Oʻzgaruvchining nomi albatta ajratib koʻrsatish belgisi[“_”] yoki harf bilan boshlanishi kerak. Masalan, oʻzgaruvchining nomini “1x” deb yozilsa

xatolik yuz beradi, “x1” deb yozish to`g`ri. Bulardan tashqari o`zgaruvchining nomi sifatida **Actionscript**ning rezervlashgan so`zlarini ishlatish mumkin emas. **Actionscript**da bosh harf bilan nomlangan o`zgaruvchi bilan kichik harf bilan nomlangan o`zgaruvchi 1 ta o`zgaruvchi ya`ni “x” va “X” ayni bir o`zgaruvchilardir. Ma`lumotlarning oddiy tipiga belgilar, sonlar va mantiqiy tiplar kiradi. Bulardan tashqari Flashda yanada murakkab tiplar mavjud. Bular: massivlar, obyektlar, **movie clipl**ar va boshqalar. **Actionscript**da o`zgaruvchilarning tipini e`lon qilish shart emas. Har bir o`zgaruvhining tipini u avtomatik tarzda aniqlaydi. Hozir belgili va sonli tiplarga to`xtalib o`tamiz. Quyida *x* va *y* o`zgaruvchisi e`lon qilinib, *x* o`zgaruvchisiga 5 sonini va *y* o`zgaruvchisiga 5 satri o`zlashtirilgan.

```
► var x=5;
```

```
y=' 5' ;
```

Bu yerda “=” belgisi o`zlashtirish operatori bo`lib hisoblanadi. Ko`rinib turibdiki o`zgaruvchilarning nomidan oldin **var** so`zi ishlatilgan so`ng birinchi o`zgaruvchining nomi, o`zlashtirish belgisi va 5 soni. Agar 5 qo`shtirnoq ichiga olinmasa, u son sifatida, agar qo`shtirnoq ichiga olinsa, belgi sifatida qabul qilinadi. Ikkinchi o`zgaruvchining nomidan oldin **var** so`zini ishlatish talab qilinmaydi.

Trace funksiyasi, rolikga izoh yozish va oddiy arifmetik operatorlar

Trace funksiyasi –ma`lumotlarni ekranga chiqarish uchun ishlatiladi. U rolikdagi xatoliklarni namoyon qiluvchi vosita. **Trace Global functions => miscellaneous functions** papkasida joylashgan. Uni ishlatish oson masalan, **trace()**; yozuvi chiqqandan so`ng qavs ichida biror yozuvni (tirnoq ichida) yoki o`zgaruvchini nomini yozsak, testdan o`tkazgan paytda ekranda **output** (chiqarish) oynasida yozilgan yozuv yoki o`zgaruvchining qiymati chiqadi. Agar chiqariladigan ma`lumotlar birqancha bo`lsa, ular qavs ichiga qo`shish belgisi yordamida birlashtirib chiqariladi.

z o`zgaruvchining qiymatini izoh bilan ekranga chiqarish:

```
► var
```

```
z=15+20;
```

```
trace('z='+z);
```

Ushbu matnni kiritish oynasiga yozganimizdan so`ng [ctrl]+[enter] tugmalari bosiladi. Ekranda **Output** oynasi ko`rinadi va unda *z=35* yozuvi chiqadi.

Skript tushunarli bo`lishi uchun unga izoh yozib turishga to`g`ri keladi. Izoh bu faqat dasturchining o`ziga qulaylik tug`dirishi uchun ishlatilib, u skriptni qayta o`qishda yordam vazifasini bajaradi. Kompyuter skriptni testdan o`tkazish vaqtida bunday izohlarni testlab o`tirmaydi. Izoh yozishning bir qancha turlari bor, agar yoziladigan izoh faqat bir qatordan iborat bo`lsa, unda 2 ta slesh [//] belgisidan foydalanish qulay. Agar yoziladigan izoh bir necha satrdan iborat bo`lsa, har bir satr oldida 2 ta slesh [//] qo`yish kerak, yoki bo`lmasa, izoh boshiga [/*] belgisini qo`yib izoh oxirida ham [\*/] belgilarini qo`yish kerak. Har ikkala holda ham yoziladigan izohning rangi o`zgaradi (**Actionscrip**ni boshqacha sozlab, izohlarning rangini o`zgarmaydigan qilish mumkin). Izohlar yoziladigan skriptning hajmini oshmaydi. Bu usul yordamida yozilgan skriptning ma`lum qismini bajarilmasligini ta`minlash mumkin. Endi quyidagi matnni kiritib ko`raylik:

```
► var x=5;
```

```
y='5';  
z=x+y;
```

Bu yerda x sonli tipga, yoki belgili tipga tegishli ekanligi tushunarli. Xo'sh z qaysi tipga tegishli? Bunday vaziyatda biz bilamizki satr bilan sonni qo'shib bo'lmaydi. **Actionscript** bunda x ning tipini ham satrli deb oladi va satr bilan satrni qo'shganda yana satr hosil qiladi. Demak z ham satrli tipga tegishli bo'lib uning qiymati "55" ga tegishli bo'ladi. Boshqa amallarni ham qarab o'taylik.

Berilgan ikki sonning yig'indisi, ayirmasi, ko'paytmasi va bo'linmasini hisoblab natijani ekranga chiqarish:

```
► var //o'zgaruvchilar e'lon qilish  
x=150; //x o'zgaruvchisiga 150 soni o'zlashtirish  
y=20; //y o'zgaruvchisiga 20 soni o'zlashtirish  
z1=x+y; // z1 o'zgaruvchisi 2 son yig'indisini qabul qiladi  
z2=x-y; // z2 o'zgaruvchisi x va y ning ayirmasini qabul qiladi  
z3=x*y; // z3 o'zgaruvchisi 2 son ko'paytmasini qabul qiladi  
z4=x/y; // z4 o'zgaruvchisi 2 son bo'linmasini qabul qiladi  
trace('z1='+z1); // z1= yozuvidan so'ng z1 qiymatini chiqarish  
trace('z2='+z2);  
trace('z3='+z3);  
trace('z4='+z4);
```

Ba'zi hisoblash ishlarini olib borganimizda o'zgaruvchilarning qiymatini oshirish, kamayish, ko'paytirish yoki bo'lishga to'g'ri keladi. Buning birqancha yo'llari bor ularni ko'rib chiqamiz:

x ning qiymatini 4 taga oshirish talab qilingan bo'lsin. Buni quyidagicha yozamiz:
 $x = x+4$; Bu yerda x ning qiymati 4 taga oshirilib, yana x ga qayta uzatilyapti. Buni yana boshqacha yozish mumkin: $x+=4$; qolgan barcha amallarni ham xuddi shunday yozish mumkin. $x-=4$; $x *=4$; $x /=4$; Agar x ning qiymatini 1 ga oshirish kerak bo'lsa, uni quyidagi ko'rinishda yozish mumkin: $x++$; Buning o'rniga $x=x+1$; yoki $x+=1$; deb yozish ham mumkin. Agar x ni 1 taga kamaytirish kerak bo'lsa, $x--$; deb yozish mumkin.

Satr bu – sonlar, harflar, tinish belgilari va boshqa turli belgilar ketma-ketligidir. Satriy tiplarga misol qarab o'taylik.

```
a='Men `';  
b='universitetda `';  
c='o`qiyman`';
```

bu o'zgaruvchilar ustida arifmetik amallarni bajarib bo'lmaydi. Satrlar ustida eng ko'p bajariladigan amal bu ularni birlashtirish amalidir. Birlashtirish deganda satrlarni ketma-ket yozish tushuniladi. Satrlarni birlashtirish uchun $+$ belgisidan foydalanamiz. Yuqoridagi misolni davom ettiramiz.

Satrlarni birlashtirib ekranga chiqarish:

```
► a='Men `';  
b='universitetda `';  
c='o`qiyman `';  
z=a+b+c;  
d=a+c+b;  
trace(z+' '+d);
```

Bu misolda c o'zgaruvchisiga "o'qiyman" so'zi o'zlashtirilyapti bizga ma'lumki satriy o'zgaruvchilarga qiymat berganda, ularning qiymatlari tirnoq [''] ichiga olib

yoziladi, o`qiyman so`zini yozish uchun ham “o” harfidan keyin tirnoq belgisini qo`yish kerak, bunda satrni o`qishda xatolik yuzbermasligi uchun “o” harfidan keyin 1 tugmasining chap tomonida joylashgan (`) tugmasidan foydalanish maqsadga muvofiq.

Actionscript o`zgaruvchilarning tipini avtomatik aniqlaganligi tufayli script yozayotgan vaqtda o`zgaruvchining tipini aniqlashga to`g`ri kelib qoladi. Buning uchun **typeof()** funksiyasidan foydalanamiz. **Typeof** funksiyasi **operators** => **Miscellaneous operators** bo`limida joylashgan. Yuqoridagi misol uchun ushbu funktsiyani qarab o`taylik:

O`zgaruvchining tipini aniqlash

```
► a='Men ' ;  
b='universitetda ' ;  
c='o`qiyman ' ;  
z=a+b+c ;  
d=a+c+b ;  
trace(typeof(d)) ;
```

Flashda matematik hisoblashlarni olib borish, Math obyektining metodlari va xususiyatlari

Script yozayotgan vaqtda sonlar ustida turli amallar bajarishga to`g`ri keladi. Bu bo`limda biz shu amallarni bajarishga yordam beruvchi metod va funksiyalarni ko`rib o`tamiz. Dastlab **parseFloat** funksiyasini qrab chiqamiz. **parseFloat** – biror o`zgaruvchiga sonlar ketma-ketligidan iborat bo`lgan satrli qiymat berilganda uni songa aylantirish uchun ishlatiladi. **Global functions** => **Mathematical Functions** papkalari ichida joylashgan. **parseFloat** funksiyasi haqiqiy sonli satrni ham songa aylantira oladi.

Math bo`limidagi metodlar bilan tanishib chiqamiz:

abs() – sonning absolyut qiymatini qaytaradi;
acos() – cossinusning qiymatiga qarab uning burchak qiymatini qaytaradi;
asin() – sinusning qiymatiga qarab uning burchak qiymatini qaytaradi;
atan() – tangensning qiymatiga qarab uning burchak qiymatini qaytaradi;
atan2() – sonning arctangens qiymatini qaytaradi
ceil() – kiritilgan sonning eng katta butun songacha yaxlitlab oladi;
cos() – kiritilgan burchak qiymatini cossinusini hisoblaydi;
exp() – e konstantasining berilgan darajasini hisoblaydi;
floor() – kiritilgan sonning eng kichik butun songacha yaxlitlab oladi;
log() – sonning natural logarifmini hisoblaydi, uning asosi e konstantasi bo`lib, bu funktsiyaga faqan musbat son kiritish talab qilinadi;
max() – ikki sondan kattasini tanlaydi;
min() – ikki sondan kichigini aniqlaydi;
pow() – sonning ixtiyoriy darajasini hisoblaydi.
random() – tasodifiy sonni tanlaydi;
round() – matematik qonun qoida bo`yicha sonni yaxlitlab oladi;
sin() – kiritilgan burchak qiymatini sinusini hisoblaydi;
sqrt() – sonning ildizini hisoblaydi;
tan() – kiritilgan burchak qiymatini tangensini hisoblaydi;

Math bo`limining metodlariga misol:

```
► var a = 0.58;
```



```

b = 2; abc = Math.abs(a); trace('abc='+abc);
acos = Math.acos(a); trace('acos='+acos);
asin = Math.asin(a); trace('asin='+asin);
atan = Math.atan(a); trace('atan='+atan);
atan2 = Math.atan2(a, b); trace('atan2='+atan2);
ceil = Math.ceil(a); trace('ceil='+ceil);
cos = Math.cos(a); trace('cos='+cos);
exp = Math.exp(a); trace('exp='+exp);
floor = Math.floor(a); trace('floor='+floor);
log = Math.log(a); trace('log='+log);
max = Math.max(a, b); trace('max='+max);
min = Math.min(a, b); trace('min='+min);
pow = Math.pow(a, b); trace('pow='+pow);
random = Math.random(); trace('random='+random);
round = Math.round(a); trace('round='+round);
sin = Math.sin(a); trace('sin='+sin);
sqrt = Math.sqrt(a); trace('sqrt='+sqrt);
tan = Math.tan(a); trace('tan='+tan);

```

Bu dasturda 18 ta o'zgaruvchi kiritilgan bo'lib har bir o'zgaruvchining nomi unga o'zlashtirilgan funksiya nomida olingan. Har bir o'zgaruvchining qiymati aniqlanganidan so'ng bu qiymat ekranga chiqariladi. a va b o'zgaruvchilarining qiymatini almashtirib har xil natijaga erishish mumkin. Dasturni testdan o'tkazganimizdan so'ng u bizga quyidagi natijani qaytaradi:

Munosabat operatorlari

№	operatorlar	Natija
1	==	teng
2	!=	teng emas
3	>	katta
4	<	kichik
5	>=	katta yoki teng
6	<=	kichik yoki teng
7	%	a/b ning qoldiq qismini qaytaradi

Tarmoqlanish operatori

if (<shart>) { <operator1>} else {<operator2>}

Bu yerda **if** – agar; **else** – aks holda degan ma'nolarni bildiradi. <shart> – mantiqiy ifoda bo'lib u faqat ikkita qiymat “chin” yoki “yolg'on” ni qabul qiladi. <operator1> va <operator2> – bir yoki birnecha buyruqlar ketma-ketligi. Ushbu operatorning ishlash tamoyili quyidagicha: avval shart tekshiriladi, agar u chin qiymat qabul qilsa, <operator1>, aks holda <operator2> bajariladi. Bajariladigan operatorlar bir yoki bir necha bo'lishi mumkin.

Berilgan ikki sondan qaysi biri katta ekanligini aniqlash:

```

► var a=21;
  b=56;
  if (a<b) {trace('2-son katta')}

```

```
else {trace('1-son katta')}
```

Ushbu dasturning ishlash tamoyili quyidagicha: avval **a** o'zgaruvchisiga 21 va **b** o'zgaruvchisiga 56 qiymati o'zlashtiriladi, so'ng shartli o'tish operatori qavs ichidagi shartni tekshirib ko'radi. Agar **a** ning qiymati **b** dan kichik bo'lsa, ekranga “a kichik” degan yozuvni, agar $a > b$ bo'lsa, “a katta” degan yozuvni chiqaradi. Bizning misolimizda $21 < 56$ bo'lganligi tufayli “a kichik” yozuvi paydo bo'ladi.

Shartli o'tish operatori yordamida quyidagi sistemani yechimini qarab o'tamiz:

$$y = \begin{cases} x^5 & \text{agar } x > 0 \\ |x| & \text{agar } x \leq 0 \end{cases}$$

```
► var
x=-10;
if (x>0) {y=Math.pow(x,5)} /* agar x musbat bo'lsa,
hisoblansin */
else {y=Math.abs(x)} /* bo'lsa uning absolyut
qiymati hisoblansin*/
trace(y)//natija chop qilinsin
```

Tarmoqlanish operatorining to'liqsiz ko'rinishi bor bu quyidagicha:

if (<shart>) { <operator1>}

Agar qavs ichidagi mantiqiy amal chin qiymat qaytarsa, operator1 bajariladi aks holda bajarilmasdan boshqaruv if operatoridan keyingi operatorlarga uzatiladi.

Ikkita satr berilgan ularning elementlari birxilligini tekshiruvchi dastur:

```
► var
a='Salom';
b='Salom';
c='satrlar teng emas';
if (a==b) {c='satrlar teng'}
trace(c);
```

dasturda **a** va **b** o'zgaruvchilarning qiymati teng bo'lsa, **c** ning qiymati o'zgartirilib ekranga chiqariladi, agar teng bo'lmasa, **c** ning qiymati o'zgarmaydi va ekranga uning oldingi qiymati chiqariladi. Dasturni testdan o'tkazsak, “satrlar teng” degan yozuv chiqadi.

5-dastur

Munosabat operatorlari

Operatorlar	Natija
&	va
	yoki
&&	tezkor baholovchi va
	tezkor baholovchi yoki
!	inkor

Mantiqiy operatorga misollar:

a, b, c o'zgaruvchilarning eng katta qiymatlisini topish

```
► var
a=20;
b=36;
c=47;
```

```

        if (a>b & a>c) {q=a}; /* agar a b va c dan katta
bo`lsa q ga a ni o`zlashtir */
        if (b>a & b>c) {q=b}; /* agar b b va c dan katta
bo`lsa q ga b ni o`zlashtir */
        if (c>b & c>a) {q=c}; /* agar c c va c dan katta
bo`lsa q ga c ni o`zlashtir */
        trace('eng katta son '+q);

```

Dastur bajarilishida & (va) munosabat operatoridan foydalanildi. Dastur bajarilganda “eng katta son 47” degan yozuv ko`rinadi.

Kiritilgan sonlar haqida ma’lumot beruvchi dastur

```

► var
a=-20;
b=36;
if (a>0 | b>0) {trace('hech bo`lmasa 1 ta son
musbat')}
/* a yoki b musbt bo`lganda bajariladi */
else {trace('birorta son ham musbat emas')};
/* a ham b ham musbt bo`lmaganda bajariladi */
if (a>0 & b>0) {trace('ikkala son ham musbat')};
/* a ham b ham musbt bo`lganda bajariladi */

```

Dastur bajarilganda quyidagi natija ekranga chiqariladi: “hech bo`lmasa 1 ta son musbat”. Agar **a** va **b** o`zgaruvchilarining qiymatini quyidagicha o`zgartirsak:

```

► var
a=20;
b=36;

```

Tanlash operatori SWITCH-CASE

Switch-case operatorining umumiy ko`rinishi quyidagicha:

```

switch (<ifoda>) {
case <1-hol> : <1-operatorlar>
case <2-hol> : <2-operatorlar>
...
case <n-hol> : <n-operatorlar>
default: <operatorlar>
}

```

Ushbu operator quyidagicha ishlaydi: Ifoda 1, 2, ...n - hollar bilan tekshiriladi. Agar uning qiymati ushbu hollardan biriga teng bo`lsa, shu holdan keyin yozilgan operatorlar bajariladi. Agar hech bir holga mos kelmasa, **Deafult** yozuvidan keyin yozilgan operatorlar bajariladi. **Break** – operatori tarmoq ishini tugatish yoki eng kichik siklni ishini tugatish uchun ishlatiladi.

Yanada tushunarli bo`lishi uchun quyidagi misolni ko`rib chiqamiz.

Berilgan raqamni so`z orqali yozib chiqarish:

```

► var
j=7;
switch (j) {
case 0 : trace('nol'); break;
case 1 : trace('bir'); break;

```

```

case 2 : trace('ikki'); break;
case 3 : trace('uch'); break;
case 4 : trace('to`rt'); break;
case 5 : trace('besh'); break;
case 6 : trace('olti'); break;
case 7 : trace('yetti'); break;
case 8 : trace('sakkiz'); break;
case 9 : trace('to`qqiz'); break;
default: trace('raqam kirit!'); }

```

Ushbu dastur quyidagi tartibda ishlaydi. Avval *j* o'zgaruvchisiga 7 raqami o'zlashtiriladi. So'ng, switch operatori yordamida tekshirish bajariladi. *j* ning qiymati 0 ga teng emas, demak 1 bilan tekshiradi, 1 ga ham teng emas, 7 bilan tekshiradi uning qiymati 7 ga teng shunda u ikki nuqtadan keyingi qismni bajaradi. **Trace** funksiyasi yordamida “yeti” so'zini ekranga chiqaradi va **break** funksiyasi yordamida **switch** operatoridan keyingi qismga boshqaruv uzatiladi. Bizning misolimizda switch funksiyasidan keyin hechqanday operator bo'lmaganligi uchun dastur tugatiladi. Agar *j* ning qiymati birortir holga to'g'ri kelmasa, default operatoridan keyingi qism ishga tushadi va ekranga “raqam kirit” degan yozuv chiqadi.

Takrorlash operatorlari

Hozir takrorlash operatorlaridan biri bo'lmish **for** operatori bilan tanishib chiqamiz. Uning umumiy ko'rinishi quyidagicha:

for (<initsializatsiya>; <shart>; <orttirma>)
{ takrorlash tanasi }

Dastavval initsializatsiya ishga tushadi. Bu qism takrorlash jarayonida faqat 1 marta bajariladi. Undan so'ng 1-qadam: shart tekshiriladi. Agar shart bajarilsa, ya'ni u chin qiymat qaytarsa, takrorlash tanasi bajariladi va undan keyin orttirma bajariladi. 2-qadam: shart tekshiriladi, agar u chin qiymat qaytarsa takrorlanish tanasi va ortirma bajariladi, aks holda, boshqaruv keyingi operatorga uzatiladi. Takrorlash toki shart bajarilmagunga, ya'ni yolg'on qiymat qabul qilgunga qadar davom etadi. Takrorlash operatoriga misol sifatida 1 dan 100 gacha bo'lgan sonlar yig'indisini hisoblovchi dasturni bir necha ko'rinishini qarab chiqamiz:

```
for (k=1, s=0; k<=100; s+=k++) { } trace(s);
```

Bunda initsializatsiya qismida *k* ga 1 va *s* ga 0 o'zlashtirilyapti, shart sifatida *k*<=100 berilyapti. Bu degani *k* ning qiymati 101 bo'lganda takrorlanish tugaydi, va oxirida *s* o'zgaruvchisiga *k* ning qiymati avval o'zlashtirilib, so'ng *k* ning qiymati 1 taga oshirilyapti. Oxirgi qismini quyidagi ko'rinishda yozish ham mumkin: *s*=*s*+*k*, *k*=*k*+1 takror tanasi sifatida hech narsa ko'rsatilmagan, chunki hamma hisoblash for ning ichida bajarilyapti. Agar ochiluvchi va yopiluvchi qavslarni {} qo'ymasak *s* ning har bir qiymati ekranga chiqariladi.

```

s=0; k=1;
for (;k<=100;k++) s+=k
trace(s);

```

Dasturning bu ko'rinishida *s* va *k* ning boshlang'ich qiymati takrorlanishdan oldin e'lon qilinib, takrorlanishning initsializatsiya qismi bo'sh qoldirilgan. Shart xuddi

oldingi misoldagidek qolgan va orttirma qismida k ning qiymati 1 taga oshirilyapti va takrorlanish tanasida s ning qiymatini k taga oshirilyapti. Dastur bajarilganda shart tekshiriladi agar chin qiymat qaytarilsa, avval takrorlanish tanasi bajariladi, ya'ni s ning qiymati oshiriladi va keyin orttirma bajariladi. Takrorlanishning oxirgi qadamida s ga 100 qo'shiladi, k 101 ni qabul qiladi, shart tekshiriladi $101 < 100$ bo'lganligi tufayli takrorlash tugatilib, s ning qiymati ekranga chiqariladi.

```
s=0; k=1;
for (;;) {
  if (k>100) break
  s+=k++; }
trace(s);
```

Bu dasturda ham boshlang'ich parametrlar takrorlanishdan oldin kiritilgan. Takrorlanishning parametrlari esa bo'sh deb belgilangan. Agar takrorlanish tanasiga uni to'xtatuvchi hechqanday operator yozilmasa u cheksiz davom etadi. Shuning uchun shart hamda takrorlanishni to'xtatuvchi operatorni takrorlanishni tanasiga yozdik. Bunda avval k ning 100 dan katta emasligi tekshiriladi, chin qiymat qaytarilsa, s ning qiymati k taga oshiriladi, va k ning qiymati 1 taga oshiriladi.

While operatori

While operatorining umumiy ko'rinishi quyidagicha:

While (<shart>) { <operatorlar> }

While operatori quyidagi tartibda ishlaydi. Dastlab shart tekshiriladi. Agar mantiqiy ifoda chin qiymat qaytarsa, operatorlar ketma-ketligi bajariladi, yana shart tekshiriladi, agar chin qiymat qaytarilsa, yana operatorlar ketma-ketligi bajariladi. Mantiqiy ifoda yolg'on qiymat qaytarganda operatorlar ketma-ketligi bajarilmaydi va boshqaruv takrorlashdan keyingi qismga yuboriladi.

$n!$ ni hisoblovchi dastir tuzing.

```
► var
n=6;           //berilgan son
i=1;           //sanagich
s=1;           //ko'paytma
while (i<=n) { /* agar hali i n dan kichik bo'lsa
takrorlanish takrorlansin*/
  s*=i++ /* s ga i ko'paytirilsin va i ning qiymati 1
taga oshirilsin */ }
trace(s); //natijani chop etilsin
```

Dastur ishga tushgach output oynasida 720 soni chiqariladi. Dasturdagi n ning qiymatini o'zgartirib turli sonlarning faktorialini hisoblash mumkin.

Do while operatori

do while operatori quyidagi ko'rinishga ega:

```
do {
  <operatorlar>
```

} while (<shart>)

Uning ishlash tamoyili **while** operatoriga o'xshash faqat farqi shundaki, **while** da avval shart tekshiriladi va agar u bajarilsa, operatorlar bajariladi. do **while**da undan farqli ravishda avval operatorlar bajariladi, keyin shart tekshiriladi. Agar shart chin qiymat qaytarsa, takrorlanish takrorlanadi va yolg'on qiymat qaytarsa, takrorlanish o'z ishini tugatadi. Misollar asosida uni ishlash tamoyili bilan yaqindan tanishib chiqamiz.

m va **n** berilgan sonlar orasidagi juft sonlarning yig'indisini hisoblovchi dastur:

```
► var
m=4;    //birinchi son
n=10;   //ikkinchi son
s=0;    //yig'indi
do {
    if (m %2==0) {s+=m;} /*agar m ni 2 ga bo'lganda qoldiq
0 (m juft) bo'lsa s ga m ni qo'sh */
    m++; // m ning qiymatini 1 taga oshir
} while (m<=n); /* agar hali m<=n bo'lsa takrorlanish
takrorlansin*/
trace('berilgan sonlar orasidagi juft sonlar
yig'indisi '+s+'ga teng'); /*natiija ekranga chiqarilsin*/
```

Agar dasturda **n** ning qiymati **n** dan katta bo'lib qolsa, dasturda xatolik yuz beradi. Chunki masalan, **m**=40, **n**=10 bo'lsa **do while** operatorida avval operatorlar bajariladi, ya'ni avval 40 ni 10 ga bo'linadi, qoldiq nolga teng bo'lganligi tufayli, **s**=40 qiymatni qabul qiladi. So'ngra $m \leq n$ shart tekshiriladi, shart bajarilmaganligi tufayli, takrorlanish tugatiladi va ekranga "berilgan sonlar orasidagi juft sonlar yig'indisi 40ga teng" degan noto'g'ri javob qaytariladi. Bunday kamchilikni yo'qotish uchun dasturga qo'shimcha shart kiritamiz. **m** va **n** berilgan sonlar orasidagi juft sonlarning yig'indisini hisoblovchi dasturning to'la ko'rinishi(yuqoridagi dastur izohlarsiz o'zgartirib keltirilgan):

```
m = 40;
n = 10;
s = 0;
if (m<n) {
do {if (m%2 == 0) { s += m;}
m++; } while (m<=n);
trace('berilgan sonlar orasidagi juft sonlar
yig'indisi '+s+'ga teng');
}
else { trace('1-son 2-sondan kichik bo'lishi kerak!')}
```

Bunda dastur ishga tushgach ekranga "1-son 2-sondan kichik bo'lishi kerak!" degan yozuv namoyon bo'ladi. ▲

Tugma holatlarini qayta ishlash

Stop va **Play** funksiyalar **Actionscript**da eng oddiy funksiyalar bo'lib, ular **Global functions => Timeline control** bo'limida joylashgan.

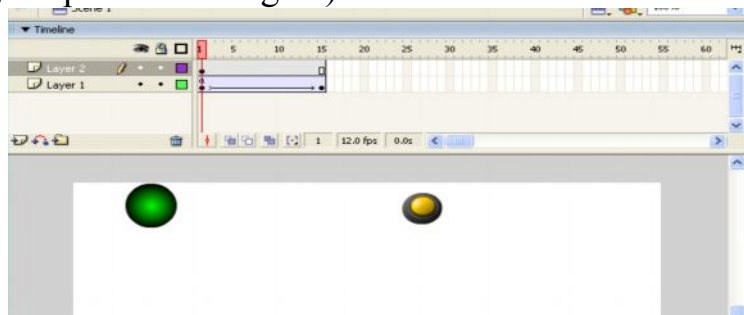
stop() – global funksiyasi, ma'lum rolikning yoki klipning namoyish etishdan to'xtatadi va foydalanuvchining harakatini kerakli kadrda kutib turadi. Uni klip yoki rolikni takrorlanishik ravishda qaytarilmasligi uchun ham ishlatiladi.

play() – global funksiyasi, ma’lum rolikning yoki klipning namoyish etish uchun ishlatiladi.

Flashda yozilgan rolik flash rolikni interfaol qilib, foydalanuvchining qandaydir faoliyatiga javob qaytarishiga imkon beradi. Ammo **Actions**scriptda yozilgan skript o’z-o’zidan bajarilmaydi, buning uchun qandaydir harakat yuz berishi kerak. Bu sichqonchanning tugma ustidagi harakati, klaviaturadagi qaysidir tugmani bosish, rolikning qaysidir kadrini namoyishi va boshqalar.

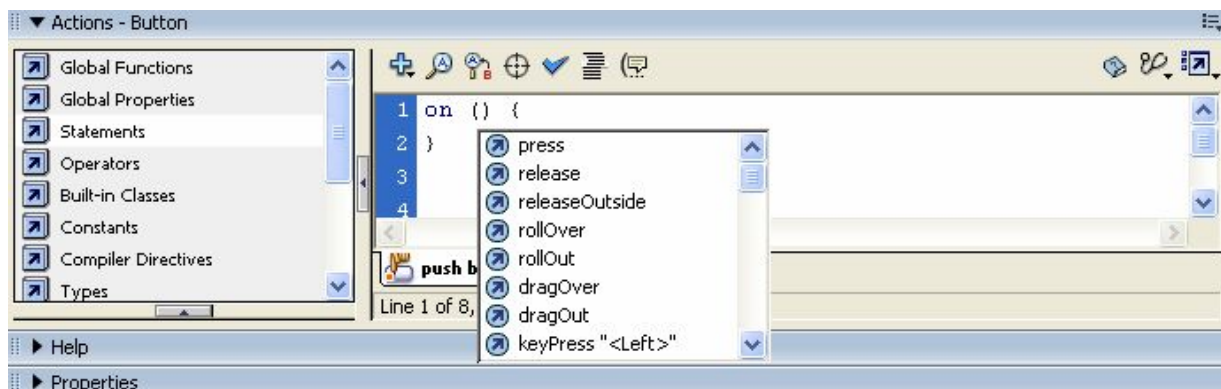
Har xil holatlarni boshqarish uchun Flashda maxsus funksiyalar yaratilgan. Ular holatlarni qayta ishlash deb nomlangan. O’z-o’zidan qandaydir holat biror harakatni bajarmaydi. Buning uchun qandaydir holat bajarilganda amalgam oshishi lozim bo’lgan skriptni kiritilishi kerak. Holatlarni qayta ishlash tygmalarga, kliplarga va ba’zi boshqa obyektlarga qo’yiladi. Bu bo’limda sichqonchanning tugma ustidagi turli faoliyati va klaviatura tugmalarini bosganda sodir bo’ladigan holarlar bilan shug’ullanamiz.

Tugma ustida sichqoncha tugmasi bosilganda, **movie clip**ning harakatlanishi: Dastlab oynada 1 ta **movie clip** yaratib qo’yamiz. Misol uchun 1 ta shar rasmini chizib, uni tepadan pastga harakatlanadigan qilib sozlaymiz va namoyish vaqtida bu shar har gal o’zidan-o’zi tepadan pastga to’xtovsiz harakatlanib turmasligi uchun uning 1-kadriga **stop()**; funksiyasini yozib qo’yamiz. Shundan so’ng, yana bitta qatlam yaratib, [F11] tugmasini bosamiz va flash kutubxonasidan oldindan tayyorlab qo’yilgan 1 ta button (tugma) ni quyidagin tarzda oynaga joylashtiramiz(o’ng tomonda harakatlanuvchi shar, chap tomonda tugma):



9-rasm. Tugma holatlari bilan ishlash

Endi tugmani belgilaymiz va **Actions**, oynasini ochib, rolikni harakatlantiruvchi skriptni kiritamiz. Tugma foydalanuvchining chaqiruviga javob qaytarishi uchun, unga dastlab, holatni qayta ishlovchi buyruqni kiritib qo’yishimiz kerak. Shundan so’ng shu buyruq ichiga belgilangan tugma nima vazifani bajarishi yozib qo’yilishi kerak. Tugmaga holatni qayta ishlovchi buyruqni yozish uchun, **on** operatoridan foydalanish kerak. Ushbu operator, **Global funktions => Movie clip Control** papkalari ichida joylashgan. Shu operator ustiga sichqonchani chertib qo’ysak, ekranga quyidagi yozuvlar ko’rinadi:



10- rasm. on() buyrug'ga murojaat

on() operatoridan keyin ochiluvchi ro'yxat ko'rinadi Biz bu ro'yxatdan **release** buyrug'ini tanlaymiz. release shu tugma ustida sichqoncha tugmasi bir marta bosib qo'yib yuborilgan holatni ifodalaydi. Agar boshqa holatni tanlamoqchi bo'lsa, shu holatni ifodalovchi so'z ustiga sichqoncha tugmasini bosamiz. on() operatorisiz birorta tugma ishlamaydi, shuning uchun uni yaxshilab eslab qolishga harakat qiling. Holatni qayta ishlovchi buyruqni yozganimiz bizga hechqanday natija bermaydi. Biz bu buyruq ichiga tugma bajarishi kerak bo'lgan skriptni kiritishimiz kerak, bizning holda tugma ustida sichqonchani bosib qo'yib yuborganda, yaratilgan rolik namoyish etilishi kerak, ya'ni to'p tepadan pastga harakatlanib yana o'z joyiga kelib to'xtashi kerak.. Buning uchun figurali qavslar ichiga `play();` deb yozamiz. Bu funksiya 1-kadrda to'xtab qolgan rolikni namoyish qiladi va yana 1-kadrga kelganda, rolik to'xtab qoladi. Xullas tugmaga quyidagi skript kiritildi:

```
on (release) {
    play();
}
```

Endi tugma ustiga bajariladigan holatlarga to'xtalib o'tamiz. Ular quyidagilar:

on (release) – tugmani bosib qo'yib yuborilgandagi holat.

on (press) – tugma ustida sichqoncha tugmasini bosgandagi holat.

on (rollOver) – tugma ustida sichqoncha ko'rsatgichi kelgandagi holat

on (rollOut) – sichqoncha ko'rsatgichi tugma chegarasidan chiqqandagi holat.

on (releaseOutside) – sichqoncha tugmasi berilgan tugma ustida bosib, uni tugma chegarasidan tashqarida qo'yib yuborilgandagi holat.

on (dragOut) – sichqoncha tugmasi berilgan tugma ustida bosib, uni tugma chegaradan tashqariga chiqarilgandagi holat.

on (dragOver) – sichqoncha tugmasi berilgan tugma ustida bosib, uni tugma chegaradan tashqariga chiqarib so'ng yana tugma ustiga keltirilgandagi holat.

on (keyPress "<Space>") – klaviaturadagi probel tugmasi bosilgandagi holat.

Bulardan tashqari `keyPress "<Left>"`, `keyPress "<Right>"`, `keyPress "<Enter>"`,... kabi holatlar ham mavjud bo'lib, ular mos ravishda nomi yozilgan tugmalar bosilganda bajariladigan holatlarni ifodalaydi. Shuni ta'kidlab o'tishimiz kerakki, bir tugma uchun bir necha holni baravariga tanlashimiz mumkin. Buning uchun turli xil hollarning nomi vergul bilan yozishimiz kerak. Masalan, yuqoridagi misol uchun tugmaga quyidagi skriptni kiritamiz:

```
on (release, keyPress "<Left>") {
    play();
}
```


}

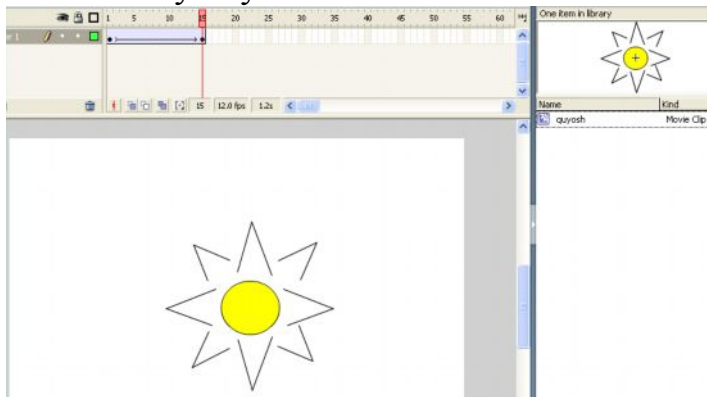
Bunda dasturni ishga tushirsak, 1 ta tugma va to'p ko'rinib turadi. Agar klaviaturadagi chap yo'nalish tugmasini bossak, to'p tepadan pastga harakatlanib, qaytib tepaga to'xtab turadi. Tugma ustiga sichqoncha tugmasini bosganimizda ham shu hol ro'y beradi.

O'tish operatori va kadrlarni belgilash

Hozirgacha tanishib o'tgan funksiyalarimiz parametrsiz bo'lib, ular eng oddiy funksiyalar hisoblanadi. Bu bo'limda **goto** funksiyasi bilan tanishib o'tamiz. Bu funksiya Flashda ko'p ishlatilib, rolik yoki klipning ixtiyoriy kadriga o'tib, shu kadrdan boshlab namoyishni davom ettirish, yoki biror kadrga o'tib, namoyishni to'xtatib qo'yish uchun ishlatiladi. Chiziqli ko'rinishga ega bo'lgan rolikning namoyish tartibini o'zgartirish uchun, o'tish operatorini rolikning biror kadriga, tugmaga yoki klipga joylashtirish mumkin. Dastlab goto funksiyasini kadrga ishlashini quyidagi misolda ko'rib chiqamiz.

Ekranda kichikina quyosh rasmi joylashtirilsin, dastur ishga tushgach, quyosh kattalashsin va to'xtovsiz o'zining atrofida aylanib tursin.

Buning uchun quyosh rasmi chizib qo'yilgan tayyor **movie clip** dan foydalanamiz. [F11] tugmasini bosib, Flash kutubxonasini ishga tushuramiz. Undan quyosh rasmini olib, oynaga qo'yamiz. **Timeline** oynasining 15-kadriga asosiy kadrni o'rnatamiz. 1-dan 15- kadrgacha bo'lgan masofani belgilab olib, sichqonchani o'ng tugmasini bosamiz va **“Create Motion Tween”** buyrug'ini tanlaymiz. Oxirgi asosiy kadrqa quyoshning kattalashgan tasvirini yasaymiz.



11-rasm. Quyoshning harakatlanishi.

Dasturni ishga tushirsak, quyosh to'xtovsiz ravishda kattalashadi. Keyingi qadam 35-kadrga asosiy kadrni o'rnatib, unda quyosh o'zining atrofida aylangan tasvirini hosil qilamiz. Endi dasturni ishga tushirsak, quyosh kattalashib keladi va o'z o'qi atrofida 1 marta aylanib yana kattalashadi va yana o'z o'qi atrofida aylanadi bu to'xtovsiz takrorlanadi. Bizga quyoshning faqat bir marta kattalashishi va o'zining atrofida aylanib turishi kerak. Buning uchun 35-kadrga **“15-kadrga o't”** buyrug'ini yozish kifoya, chunki quyoshning o'z o'qi atrofida aylanishi 15-kadrdan boshlanadi. Buning uchun quyidagi skriptni kiritamiz:

gotoAndPlay(15) ;

Yuqoridagi funksiya **Global funktions => Timeline control** papka ichida joylashgan. Ushbu funksiya **stop()** va **play()** funksiyalaridan farqli ravishda parametrga

ega bo'lgan funksiyadir. U **goto** funksiyasining bir ko'rinishi bo'lib, uning 2- ko'rinishi **gotoAndStop()**; Ularning farqi quyidagicha:

gotoAndPlay(); – ixtiyoriy kadrda o'tib, shu kadrda boshlab namoyishni davom ettiradi.

gotoAndStop(); – ixtiyoriy kadrda o'tib, shu kadrda namoyishni to'xtatib turadi.

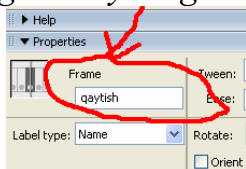
Ushbu funksiyalarni rolikning yoki klipning ixtiyoriy kadriga va sahnasiga qo'yish mumkin. Bizga ma'lumki rolik bir necha sahnadan iborat bo'lishi mumkin. Avtomatik tarzda **gotoAndPlay** va **gotoAndStop** funksiyalari joriy sahna uchun sozlangan bo'ladi. Agar biz boshqaruvni joriy sahnaga emas, biror boshqa sahnaga uzatmoqchi bo'lsak, funksiyalar quyidagicha yoziladi:

gotoAndStop("<sahna nomi>",<kadr nomeri>);

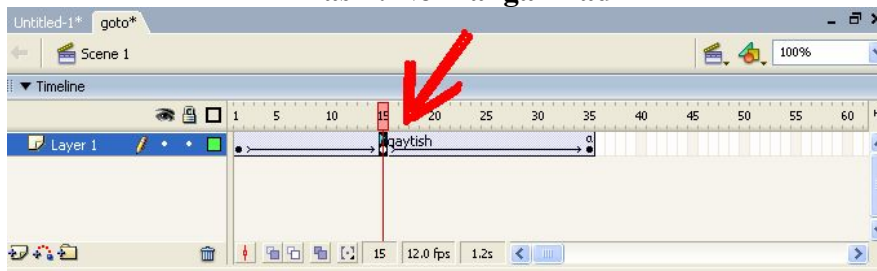
gotoAndPlay("<sahna nomi>",<kadr nomeri>);

Agar dasturga o'zgartirish kiritmoqchi bo'lsak. Masalan quyoshning kattalashishiga ketadigan vaqtni ko'paytirmoqchi yoki kamaytirmoqchi bo'lsak, unda asosiy kadr joyidan chapga yoki o'ngga siljiydi, demak o'rtada joylashgan asosiy skriptning ham nomeri o'zgaradi. Bundan kelib chiqdiki **gotoAndPlay(15)** o'rniga boshqa hosil bo'lgan yangiskriptni yozishimiz kerak. Katta hajmli dasturlar bilan ishlaganda bu ancha qiyinchilik tug'dirishi mumkin. Bunday noqulayliklarni yo'qotish maqsadida Flashda birqancha qulayliklar mavjud. Shulardan biri kadrlarni metkalash. Bu metkalar kadr bilan birga harakatlanadi va agar kadr nomeri o'rniga metkadan foydalansak, yuqoridagi hol uchun skriptlarni almashtirib o'tirishga hojat qolmaydi. Kadrda metka qo'yish uchun: Kerakli asosiy kadrni belgilab olamiz, "properties" panelidagi **frame** yozuvi ostida kadrda kerakli nom (metka) beramiz. Dastur bu nomni satr deb olganligi tufayli, nomlashda probel va tinish belgilari shuningdek kiril yozuvidan ham foydalanish mumkin. Biz 12-kadrni qaytish deb nomladik.

Nomlangan kadr yonidan uning nomi yozilganligiga e'tibor bering.



12-rasm. Nomlangan kadr



13-rasm. Nomlangan kadrda murojat

endi

```
gotoAndPlay(15);
```

skriptni o'rniga o'rniga

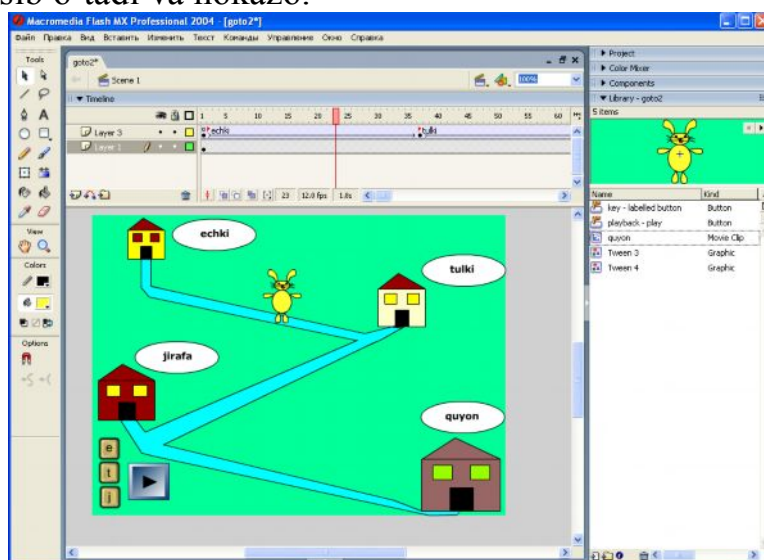
```
gotoAndPlay("qaytish");
```

ni kiritish kerak. Endi nomlangan asosiy kadrni o'ngga yoki chapga siljitsak ham, skriptni o'zgartirishga hojat qolmaydi.

Ba'zi hollarda goto funksiyasini tugmaga joylashtirishga to'g'ri keladi. Buni misol asosida qarab o'tamiz.

Quyunchani boshqa hayvonlar uyidan uyigacha eltib qo'yish

Bizga rolik oldindan tayyorlab qo'yilgan, oynada 1-qatlamda 4 ta hayvonning uyi (har bir uy yonida u qaysi hayvonga tegishliligi text komponenti yordamida yozib qo'yilgan) ba 4 ta tugma, 2-qatlamda quyoning rasmi joylashtirilgan. Quyoning rasmi **movie clip** bo'lib, u har doim oyog'ini qimirlatib xuddi yurayotgandek harakatlanadi. Dasturda 4 ta asosiy kadr o'rnatilgan bo'lib, ular "Echki", "Tulki", "Jirafa", "Quyion" deb nomlangan. "Echki" asosiy kadrigan "Tulki" asosiy kadrigacha bo'lgan orliqda quyion echkining uyidan tulkining uyigacha harakatlanadi. "Tulki" asosiy kadrigan "Jirafa" asosiy kadrigacha bo'lgan oraliqda quyion tulkining uyidan jirafaning uyigacha bo'lgan masofani bosib o'tadi va hokazo.



14-rasm. Tugma orqali obyektни boshqarish.

Bizning vazifamiz, [e] yozuvli tugmani bosganimizda quyion echkining uyi oldiga, [t] yozuvli tugmani bosganimizda, tulkini va [j] yozuvli tugmani bosganimizda quyion jirafaning uyi oldiga kelib to'xtasin. ► tugmasini bosganimizda quyion o'zi turgan uy yonidan so'qmoq bo'ylab o'zining uyiga borsin. Dasturni shu turishida testdan o'tkazsak, quyion to'xtovsiz ravishda echkining uyidan o'zining uyiga tomon harakatlanadi. Dastlab birinchi va oxirgi kadrlarga `stop()` buyrug'ini yozib qo'yamiz. Chunki quyion dastur ishga tushganida 1-kadrdan, echkining uyi oldida to'xtab turishi va oxirgi kadr ya'ni o'zining uyi oldiga borganida ham to'xtab turishi kerak. ► tugmasida quyidagi skriptni kiritamiz:

```
on (release) {  
    play();  
}
```

Play buyrug'i bilan tanishgan bizlar. U rolikni to'xtagan kadridan boshlab namoyish etadi.

Quyidagi tugmalarga quyidagi skriptlarni yozib chiqamiz:

e tugmasiga:

```
on (release) {  
    gotoAndStop("echki");  
}
```

t tugmasiga

```
on (release) {
    gotoAndStop("tulki"); }
```

 tugmasiga

```
on (release) {
    gotoAndStop("jirafa"); }
```

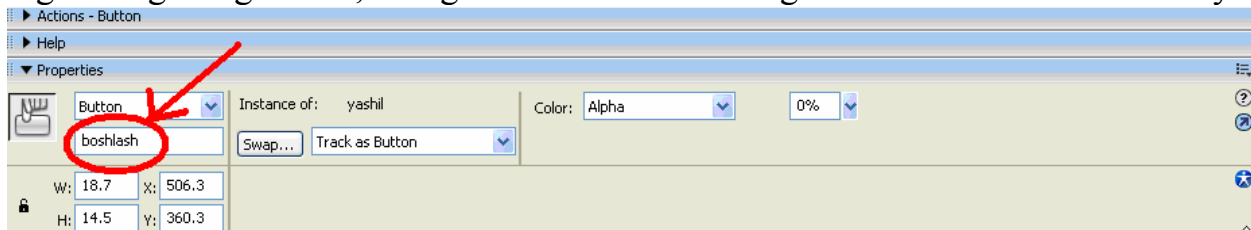
Dasturni ishga tushirib ko`rsak, o`ylangan g`oya amalgam oshadi.

Bizlar yuqoridagi skriptlarni har bir tugmaga kiritib o`tirmasdan tugmalar joylashgan kadrda yozishimiz mumkin. Buning uchun quyidagi funksiya ishlatiladi:

```
<tugma nomi>.onRelease = function() {
    play();
};
```

Bunda avval har bir tugmani nomlashimizga to`g`ri keladi. Tugmalar quyidagi tarzda nomlanadi:

Avval tugmani belgilab olamiz va **properties** panelida **button** yozuvining ostidagi qatorda tugmani nomini kiritamiz. Yuqoridagi misol uchun  tugmadagi skriptni tugmaning o`ziga emas, kadrda kiritib ko`ramiz. Tugmani “boshlash” deb nomlaymiz.



15-rasm. Tugmani nomlash.

1-kadrni belgilaymiz va unda quyidagi skriptni kiritamiz:

```
boshlash.onRelease = function() {
    play(); };
```

Skriptlarni kadrda yozishning afzalligi shundaki, dasturni qayta o`qishda qiyinchilik tug`ilmaydi va skriptlarning hammasi bir joyda saranjom turadi.

Roliklar va videokliplar xususiyati

Har bir rolik, videokliplar va boshqa Flashning obyektlari o`zining xarakterini ifodalovchi xususiyatlariga ega. Biz videoklipning xususiyatlari bilan tanishamiz. Ularni ba`zilarini quyida keltirilgan:

_alpha – klipning shaffoflik darajasi. Agar u 0 qiymar qabul qilsa klip butunlay ko`rinmaydi, agar 100 qiymatni qabul qilsa, 100% ko`rinadi.

_currentframe – joriy kadr raqami.

_drogtarget – _ belgilangan flash simvolning nusxasining nomi.

_framesloaded – yuklangan kadrlar soni.

_height – klipning balandligi.

_name – klip ekzimplifyari nomi.

_qualitu – joriy namoyishning sifati

_parent – boshlang`ich klipga murojaat.

_rotation – qaytish burchagi (graduslarda).

_target – eski notatsiyadagi klipga to`la yo`l.

_totalframes – vaqtinchalik diagrammadagi kadrlar soni.

_url – yuklangan rolikning tarmoqdagi adresi.

_visible – klipning ko`rinish darajasi, agar u 0 qiymat qabul qilsa, klip ko`rinmaydi, agar 1 qiymat qabul qilsa, ko`rinadi.

_width – klipning kengligi (piksellarda).

_x – klipning gorizont bo`yicha koordinatasi.

_xmouse – sichqoncha ko`rsatgichining gorizont bo`yicha koordinatasi.

_xscale – gorizont bo`yicha masshtabi (% da)

_y – klipning vertikal koordinatasi.

_ymouse – sichqoncha ko`rsatgichining vertikal koordinatasi.

_yscale – vertikal masshtabi (% da).

Endi misol asosida ularni chuqurroq o`rganishga harakat qilamiz.

Movie clipni xususiyatlarini o`zgartiring. Tugma yordamida: uni o`ngga, chapga, yuqoriga va pastga harakatlantiring; shaffofligini, masshtabini, burilish burchagini o`zgartiring. **Movie clip**ni ko`rinishi yoki ko`rinmasligini boshqaruvchi tugmalarni o`rnating.

Ekranda 13 ta tugmabni izohi bilan joylashtiramiz. Ikkinchi qatlamda ayiq rasmi tasvirlangan **movie clip**ni kutubxonadan olib, ekranga **x** va **y** o`qlari bo`yicha 200 pikselda joylashtiramiz va uni ayiq deb nomlaymiz. Dastur quyidagi ko`rinishga ega bo`ladi:



16-rasm. Movie clipni tugmlar orqali boshqarish.

Endi tugmalarga kerakli skriptlarni kiritish bilan shugullanamiz. Ustiga “Harakatlantirish” deb yozilgan tugmalarga quyidagi skriptlarni kiritamiz:

⬅ tugmasi ayiq nomli obyektни gorizont 10 pikselga chapga siljitadi:

```
on (release) {  
    ayiq._x -= 10;}
```

➡ tugmasi ayiq nomli obyektни gorizont 10 pikselga o`ngga siljitadi

```
on (release) {  
    ayiq._x +=10;}
```

⬆ tugmasi ayiq nomli obyektни vertikal 10 pikselga yuqoriga siljitadi

```
on (release) {  
  ayiq._y -=10;}
```

⦿ tugmasi ayiq nomli obyektни vertikal 10 pikselga pastga siljitadi

```
on (release) {  
  ayiq._y +=10;}
```

Ustiga “masshtab va burilish” deb izohlangan tugmalarga quyidagi skriptlar kiritilgan:

◀ **movie clipni 10 pikselga kichiklashtiradi:**

```
on (release){  
  ayiq._xscale -=10;//vertical bo`yicha  
  ayiq._yscale -=10;//gorizontal bo`yicha}
```

▶ **movie clipni 10 pikselga kattalashtiradi:**

```
on (release){  
  ayiq._xscale +=10;  
  ayiq._yscale +=10;}
```

↺ **movie clipni 10° ga soat strelkasiga qarama-qarshi tomonga buradi:**

```
on (release) {  
  ayiq._rotation -=10;}
```

↻ **movie clipni 10° ga soat strelkasi bo`ylab buradi:**

```
on (release) {  
  ayiq._rotation +=10;}
```

“Shaffoflik va ko`rinish” deb izohlangan tugmalar:

⦿ tugmasi ayiq nomli obyektни ko`rinmas holga keltiradi

```
on (release){  
  ayiq._visible =0;}
```

⦿ tugmasi ayiq nomli obyektни ko`rinadigan qilib sozlaydi

```
on (release){  
  ayiq._visible =1;}
```

⦿ tugmasi ayiq nomli obyektни shaffofligini oshiradi

```
on (release){  
  ayiq._alpha +=20;}
```

⦿ tugmasi ayiq nomli obyektни shaffofligini kamaytiradi

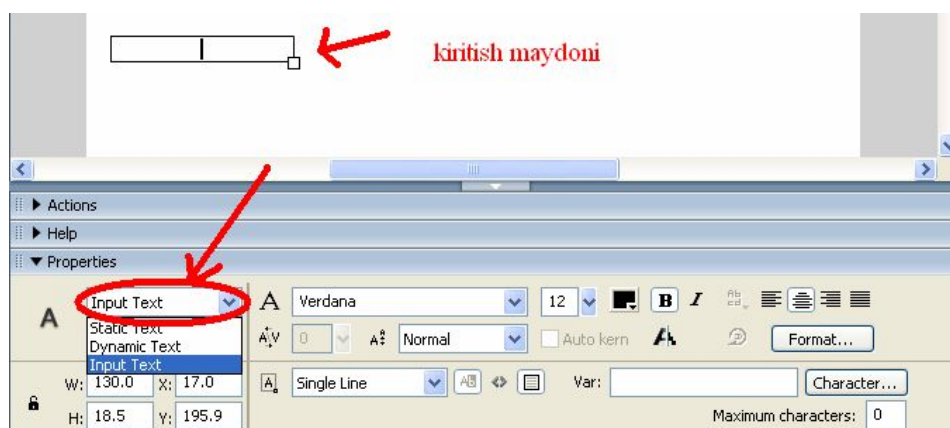
```
on (release){  
  ayiq._alpha -=20;}
```

🔵 **Ayiqni boshlang`ich vaziyatiga qaytaradi:**

```
on (release) {  
  ayiq._x=200;  
  ayiq._y=200;  
  ayiq._xscale=100;  
  ayiq._yscale=100;  
  ayiq._rotation=0;  
  ayiq._alpha=100;  
  ayiq._visible=1; }
```


Kiritish madonlarini hosil qilish. if, if...else shartli operatorlarini rolikda qo'llanilishi

Dasturda turli xil izohlar yozish qiyin emas. Buning uchun “tools” panelidagi “text tool” asbobidan foydalanamiz. Ammo bizlarga ba’zida ma’lumotni kiritishga to’g’ri kelib qolishi mumkin. Buning uchun kiritish maydoni hosil qilishimiz kerak. Kiritish maydonini hosil qilishda “text tool” asbobidan foydalanib kiritish maydonini uzunligini chizib qo’yamiz. So’ng “properties” panelidan ochiluvchi ro’yxatdan “input text”ni tanlaymiz va bizda ma’lumotlarni qabul qiluvchi kiritish maydoni hosil bo’ladi.



17-rasm. Kiritish maydonini hosil qilish.

“Properties” panelidan kiritish maydonining shrifti, o’lchami, yozuv rangi, chegarasining ko’rinish ko’rinmasligini sozlab olishimiz va nomlashimiz mumkin.

Biz shu vaqtgacha ko’rib o’tgan misollarimiz chiziqli algoritmlar bo’lib, ular qadamma-qadam 1-sidan boshlab oxirigacha barcha qadamlari bajariladi. Shartli operatorlar bundan mustasno.

Parolni tekshiruvchi dastur tuzamiz.

Dastur 3 ta oynadan iborat birinchi oyna quyidagi ko’rinishga ega:



18-rasm. Parolni tekshiruvchi dasturning asosiy oynasi.

Unda 1 ta tugma va 1 ta kiritish maydoni mavjud pastda parolni topishga yordam beruvchi iyyrogliflar(belgilar) joylashtirilgan. Bu oyna “dastlabki” deb nomlangan va undagi kiritish maydoni “parol” deb nomlangan. Agar parol to’g’ri kiritilib qizil tugma bosilsa, Unda quyidagi “bir” deb nomlagan oyna ochiladi:



19-rasm. Parolni tekshiruvchi dasturning birinchi oynasi.
Agar parol xato kiritilsa quyidagi “ikki” deb nomlangan oyna ochiladi:



20-rasm. Parolni tekshiruvchi dasturning ikkinchi oynasi.

Foydalanuvchi yana qayta parolni kiritib ko`rish imkoniga ega bo`lishi uchun pastda kichkina qora rangli tugma joylashtirilgan. Agar u bosilsa yana parol kiritiladigan oyna ochiladi.

Bir necha oyna (sahna) ochish uchun [shift]+[F2] tugmalarini bosamiz va bizga quyidagi oyna ochiladi:

Uning ostida joylashgan [+] tugmasi yordamida yangi oynalar yaratamiz. Dastlab 1-oyna “**Scene 1**”, 2-oyna “**Scene 2**”,..., deb nomlangan bo`ladi. Oyna nomi ustiga sichqonchani ikki marta chertib, uning nomini o`zgartirishimiz mumkin. Shu tariqa 3 ta oynani yaratib, ularni yuqoridagidek ko`rinishga keltiramiz. Shundan so`ng har bir oynaning birinchi kadriga `stop()` buyrug`ini yozib chiqamiz, aks holda oynalar to`xtovsiz namoyish qilib boriladi. Endi “dastlabki” oynadagi tugmaga kerakli skriptni yozamiz.

```
on (release) {
    if (parol=='ABC') {gotoAndStop('Bir',1);}
    else {gotoAndStop('ikki',1);}
}
```

Demak parol o`zgaruvchisining qiymati tekshiriladi. Agar u ABC gat eng bo`lsa, “bir” nomli oyna ochiladi, agar “ABC” gat eng bo`lmasa, “ikki” nomli oyna ochiladi.

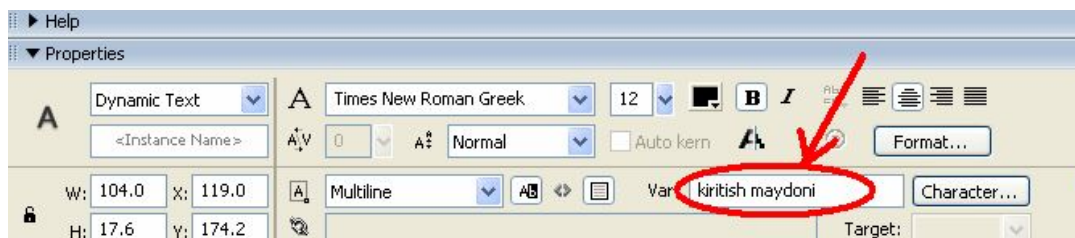
“ikki” nomli oynada joylashgan qora tugmaga esa quyidagi skriptni kiritamiz:

```
on (release) {
    gotoAndStop('dastlabki',1);}
}
```

Dinamik maydonlarning asosiy xossalari

Biz yuqorida foydalanuvchi ma`lumotini qabul qiluvchi kiritish maydonini hosil qilib ko`rgan edik. Endi dinamik maydonni hosil qilib ko`ramiz. Dinamik maydon,

nomidan ko`rinib turibdiki dastur bajarilish davomida dinamik ravishda o`zgarib boradi. Dinamik maydon ham xuddi kiritish maydoni singari hosil qilinadi. Faqat uni “properties” panelidan **“input text”** deb emas, uni **“dinamik text”** deb ko`rsatishimiz kerak. Shu panelning o`zidan dinamik maydonning shrifti, o`lchami, yozuv rangi, chegarasining ko`rinish ko`rinmasligini sozlab olishimiz va nomlashimiz mumkin. Dinamik maydonni nomlashda var kiritish maydonidan foydalanish afzal.



21-rasm. Dinamik maydonni nomlash.

Dinamik maydonni boshqarishga oddiy misol ko`rib o`taylik.

Tugma bosilganda ekranda “Salom dunyo” yozuvi ko`rinsin.

Dasturda 1 ta dinamik maydonni va 1 ta tugmani joylashtiramiz. Dinamik maydonni nomini “a1” deb nomlaymiz. Tugmaga quyidagi skriptni kiritamiz:

```
on (release) {
    _root.a1='Salom dunyo'; }
```

Bu yerda dinamik maydon o`zgaruvchi sifatida olinadi. O`zgaruvchiga quyidagi tarzda murojaat qilinadi: `_root.<o`zgaruvchi nomi>`. Xullas “a1” dinamik maydonga “Salom dunyo” satrini o`zlashtirdik.

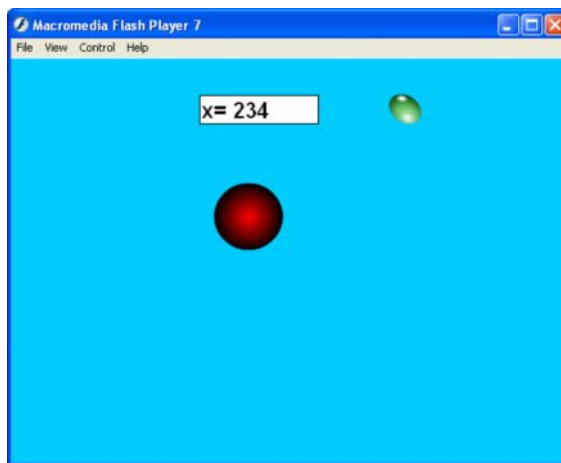
Endi yuqoridagi misolni qiyinlashtiramiz.

Har gal tugma bosilganda shar gorizontal 10 pikselga harkatlansin va uning x o`qi bo`yicha joriy koordinatalari dinamik maydonda ko`rinsin.

Oynaga 1 ta tugma, 1 ta “a1” nomli dinamik maydon va “koptok” nomli sharni joylashtiraiz va tugmaga quyidagi skriptni kiritamiz:

```
on (release) {
    koptok._x +=10;
    a1="x= "+koptok._x; }
```

Bunda `koptok._x +=10` koptokning `x` bo`yicha koordinatasini 10 pikselga oshiradi. `a1="x= "+koptok._x` “a1” dinamik o`zgaruvchisiga “x=” yozuvidan so`ng sharning joriy koordinatasini qo`shib qo`yadi. “x=” satr tipiga tegishli `koptok._x` son qiymatni qaytaradi. `Satr+Son=Satr` bo`lganligi uchun **dinamik matnga** satr yuboriladi. Dasturning oynasi quyidagi ko`rinishga ega bo`ladi:



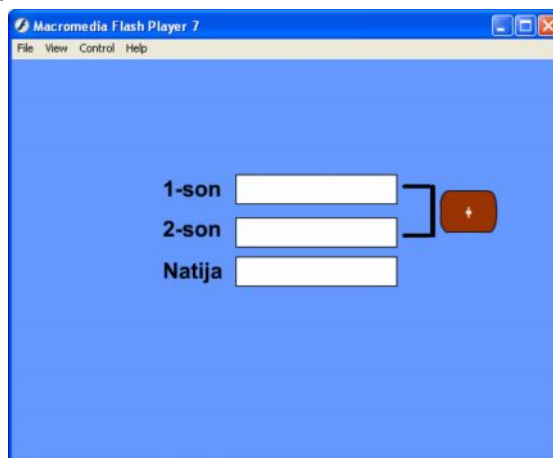
22-rasm. Obyekt koordinatasini aniqlash.

Oddiy hisoblagichni tayyorlash

Hisoblagichni tayyorlashdan oldin shuni ta'kidlab o'tish lozimki, matli maydon faqat sonlardan iborat bo'lsa ham, u satrli tipga tegishli. Matnli maydon qiymatini o'zida saqlovchi o'zgaruvchilar ham satrli tipga tegishli bo'ladi. Matnli maydonga sonlarni kiritib, ular ustida arifmetik amallarni bajarmoqchi bo'lsak, avval satrlarni songa aylantirishga to'g'ri keladi. Buni keyingi misolda batafsil ko'rib chiqamiz.

Ikki sonni qo'shuvchi hisoblagich yarating.

Oynaga ikkita kiritish maydoni (mos ravishda bir, ikki deb nomlangan), 1 ta dinamik maydon (natija deb nomlangan) va 1 ta tugma keltirib qo'yamiz, **static text** yordamida izohlar yozamiz:



23-rasm. Hisoblagich.

Endi tugmaga quyidagi skriptni kiritamiz:

```
on (release) {
    natija= bir+ikki;}
```

Dasturni ishga tushirib ishlatib ko'ramiz. 1-son sifatida 11 sonni ikkinchi son sifatida 21 sonini kiritib tugmani bossak natija sifatida "1121" xato qiymat qaytariladi. Sababi biz satrli o'zgaruvchini songa aylantirmadik. Shuning uchun **Actionscript** 1 va 2-sonlarni satr sifatida qo'shgan ya'ni ularni ketma-ket yozib qo'ygan. Endi skriptni to'g'irlaymiz. Buning uchun bizga satrni songa aylantiruvchi funksiya kerak bo'ladi bu funksiya **Number()** funksiyasidir.

Number() – funksiyasi satrni songa aylantirish uchun ishlatiladi.

Tugma skriptini quyidagicha o'zgartiramiz:

```
on (release) {
    natija=Number(bir)+Number(ikki); }
```

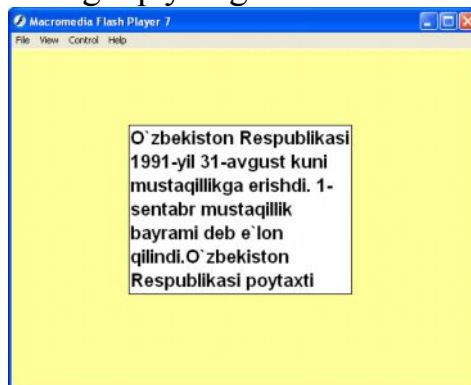
Endi dasturimiz to`g`ri natija qaytaradi.

Scroll funksiyasi

Scroll – funksiyasi bir necha satrli dinamik maydonni joriy vaqtda eng yuqorisidagi satrning tartibini o`zida saqlaydi. Dastlab u 1 qiymatini qabul qiladi. Chunki dastur ishga tushgach har doim 1-satr namoyish qilib turiladi. Keyinchalik uni qiymatini o`zgartirish mumkin. Endi uni amalda qo`llab ko`raylik. Ba`zan matnli maydon bir necha satrdan iborat bo`lishi mumkin. Agar ma`lumot ko`p bo`lsa, uning ayrim qismi maydon ostiga qolib ketadi. Foydalanuvchi ularni o`qishi uchun sichqoncha ko`rsatgichini matnli maydon ustiga chertib, so`ng yo`nalish tugmalari yordamida matnning ko`rinmas qismini o`qib olishi mumkin. Masalan ekranga 1 ta matn nomli, bir necha satrli dinamik maydon hosil qilamiz va 1-kadrga quyidagi skriptni kiritamiz:

```
matn='O`zbekiston Respublikasi 1991-yil 31-avgust kuni
mustaqillikga erishdi. 1-sentabr mustaqillik bayrami deb
e`lon qilindi.O`zbekiston Respublikasi poytaxti Toshkent
shahri. Yaqinda Toshkentning 2500 yilligi nishonlandi';
```

Scriptni ishlatib ko`rsak, ekranga quyidagi hosil bo`ladi:



24-rasm. Ma`lumot kiritilgan dinamik text.

Ko`rinib turibdiki matnning ayrim qismi ko`rinmay qolgan. Dasturga foydalanuvchiga qulaylik yaratish maqsadida matnli maydoning ko`rinmas qismini ekranga namoyish qiladigan tugmalarni o`rnatishimiz mumkin. Buni keyingi misolda qarab chiqamiz.

Matnning ko`rinmas qismini namoyish qiluvchi tugmalar hosil qilinsin.

Yuqoridagi misolni davom ettirib, dinamik maydon oldiga 2 ta tugma qo`shib qo`yamiz.

1- tugmaga quyidagi skriptni kiritamiz:

```
on (release) {
    matn.scroll++; }
```

Bu yerda `matn.scroll++;` dinamik maydonda namoyish qilinayotgan 1-satrga keyingi satrni o`zlashtiradi, ya`ni dastlab u 1 qiymatni qabul qiladi, agar tugma bosilsa uning qiymati 2 ga teng bo`ladi va 2-satr eng yuqorida namoyish etiladi. Xullas shu tarzda matnning barcha ko`rinmas qismini ko`rib o`tishimiz mumkin. Keyingi tugmada esa quyidagi skriptni kiritamiz:

```
on (release) {  
    matn.scroll--; }  
}
```

U tepadan pastga harakatlantiradi.

Klip holatlarini qayta ishlash haqida umumiy ma'lumotlar

Klip holatini qayta ishlash uchun quyidagi operator ishlatiladi:

```
onClipEvent () { }
```

Qavs ichiga esa har bir buyruqqa mos skript yoziladi. Ularni qisqacha izohlab o'tamiz. Holatlarni ikkiga bo'lishimiz mumkin. Bular

Klip namoyish qilinganda bajariladigan holatlar:

load – klip ekranga birinchi marta ko'ringanda, yoki klipning yangi nusxasini saqlovchi assosiy kadrda o'tganda bajariladigan holat.

unload – loadga teskari bo'lib, klip ekranda ko'rinmaganda bajariladigan holat.

enterFrame – bu holat klipning har bir kadriga o'tganda bajariladigan holat. Agar klip faqat bitta kadrda iborat bo'lsa ham u rolikning namoyish qilish vaqti bilan bir xil ravishda ya'ni 1 sekundda 12 marta tezlik bilan takrorlanadi.

data – holati klipga tashqi ma'lumot yuklanganda bajariladi. U matnli faylning ma'lumotlarini biror o'zgaruvchiga olganda, yoki tashqi swf fayllar yuklanganda bajariladigan holatni ifodalaydi.

Foydalanuvchi harakatiga ko'ra bajariladigan holatlar:

mouseDown – sichqonchani bosilganda bajariladigan holat. Bu holat sichqoncha ishchi stolning qaysi obyekt ustida bo'lmasin bajariladi.

mouseUp – sichqoncha ko'rsatgichi qaysi obyekt ustida bo'lmasin, uni tugmasini bosib, so'ng qo'yib yuborilganda bajariladigan holat.

mouseMove – sichqoncha tugmasi qaysi obyekt ustida bo'lmasin, o'ziga siljiganda bajariladigan holat.

keyDown – klaviaturadagi ixtiyoriy tugma bosilganda bajariladigan holat.

keyUp – klaviaturadagi bosilgan tugmani qo'yib yuborilganda bajariladigan holat.

Bu holatlarning har biriga keyinchalik batafsil to'xtalib, misollar asosida tushunirib o'tamiz. Hozir esa **load** va **unload** holatlari bilan tanishib o'tamiz.

Yonib o'huvchi chiroqni **load** va **unload** holatlaridan foydalanib tasvirlang.

1-asosiy kadrda uzib ulagichning yoqilgan holdagi rasmini keltirib qo'yamiz, 9-kadrgacha oddiy kadrlarni joylashtiramiz, 10-kadrdan asosiy kadr deb olib, unga uzib ulagichni o'chirilgan holdagi tasvirini hosil qilamiz. 19-kadrgacha oddiy kadrlarni qo'yamiz. Xullas, 1-kadrdan 9-kadrgacha ulangan uzib-ulagich, 10-kadrdan 19-kadrgacha o'chirilgan uzib ulagich tasvirlangan. Yana bir qatlam hosil qilib unga **movie clip** va **movie clip**dan tarqalayotgan yorug'lik (nur deb nomlangan) **movie clip**larni joylashtiramiz. Ular quyidagi rasmda tasvirlangan:

Bizda yorug'lik va **movie clip** alohida-alohida klip bo'lgani uchun ularni boshqarish oddiy. 10-kadrga o'tib uzib-ulagichni belgilaymiz va unga quyidagi skriptni yozamiz:

```
onClipEvent (load) {  
    _root.nur._visible=0;  
}  
onClipEvent (unload) {
```

```

        _root.nur._visible=1;
    }

```

Demak 10-kadrga kelganda `_root.nur._visible=0;` buyrug'I bajariladi, ya'ni nur ko'rinmaydi. 19-kadrga kelganda `_root.nur._visible=1;` buyrug'I yordamida yana qaytib ko'rinadi. Dasturni ishga tushirsak, uzib ulagich ulanadi, **movie clip** yoqiladi, uzib-ulagich o'chiriladi va **movie clip** o'chadi, shu rolik to'xtovsiz namoyish qilinib boriladi.

EnterFrame holati

O'ngdan chapga tomon harkatlanuvchi sharni dasturlash orqali hosil qiling. Dasturni tuzish uchun ekranning chap tomoniga birorta shar rasmi tasvirlangan moie klipni joylashtiramiz. So'ng unga quyidagi skriptni yozamiz::

```

onClipEvent (enterFrame) {
    _x+=10; }

```

Shar to'xtovsiz chapdan o'ngga harakatlanadi.

To'xtovsiz ravishda avval kattalashib so'ng kichiklashuvchi shar tasvirini **interFrame** holati yordamida hosil qiling.

Oynaga 1 ta shar rasmi chizilgn **movie clip**ni joylashtiramiz. O'ylangan g'oyani amalga oshirish uchun bizga 1 ta qo'shimcha o'zgaruvchi kerak bo'ladi. Uning qiymati faqat bir marta klip namoyishi boshlanganda aniqlanishi kerak. Buning uchun uni load holatida joylashtiramiz va uning qiymati sifatida **true** (chin)ni yozamiz:

```

onClipEvent (load) {
    a=true;}

```

Keyingi qadam shart operatori yordamida tekshiramiz, agar sharning hajmi 400 pikseldan katta bo'lmasa va `a=true` qiymat qabul qilgan bo'lsa, u holda shar vertikalva gorizontal 10 rikselga kattalashadi. Agar qo'yilgan shart bajarilmasa, `a=false` qiymatni qabul qiladi. Bunda shar 400 pikselgacha kattalashadi. Bu quyidagicha ifodalaniladi:

```

onClipEvent (enterFrame) {
    if (_xscale<400 && a) {
        _xscale+=10;
        _yscale+=10;
    }else {
        a=false;
    } }

```

Sharni qayta kichiklashtirish uchun quyidagi skriptni kiritamiz:

```

onClipEvent (enterFrame) {
    if (_xscale>10 && a==false) {
        _xscale-=10;
        _yscale-=10;
    }else {
        a=true;
    } }

```

Shu skriptlarni hammasini **movie clip**ga kiritsak, Shar pulsiv tarzda kattalashib, kichiklashadi.

getTimer () funksiyasi

getTimer() o'tgan millisekundni qaytaradi. Ushbu funksiya yordamida vaqtni ifodalovchi **taymer**lar yaratishimiz, rolikning biror kadrini bir necha soniyaga to'xtatib qo'yishimiz mumkin. Uning qo'llanilishini misol asosida ko'rib o'tamiz.

O'tilgan vaqtni ifodalovchi taymer hosil qlamiz.

Vaqt degan **movie clip** hosil qilamiz. Unda 1 satrli "soniya" nomli dinamik maydonni keltirib qo'yamiz. **Movie clip**ni oynaga keltirib qo'yamiz va unga quyidagi skriptni kiritamiz:

```
onClipEvent (enterFrame) {  
    a=Math.floor(getTimer()/1000);  
    soniya='o'tgan vaqt:'+a; }
```

getTimer() funksiyasi millisekundni qaytaradi bizga esa sekundlar ko'rinishi kerak, shuning uchun uning qiymatini 1000 ga bo'lib olamiz va Math.floor() funksiyasi yordamida uning qiymatini eng kichik butun songacha yaxlitlab olamiz. "Soniya" nomli obyektga uning qiymatini izoh bilan chiqaramiz.

Agar vaqt teskari tartibda kamayib borishini xohlasak, unda **movie clip**ga quyidagi skriptni kiritamiz:

```
onClipEvent (enterFrame) {  
    a=Math.floor(getTimer()/1000);  
    if (a<10){  
        soniya=(10-a)+' soniya qoldi';  
    } else {soniya='vaqt tugadi'}}
```

Bu dasturda o'tilgan vaqt 10 soni bilan tekshiriladi. Agar o'tgan vaqt 10 dan kam bo'lsa unda o'tgan soniyalar chiqariladi. Agar 10 ga teng yoki 10 dan katta bo'lsa, vaqt tugadi degan yozuv ko'rinadi.

Misol sifatida, elektron kitob yaratamiz. Har 5 sekundda uning yangi sahifasi ko'rinsin.

Oynaga 1 ta dinamik maydondan iborat bo'lgan **movie clip**ni oynaning 1-qatlamiga olib qo'yamiz. Buning uchun avval yangi **movie clip**ni hosil qilamiz va unga 1 ta dinamik maydonni keltirib qo'yamiz. Dinamik maydonni "soniya" deb nomlaymiz. Oynhaga o'tib F11 tugmasini bosamiz va kutubxonadan dinamik maydondan iborat **movie clip**ni oynaga olib qo'yamiz. So'ng ikkinchi qatlamni hosil qilamiz. Unda 1-kadrga **static text** o'rnatib kitobning birinchi betidasgi ma'lumotlarni kiritamiuz. 2-kadriga ham static text qo'yib, unga kitobning 2-betidagi ma'lumotlarni kiritamiz... 1-qatlamning 1-kadriga joylashgan **movie clip**ga quyidagi skriptni kiritamiz:

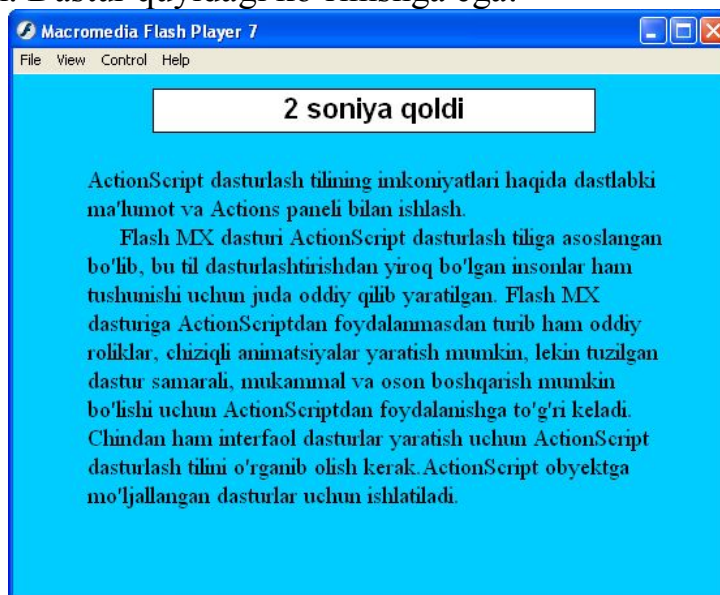
```
onClipEvent (load) {  
    q=0; }  
onClipEvent (enterFrame) {  
    a = Math.floor((getTimer()-q)/1000);  
    if (a<5) {  
        soniya = (5-a)+' soniya qoldi';  
    } else {  
        _root.nextFrame();  
        q=getTimer(); } }
```

Avval q o'zgaruvchisining qiymatini o'tilgan vaqtga teng deb olamiz va a o'zgaruvchisiga o'tgan vaqtdan q ni ayirib, so'ng uni soniyaga aylantiramiz. Agar aning qiymati beshdan kichik bo'lsa, o'tgan vaqt kamayib borish tartibida izoh bilan e'lon

qilinadi. Agar 5 ga teng bo'lsa, keyingi kadr namayish qilinadi. **nextFrame()** – keyingi kadrda o'tish buyrug'idir. Biz unga o'zgaruvchi sifatida murojaat qilamiz, shuning uchun uning oldiga `_root` so'zini qo'shib qo'yamiz. Q ga o'tgan vaqtni olamiz. Bu keyingi galda a ning qiymatini yana 5 dan boshlash uchun kerak. Xullas dastur ishga tushgach bizga birinchi sahifa ko'rinadi, 5 soniyadan so'ng ikkinchi, yana 5 soniyadan so'ng uchinchi,... oxirgi sahifaga kelganda **timer** 5 dan 1 gacha teskari tartibda sanaydi va yana hechqanday o'zgarishsiz yana 5 dan 1 gacha teskari tartibda sanaydi. Bizga **timerning** to'xtovsiz sanashi kerak emas. Shuning uchun 1-qatlamning 5-kadriga asosiy kadrni hosil qilib, **movie clip**ni tanlaymiz va unga quyidagi skriptni kiritamiz:

```
onClipEvent (load) {
    q=getTimer(); }
onClipEvent (enterFrame) {
    a = Math.floor((getTimer()-q)/1000);
    if (a<5) {
        soniya = (5-a)+' soniya qoldi';
    } else {
        soniya='Bu kitobning oxirgi beti';    }}
```

Bunda 5-sahifa yangilanganda 5 soniyadan so'ng bizga “bu kitobning oxirgi beti” degan yozuv ko'rinadi. Dastur quyidagi ko'rinishga ega:



25-rasm. Elektron kitob.

Ekranda namoyish qilinadigan rolikni bir necha soniyaga to'xtatib turing va rolikni namoyishini davom ettirish.

Buning uchun tayyor rolikning to'xtashi lozim bo'lgan kadriga quyidagi skriptni kiriting.

```
q=getTimer();
while (getTimer()-q<200){
    stop();} play();
```

q ga joriy vaqtni olamiz. Uning qiymati bizni qiziqitirmaydi. So'ng dastlabki vaqtning qiymati joriy vaqtning qiymatidan 200 ms farq qilgunga qadar `stop()`; buyrug'i amal qiladi, ya'ni rolik namoyishi to'xtaydi. 200 ms so'ng `play()`; buyrug'i amal qiladi, ya'ni rolik namoyishi davom ettiriladi.

mouseMovie vasifasi

mouseMovie holati sichqoncha ishchi maydon ustiga siljigandagi holat bo'lib u sichqoncha siljiganda bir necha marta protsessorning holatni qayta ishlash tezligiga bog'liq tezlikda ishlaydi.

Shaxsiy kursorini yaratish masalasini qaraymiz.

Shaxsiy kursorini yaratish uchun avvalambor shu kursorning rasmini chizish, yoki uni **import** qilish lozim. Biz kursor sifatida archani tanlaymiz. Biror tugma ustiga kelganda kursorning ko'rinishi o'zgarganligi tufayli 2-rasm sifatida strelkani tanlaymiz. 1 ta **movie clip** hosil qilib, uni 1-kadriga archaning 2-kadriga chiroqning rasmini keltirib qo'yamiz. Har ikkala kadrda ham stop()buyrug'ini yozib, **Movie clip**ni kursor deb nomlaymiz va uni oynaga keltirib qo'yamiz. Endi kursorni ko'rinmas holatga keltiramiz. Uni faqat bir marta ko'rinmas holga keltirish kifoya, shuning uchun uni **load** holatiga joylashtiramiz:

```
onClipEvent (load) {  
    Mouse.hide(); }
```

Hosil qilgan kursorimiz, ko'rinmas kursor ortidan harakatlanishi uchun, yangi kursorga ko'rinmas kursorning koordinatalarini o'zlashtirishimiz lozim. Sichqoncha faqat siljiganda uning koordinatalari o'zgaradi. Shuning uchun uni **mouseMove** holatiga joylashtiramiz. Ushbu skriptni enterFrame holatiga joylashtirsak ham bo'ladi, lekin bu holat sichqoncha siljimaganda ham har doim bajarilib turganligi tufayli u protsessorga ortiqcha vazifa yuklaydi. Shuning uchun skriptni mouseMove holatiga yozamiz.

```
onClipEvent (mouseMove) {  
    _x=_root._xmouse;  
    _y=_root._ymouse;  
    updateAfterEvent(); }
```

Bu yerda _xmouse Ko'rinmas kursorning koordinatalari va _x joriy kursorning koordinatasi. updateAfterEvent() – rolikning namoyishini tezlashtirish uchun ishlatiladi. Agar ushbu skriptni yozmasak sichqoncha harakatlanganda hosil qilgan yangi kursorimiz to'xtab-to'xtab harakatlanadi, chunki uning harakati filning tazligiga bog'liq bo'ladi. Bu buyruq faqat **mouseMove** holatining ichidagina ishlaydi. Dasturni testdan o'tkazib, uni tekshirib ko'ramiz.

Navbatdagi masala sichqoncha tugma ustiga kelganda uning ko'rinishi o'zgarishi kerak. Buning uchun ekranga tugmani joylashtiramiz. Bunda tugma 2-qatlamda, kursor joylashgan qatlam ostida joylashgan bo'lishi lozim, aks holda kursor tugma tagiga qoladi. Tugmaga quyidagi skriptni ko'ramiz:

```
on (rollOver) {  
    cursor.gotoAndStop(2); }  
on (rollOut) {  
    cursor.gotoAndStop(1); }
```

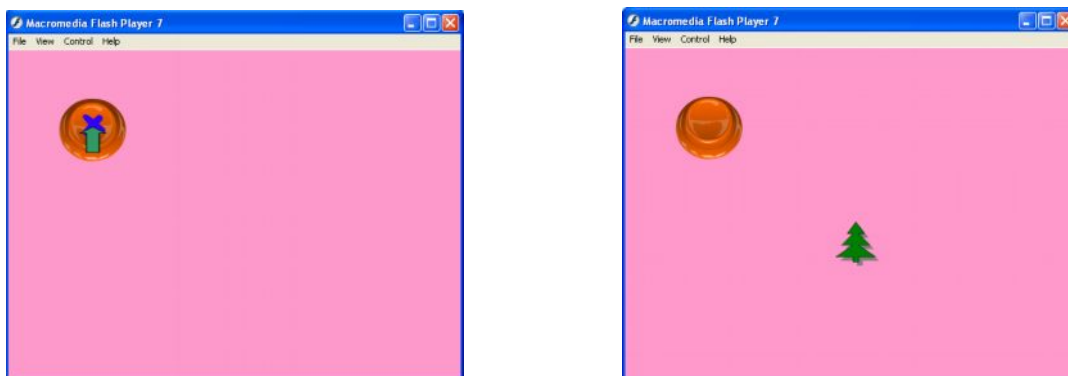
Bunda tugma ustiga sichqoncha ko'rsatkichi kelganda kursorning 2-ko'rinishi, ya'ni sichqonchanning kursori, tugma ustidan ketganda 1-ko'rinishi ya'ni archa ko'rinadi.

Bulardan tashqari sichqoncha tugmasi bosilganda uning bosilganligini bildiruvchi biror o'zgarish bo'lishi kerak bo'lsa, masalan, ko'rsatkich uchida yorqin rang bo'lishini

xohlasak, quyidagilarni amalgam oshirish lozim. Avval yorqin rangni hosil qilamiz. Men quyidagini tanladim: ✕. Uni oynadan tashqariga ko`rinmas holda joylashtiramiz va unga quyidagi skriptni kiritamiz:

```
onClipEvent (mouseDown) {  
    _x = _root._xmouse;  
    _y = _root._ymouse;  
    this._visible = 1;  
    updateAfterEvent();}  
onClipEvent (mouseUp) {  
    this._visible = 0;  
    updateAfterEvent();}
```

Ushbu skript quyidagicha o`qiladi: Sichqoncha tugmasi bosilganda, uning koordinatalarini ushbu **movie clip**ga o`zlashtir va **movie clip**ni ko`rinadigan holatga keltir, hodisani tezlashtir. Sichqoncha tugmasi qo`yib yuborilganda, **movie clip**ni ko`rinmas holga keltir va hodisani tezlashtir. Dasturni ishga tushirsak bizga tayyor foydalanuvchi kursori hosil bo`ladi. Quyida hosil qilingan kursorning tugma ustuda bosilgandagi va tugma ustida bo`lmagan holatlarining tasviri keltirilgan.



26-rasm. Ko`rinishi o`zgartirilgan sichqoncha ko`rinishi va tugma.

Linza yordamida kichkina xaritani kattalashtirish masalasini ko`ramiz.

Xarita rasmi tasvirlangan **movie clip**ni oyna markaziga joylashtiramiz. Linza nomli **movie clip** hosil qilamiz va unga xarita rasmini joylashtiramiz, xaritaga “mc” deb nom qo`yamiz. **Alligin** oynasidan foydalanib, uni markazga joylashtiramiz. Ikkinchi qatlamni hosil qilib, unga linzaning rasmini keltirib qo`yamiz va markazga joylashtiramiz. 1-va 2-qatamlar orasida yangi 3-qatlamni hosil qilamiz va unga “xxx” deb nom beramiz. Bu qatlamda pastki qatlamda jolashgan rasmning ko`rinmasligi kerak bo`lgan qismi yashiriladi. Linza aylanisiga mos keluvchi aylanani rasmini chizamiz, uni linza ichiga choylashtiramiz. U bizga tuynuk vazifasini bajaradi. So`ng, “xxx” qatlam ustiga sichqonchanning o`ng tugmasini bosamiz va maska bo`limini tanlaymiz.

Maskada joylashgan grafika rolikda namoyon qilinmaydi va undan pastda joylashgan, niqoblangan qatlamlardagi rasmlarni namoyon qiluvchi oyna (darcha) vazifasini bajaradi. **Statik text**, grafik simvollar va rasmlarni maska sifatida ishlatish mumkin, agar ular 1 qatlamda joylashgan bo`lsa. **Dinamik text** va chiziqlar maska bo`lolmaydi.

xxx va 1-qavat orasiga yana bir qavatni hosil qilamiz va uni eng pastga ko`chiramiz. Hosil qilingan qatlamga xxx qatlamdagi aylanani nusxasini keltirib

qo'yamiz. U ham maska qilingan qatlam ichiga kirib, fon vazifasini bajaradi. Linza orqali qaralganda rasmning asl nusxasi ko'rinmasligi uchun ishlatiladi, aylananing rangini oyna rangi bilan bir xil qilish kerak.

Oynaga hosil qilingan linza **movie clip**ini keltitib qo'yamiz. Linzaga quyidagi skriptni kiritamiz:

```
onClipEvent (load) {  
    n = 2; }  
n o'zgaruvchisi linzaning necha marta kattalashishini belgilaydi.
```

```
onClipEvent (enterFrame) {  
    _xscale = 100*n;  
    _yscale = 100*n;  
    _x = _root._xmouse;  
    _y = _root._ymouse;  
    mc._x = -_x+275;  
    mc._y = -_y+200; }
```

Xaritaning o'lchamini x va y o'qlari bo'yicha 200% (n=2 bo'lganda) ga oshiradi, linzaning koordinatalarini sichqonchanning koordinatalari bilan bir xil qilib boriladi. Bunda linza sichqoncha ko'rsatkichi bilan harakatlana boshlaydi. $mc._x = -_x+275$ xaritaning gorizontol koordinatasiga linzaning gorizontol koordinatasi qarama-qarshi ishora bilan o'zlastilyapti, bunda xarita linza harakatlangan tomonga qarama-qarshi harakatlanadi. Xaritaning mc nomli nusxasi joriy kadr markazida joylashgan, oynadagi xaritaning rasmi esa, oyna markaziga joylashgan. Ular orasidagi farq 275, 200 bo'lganligi uchun x va y koordinatalarga ular mos ravishda qo'shiladi. Shu skript yozilgandan so'ng linza markaziga xaritamizning joriy qismi tasvirlanadi.

```
onClipEvent (mouseDown) { n = 4; }
```

```
onClipEvent (mouseUp) { n = 2; }
```

Sichqoncha bosilganda xaritaning o'lchami 4 marta oshadi, qo'yib yuborilganda xarita yana 2 marta kattalashadi. Dasturimiz quyidagi ko'rinishga ega:



27-rasm. Xaritani kattalashtirish.

Key obyektining asosiy metodlari

Bu bo'limda biz Flash obyektlarini klaviatura yordamida boshqarishni qarab chiqamiz. Hamda, **key** obyektining asosiy metodlari bilan tanishib chiqamiz. Bu obyekt bizga hozirgi vaqtda qanday tugma bosilgan va oxirgi bosilgan tugmaning qaysiligini aniqlash uchun ishlatiladi. Klaviaturadagi ixtiyoriy tugma bosilganda, 3 xil holat sodir

bo'ladi. Bular **keyPress**, **keyDown** va **keyUp**. Ularning har birini imkoniyatlarini misollarda qarab o'tamiz.

Yo'nalish tugmalari yordamda **movie clip**ni harakatlantirish.

Buning uchun ekranga 1 ta **movie clip**ni keltirib qo'yamiz. Biz **movie clip** sifatida quyonni rasmi tasvirlangan **movie clip**ni tanladik. Bu **movie clip**ning 1-4 gacha quyon oyoqlarini qimirlatib yuradigan holdagi tasviri o'rin olgan 5-8 gacha quyonning orqa tomondagi tasviri tasvirlanib bunda ham quyon harakati tasvirlangan.

4- kadrğa

```
gotoAndPlay(1);
```

8-kadrga

```
gotoAndPlay(5);
```

buyruqlari yozilgan. Dastur quyidagicha ishlashi kerak. Agar pastga qaralgan yo'nalish tugmasi bosilsa, quyon pastga 3 pikselga ekranga qaragan holatda harakatlanib to'xtashi kerak. Agar tepaga qaralgan yo'nalish tugmasi bosilsa, quyon tepaga ekranga orqa tomonini o'girib 3 pikselga harakatlanib to'xtashi kerak. Agar o'ng va chap tugmalar bosilsa, quyon bundan oldin qanday holatda bo'lsa, shu holatda (ekranga qarab o'tirgan bo'lsa, ekranga qaragan holda, ekranga o'girilgan bo'lsa, o'girilgan holda) mos ravishda o'ngga va chapga 3 pikselga harakatlanib to'xtashi kerak. Demak quyon rasmi tasvirlangan **movie clip**ni ekranga o'rnatib, uni quyon deb nomlaymiz va unga quyidagi skriptni yozamiz:

```
onClipEvent (keyDown) {  
    if (Key.isDown(38)) {  
        _y -= 3;  
        _root.quyon.gotoAndPlay(5);  
    }  
    //yuqoriga qaragan tugma bosilganda  
    if (Key.isDown(40)) {  
        _y += 3;  
        _root.quyon.gotoAndPlay(1);  
    }  
    //pastga qaragan tugma bosilganda  
    if (Key.isDown(37)) {  
        _x -= 3; } //o'ngga qaragan tugma bosilganda  
    if (Key.isDown(39)) { _x += 3;  
    }  
    //chapga qaragan tugma bosilganda  
    updateAfterEvent(); }
```

Bu yerda `Key.isDown(38)` klaviaturadan qaysi tugma bosilganini tekshiradi. Qavs ichidagi son o'rniga har bir tugmaning ASCII kodini yozish kerak. Flashda ko'p ishlatiladigan klavishlarning nomi konstanta deb e'lon qilingan. Qavs ichidagi son o'rniga klavishning nomini yozsak ham bo'ladi. Masalan bo'sh joy tugmasining bosilganligini rekshirmoqchi bo'lsak shartga `Key.isDown(SPACE)` deb yozishimiz mumkin. Dastur ko'rinishini o'zgartirish uchun unga quyosh va osmon rasmini chizamiz. Dastur quyidagi ko'rinishga ega bo'ladi.



30-rasm. Tugma orqali obyektни harakatlantirish.

Klaviaturadagi bosilgan ixtiyoriy klavishning skriptini bilib olish uchun ekranga biror **movie clip**ni hosil qilib unga quyidagi skriptni o`zlashtiramiz:

```
onClipEvent (keyDown) {
    trace (Key.getCode ());
}
```

Bu yerda **Key.getCode()** – oxirgi bosilgan tugmaning skriptini qaytaradi.

Biror tugma bosilganda tezlanish bilan harakatlanuvchi **movie clip**ni hosil qiling.

Ekranga – joyidan qimirlamagan holda oyoq-qo`llarini harakatlantiruvchi, toshbaqa **movie clip**ini keltirib qo`yamiz. Yo`nalish tugmalari bosilganda bu toshbaqa harakatlanishi, agar tugma bosib turilsa, u tezlanuvchan harakatlanishi kerak. Toshbaqa **movie clip**iga quyidagi skriptlarni kiritamiz:

```
onClipEvent (load) {
    a = 0;
    b = 0;
}
```

a,b o`zgaruvchilariga 0 qiymatini o`zlashtiramiz. Bu o`zgaruvchilar, **movie clip**ning x va y o`qlari bo`yicha nechki pikselga siljishini aniqlaydi.

```
onClipEvent (enterFrame) {
    if (Key.isDown(38)) {
        b--;
    }
    /*agar yuqoriga qaragan yo`nalish tugmasi bosib turilsa, b
    ning qiymati 1 pikselga kamayib boraveradi.*/
    if (Key.isDown(40)) {
        b++;
    }
    if (Key.isDown(37)) {
        a--;
    }
    if (Key.isDown(39)) {
        a++;
    }
    _y += b;
    _x += a;
}
```

*/*x va y o`qlariga a va b ning qiymatlari qo`shiladi. Masalan agar yuqoriga qaragan yo`nalish tugmasi b ning qiymati 12 ga teng bo`lgunga qadar bosib turilsa, toshbaqa 12 piksel tezlik bilan yuqoriga qarab harakatlana boshlaydi. Agar pastga qaralgan yo`nalish tugmasi bosib turilsa, b manfiy qiymat qabul qiladi, y ga manfiy qiymat qo`shiladi va toshbaqa pastga qarab harakatlanib boradi.*/*

Dastur quyidagi ko`rinishga ega:



28-rasm. Obyektga tezlanish berish.

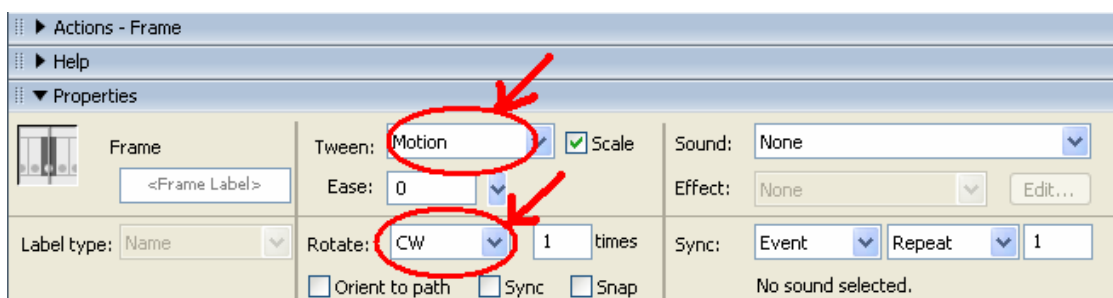
Quyosh atrofida aylanuvchi yer tasvirini hosil qiling. Yerning tezligini va yo`nalishini o`ng va chap yo`nalish tugmalari orqali boshqarish mumkin bo`lsin.

Yangi **movie clip**ni hosil qilamiz. Unga yerning rasmini keltirib qo`yamiz. Oynaga shu **movie clip**ni joylashtiramiz.. Uni berilgan nuqta atrofida aylanadigan qilib harakatlantiramiz. Buning uchun **movie clip**ni **Tools** panelida joylashgan  – **free transform tool(q)** asbobi yordamida belgilab olamiz va uni markazida joylashgan nuqtani yerning tasviridan uzoqroq masofada joylashtiramiz:



29-rasm. Obyektni berilgan nuqta atrofida harakatlantirish.

Yuqoridagi rasmda ushbu nuqta yo`nalish tugmasi yordamida ko`rsatilgan. Yer va nuqta orasidagi masofa radius bo`lib yer shu radiusda aylana boshlaydi. 30-kadrni asosiy kadr deb ko`rsatamiz va qaytib 1-kadrga o`tamiz. **Properties** panelidagi “Tween” yozuvi oldidagi ochiluvchi ro`yxatdan “Motion” yozuvini tanlaymiz. “Rotate” yozuvi oldidagi ochiluvchi ro`yxatdan “CW” yozuvini tanlaymiz.



30-rasm. Motion buyrug`ga murojaat

Bunda yer markaziy nuqta atrofida aylana boshlaydi.

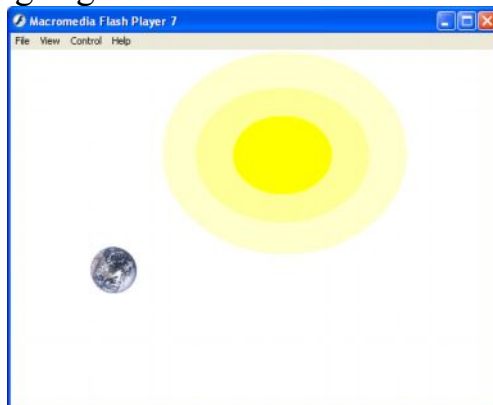
Animatsiya o'z-o'zidan namoyish qilinmasligi uchun uni 1- kadrda to'xtatib qo'yamiz (buning uchun 1-kadrga stop() buyrug'ini kiritamiz). Oynaga yangi qatlam hosil qilib unga quyoshning rasmini ham keltirib qo'yamiz. Dasturni boshqarish uchun bizga bo'sh (hech narsani tasviri aks etmagan) **movie clip** kerak bo'ladi. Yangi **movie clip**ni hosil qilamiz va unga hech narsaning rasmini chizmay, uni ekranga, yangi qatlam hosil qilib, shu qatlamga olib qo'yamiz. **Actions** oynasini ochib, bo'sh **movie clip**ga quyidagi skriptni yozamiz:

```
onClipEvent (load) {
    k=1; n=1;
    tf=_root._totalframes-1; }
```

k va **n** ga boshlang'ich qiymat berib, tf o'zgaruvchisiga animatsiyadagi kadrlar soninidan 1 ta kam sonni o'zlashtiramiz. K o'zgaruvchisi rolikning tezligini boshqarish uchun, n esa namoyish qilinayotgan kadr raqamini saqlash uchun ishlatiladi.

```
onClipEvent (enterFrame) {
    n+=k;
    if (n>tf) {
        n-=tf }
    if (n<1) {
        n+=tf }
    _root.gotoAndStop(n);
    if (Key.isDown(Key.LEFT)) {
        k--;}
    if (Key.isDown(Key.RIGHT)) {
        k++; }
}
```

Dastur quyidagi ko'rinishga ega:



31-rasm. Quyosh atrofida yerning aylanishi.

Dastur yordamida flash simvollari nusxasini hosil qilish

Biror obyektning nusxasini hosil qilish uchun, har bir obyektни oynaga 1 tadan keltirib qo'yish shart emas. Maxsus buyruqlar orqali biror obyektning bir necha nusxalarini hosil qilib bo'ladi. Masalan 1 dona tomchi tasviridan foydalanib, yomg'irni tasvirlash mumkin. Hosil qilingan nusxalarni o'chirish ham mumkin, lekin oynaga oldindan joylashgan obyektlarni o'chirishning imkoni yo'q. Oynada hosil qilingan har bir obyektning o'z nomi bo'ladi. Oynada joylashgan obyektning nusxasini hosil qilish uchun quyidagi buyruqdan foydalanamiz:

duplicateMovieClip("<a1>", "<a2>", <a3>);

Bu yerda a1- asl nusxaning nomi, a2- hosil qilingan nusxaning nomi, a3- obyektning nusxasi joylashishi kerak bo'lgan stekning chuqurlik darajasi bo'lib son qiymat qabul qiladi. Ikkita klip ustma-ust tushganda stek darajasi past bo'lgan klip, darajasi yuqori bo'lgan klip ostiga qolib ketadi.

Oynada hosil qilingan nusxani yo'qotish uchun quyidagi funksiyadan foydalanamiz:

removeMovieClip("a1")

Bu yerda a1-o'chirilishi kerak bo'lan nusxaning nomi.

Yana bir funksiya bilan tanishib o'tamiz, bu funksiya bizlarga obyektning ixtiyoriy xususiyatini o'zgartirish uchun kerak bo'ladi. Bu funksiya quyidagi ko'rinishga ega:

setProperty("<a1>",<a2>,<a3>);

Bu yerda a1-obyekt nomi; a2- bu yerda o'zgartirilishi kerak bo'lgan xususiyat yoziladi. a3-xususiyatning qiymati; Endi Bu funksiyalarni amalda qo'llab ko'ramiz.

Ekranda yulduzli osmonni hosil qilish masalasini ko'ramiz.

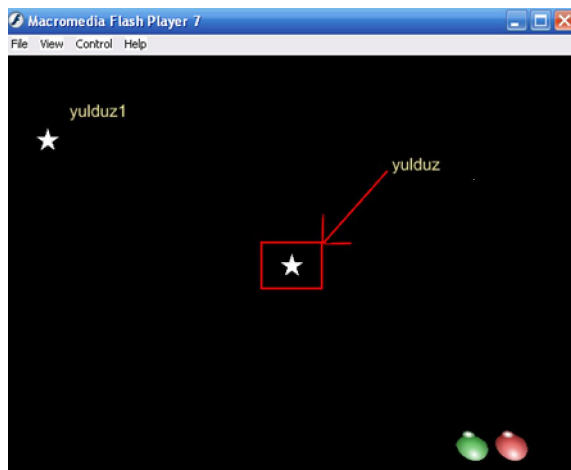
Buning uchun dastlab **movie clip**ni hosil qilamiz. Unga yulduz rasmini chizamiz. Ishchi oynaga shu **movie clip**ni "yulduz" nomi bilan keltirib qo'yamiz. Avval 1 ta **movie clip**ni 2- nusxasini hosil qilib ko'ramiz. Buning uchun ekranga 2 ta tugma qo'yamiz. 1-tugma yashil rangda bo'lib, u bosilganda oynadagi yulduzning 2- nusxasi ko'rinsin, 2-tugma qizil rangda bo'lib, ushbu tugma bosilganda yulduzning nusxasi yo'qolsin. Avval 1 - tugmani belgilab **Actions** oynasini ochamiz va unga quyidagi skriptni yozamiz:

```
on (release) {  
    duplicateMovieClip("yulduz", "yulduz1", 1);  
}
```

Lekin dasturni ishga tushirsak, hech qanday hodisa sodir bo'lmaydi. Aniqroq qilib aytadigan bo'lsak, sodir bo'ladi, lekin biz buni ko'rolmaymiz. Sababi biz hosil qilgan yulduz nomli **movie clip**imizning nusxasi "yulduz1" nomi bilan yaratilyapti lekin **1-movie clip** bilan ustma-ust tushayapti. Nusxani hosil bo'lganiga aniq ishonch hosil qilish uchun uni avvalgi yulduzning koordinatalaridan biroz uzoqroqqa joylashtiramiz. Buning uchun skriptni quyidagicha yozamiz:

```
on (release) {  
    duplicateMovieClip("yulduz", "yulduz1", 1);  
    setProperty("yulduz1", _x, 40);  
    setProperty("yulduz1", _y, 80);  
}
```

Dastur ishga tushgandan keyin, yashil tugma bosilganda, 1-yulduzning ikkinchi nusxasi hosil bo'ladi.



32-rasm. Obyektning birta nusxasini hosil qilish

Endigi ishimiz qizil tugma bosilganda shu hosil qilingan nusxani yo`qotish. Buning uchun bu tugmaga quyidagi skriptni yozamiz:

```
on (release) {  
    removeMovieClip("yulduz1");}
```

Bunda yulduz1 nomli nusxalangan obyekt yo`qoladi.

Endi yashil tugma bosilganda 1 dona emas bir nechta nusxani hosil qilishni qarab o`tamiz. Buning uchun yashil tugmaga quyidagilarni kiritamiz:

```
on (release) {  
    for (i=1; i<=10; i++) {  
        duplicateMovieClip("yulduz", "yulduz"+i, i);  
        setProperty("yulduz"+i, _x, 40*i);  
        setProperty("yulduz"+i, _y, 80);}}
```

Biz for takrorlash operatori bilan oldin tanishib o`tganmiz. Uning dastlabki ikki qadamini qarab o`tamiz.

1-qadam: i o`zgaruvchisi 1-qadamda 1 qiymatini qabul qiladi. **duplicateMovieClip** “yulduz” nomli obyektning nusxasini yaratadi. Bu nusxani nomlayotgan vaqtda “yulduz” satrini oladi va unga i ning joriy qiymatini qo`shadi (i ning joriy qiymati 1 ga teng) “satr”+“son”=“satr” bo`lgani uchun yulduz(satr)+1(son)=yulduz1(satr) bo`ladi. Demak hosil qilingan nusxaning nomi “yulduz1”, Biz bu takrorlanish jarayoniga 10 ta yulduz hosil qilamiz. Ularning steklari ustma-ust tushmasligi uchun har bir yulduzni alohida stekka qo`yib chiqamiz. 1-yulduzni birinchi, 2-yulduzni ikkinchi va hokazo. steklarga joylashtirib chiqamiz. “Yulduz1” nomli nusxamiz 1-stekka joylashgan. **duplicateMovieClip** ning vazifasi tugadi. `setProperty("yulduz"+i, _x, 40*i)`—bunda har bir nusxaning nomiga murojaat qilinyapti, 1-hol uchun yulduz satriga 1 sonini qo`shib yulduz1 obyektiga murojaat qiladi, so`ng uning x o`qi bo`yicha koordinatasining qiymati ( $40*i$ ) ekanligi ko`rsatilyapti, ya`ni  $40*1=40$  demak x bo`yicha koordinatasi 40 ga teng. y o`qi bo`yicha koordinatasani 80 deb olinadi. Hamma nusxalarning y o`qi bo`yicha koordinatasi bir xil bo`lganligi uchun, ular 1 qatorda joylashadi.

2-qadam: i o`zgaruvchining qiymati 2 ga teng bo`ladi. Yulduz obyektining yulduz2 nomli nusxasi 2-stekka yaratilib qo`yiladi. yulduz2 obyektning x o`qi bo`yicha koordinatasi 80 ( $40*2=80$ ) ga teng bo`ladi. Uning y o`qi bo`yicha koordinatasi ham 80 ga teng.

Qolgan qadamlar ham xuddi shu tartibda bajariladi va berilgan 1 dona yulduzning 10 ta nusxasi ekranga 1 qatorda joylashtirilib chiqiladi.

Qizil tugmaga quyidagi skriptni kiritamiz:

```
on (release) {  
    for (i=1; i<=10; i++) {  
        removeMovieClip("yulduz"+i);  
    }  
}
```

Bu bu takrorlanish jarayonida hosil qilingan 10 ta yulduzning hammasi o`chirilyapti. 1-qadamda i o`zgaruvchisining qiymati 1 ga teng bo`ladi va “yulduz1” nomli nusxa o`chiriladi, ikkinchi qadamda i ning qiymati 2 ga teng demak “yulduz2” nomli obyekt o`chiriladi va boshqalar.

Dastur quyidagi ko`rinishga ega bo`ladi:



33-rasm. Obyektning n ta nusxasini hosil qilish

Biz tayyorlagan bu dastur yulduzli osmonga o'xshamaydi. Yulduzli osmonni tasvirlash uchun yulduzlar har xil koordinatada joylashishi kerak. Buning uchun oynaning bo'sh joyiga sichqonchani 1 marotaba cherib **actions** oynasini ochamiz va unga quyidagi skriptni yozamiz:

```
j=1;
while (j<=50) {
    yulduz.duplicateMovieClip("yulduz"+j,j);
    _root["yulduz"+j]._x=Math.random()*550;
    _root["yulduz"+j]._y=Math.random()*400;
    j++; }
```

Endi ushbu skriptni chuqurroq qarab chiqamiz. **While** operatori bilan oldindan tanishib chiqqanmiz. *j* o'zgaruvchisiga 1 qiymatni o'zlashtiramiz. Bu o'zgaruvchi takrorlash tanasini boshqarish uchun ishlatiladi. **While** operatoriga shart qo'yilyapti. Agar *j* o'zgaruvchisining qiymati 50 dan kichik bo'lsa, takrorlash tanasi bajariladi. *j* sanagichning qiymati takrorlash oxirida 1 ta ga oshiriladi. `duplicateMovieClip("yulduz"+j,j)` yulduz obyektining "yulduz+j" nomli nusxasini yaratadi. Bu funksiyaning ikkinchi ko'rinishi. Keyingi qadamda yaratilgan nusxaning koordinatalari aniqlanilayapti. Nusxalarning koordinatalarini `setProperty` funksiyasi yordamida aniqlasa ham bo'ladi, lekin biz shu usulni tanladik. `_root["yulduz"+j]` obyektga massiv sifatida murojaat qilishni taminlaydi `_x` esa `x` o'qi bo'yicha koordinatasini aniqlaydi. `Math.random()`; funksiyasi bilan biz oldin tanishib o'tganmiz. U bizga 0 va 1 oralig'idagi tasodifiy sonni qaytaradi va olingan natija mos ravishda 550 va 400 sonlariga ko'paytiriladi (ekranning o'lchami 550x400 bo'lgani uchun). Shunday tartibda ekranga berilgan 1 dona yulduzning 50 ta nusxasi ko'rinadi. Ekranda hosil qilingan 50 ta 1 xil nusxaning ko'rinishi unchalik chiroyli bo'lmaydi. Bizning vazifamiz betartib joylashgan ushbu yulduzlarning o'lchamlarini turli xil qilib ko'rsarishdir. Buning uchun yuqoridagi skriptga takrorlash ichiga quyidagini qo'shib yozamiz:

```
aa=Math.random();
_root["yulduz"+j]._xscale-=aa*15;
_root["yulduz"+j]._yscale-=aa*15;
```

"aa" o'zgaruvchisiga [0;1] oraliqdagi tasodifiy son tanlanadi. Har yulduzning `x` va `y` bo'yicha koordinatalari `aa*15` ga kamaytiriladi. `_xscale` va `_yscale` buyruqlari bilan oldin tanishib o'tganmiz. Xullas dastur har gal ishga tushganda yulduzlar oynaning turli tuqtalariga turli o'lchamlarda ko'rinib turadi. Yulduzlarning joylashish o'rni va

o`lchamini yana boshqa kombinatsiyalarda joylashtirmoqchi bo`lsak, dasturni yana qayta yopib, ishga tushirish kerak. Bzga bu noqulay. Buni oldini olish uchun uni vaqtga bog`laymiz. Buning uchun ekrandagi ikkita tugmani olib tashlaymiz va “yulduz” nomli obyektimizni oyna chegarasidan tashqarida joylashtiramiz. Chunki keyinchalik har 2 soniyada yulduzlarning joylashish o`rni va o`lchami o`zgarib boradi, asl nusxa esa o`z joyidan jilmasdan turadi. Umumiy ko`rinishni buzmaslik uchun uni ko`rinmas holga keltirib qo`yamiz. Dasturni vaqtga bog`lash uchun Yulduzni belgilab unga quyidagi skriptni yozamiz:

```
onClipEvent (load) {
    q = getTimer(); }
onClipEvent (enterFrame) {
    a = Math.floor((getTimer()-q)/1000);
    if (a>2) {
        j = 1;
        while (j<50) {
            _root.yulduz.duplicateMovieClip("yulduz"+j, j);
            _root["yulduz"+j]._x = Math.random()*550;
            _root["yulduz"+j]._y = Math.random()*400;
            aa = Math.random();
            _root["yulduz"+j]._xscale -= aa*15;
            _root["yulduz"+j]._yscale -= aa*15;
            j++;}
        q = getTimer();}
```

Sichqoncha ko`rsatgichi o`rniga sehirli tayoqcha hosil qilish

Dastlab tayoqchanning rasmi chizilgan **movie clip** hosil qilamiz. Dastur ishga tushganda sichqonch ko`rsatgichi ko`rinmas bo`lib, uning o`rniga shu tayoqchamiz ko`rinishi kerak. Buning uchun tayoqchaga quyidagi skriptni yozamiz:

```
onClipEvent (load) {
    Mouse.hide();
}
onClipEvent (mouseMove) {
    _x=_root._xmouse;
    _y=_root._ymouse;
    updateAfterEvent();
}
```

Bu skript bilan biz oldin tanishib o`tganmiz shuning uchun unga to`xtalib o`tirmaymiz. Biz sichqonchanning o`rniga harakatlanuvchi tayoqchani hosil qildik, lekin uning ko`rinishi sehirli tayoqchani eslatmaydi. Endigi qiladigan ishimiz tayoqcha harakatlanganda o`zidan yurakchalar sachratib harakatlanishi kerak. Buning uchun o`ng tomonga 25 pikselga harakatlanuvchi yurakchanning animatsiyasini hosil qilamiz, animatsiya o`z-o`zidan namoyon bo`lmasligi uchun animatsiyaning oxirgi kadriga stop(); buyrug`ini yozib qo`yamiz. Uni oyna chegaasidan tashqarida joylashtiramiz. Yuqoridagi skriptni quyidagicha o`zgartiramiz:

```
onClipEvent (load) {
    Mouse.hide();
    i=0;
}
onClipEvent (mouseMove) {
```

```

_x=_root._xmouse;
_y=_root._ymouse;
_root.yurak.duplicateMovieClip("s"+i, i);
_root["s"+i]._x=_root._xmouse;
_root["s"+i]._y=_root._ymouse;
_root["s"+i]._rotation=Math.random()*360;
i++;
updateAfterEvent();
}

```

Bu yerda:

_root.yurak.duplicateMovieClip("s"+i, i) – yurak obyektini <s1,s2,s3...> nomli nusxalarini hosil qiladi.

_root["s"+i]._x=_root._xmouse – hosil qilingan nusxalarga sichqonchani koordinatalarini o'zlashtiradi.

root["s"+i]._rotation=Math.random()*360 – yurakning nusxasini tasodifiy burchakka buradi.

Agar nusxaga tasodifiy burilish burchagini bermasak harakatlanuvchi yurakchalar faqat 1 tomonga harakatlanadi. Shu buyruq tufayli sichqoncha harakatlanganda ko'rsatkich vazifasini bajaruvchi tayoqchadan yurakchalar ajralib chiqadi va ular har tomonga harakatlanib to'xtaydi. Lekin, dasturimizda katta xato mavjud sichqonchani biroz vaqt mobaynida harakatlantirib tursak, ekranda juda ko'p yurakchalar tasviri hosil bo'ladi va oyna yurakchalardan to'lib qoladi. Bu protsessorning normal ishlashiga to'sqinlik qiladi. Buning oldini olish uchun dastur oxiriga quyidagi skriptni qo'shib qo'yamiz:

```

if (i>=40) {i=0}
updateAfterEvent();

```

Bunda ekrandagi yulduzchalarning soni 40 tadan oshmaydi. 41- nusxa yana qaytib s1 nomi bilan ekranda namoyon bo'ladi. Bunda agar sichqonchani harakatlantirsak va harakatlantirishdan to'xtatsak, tayoqchani uchidan sachrayotgan yurakchalarning oxirgi 40 donasi har xil tomonga harakatlana boshlaydi va harakat oxirida to'xtab qoladi. Chunki oxirgi kadrda stop() buyrug'ini yozib qo'yganmiz. Bizga oxirgi sachragan yurakchalar o'z harakat trayektoriyasini bosib o'tgandan so'ng ko'rinmas holga kelishi kerak. Buning uchun yurakcha **movie clip**ini ochamiz va uning oxirgi kadridagi skriptni quyidagicha o'zgartiramiz:

```

stop();
removeMovieClip(this);

```

Bunda yulduzcha harakatlanib bo'lgandan so'ng ko'rinmas bo'ladi. Ya'ni nusxa o'chiriladi. Tayoqchadan chiqayotgan yurakchalarning rangi har xil bo'lsa, uning ko'rinishi yanada chiroyli bo'ladi. Buning uchun oynadagi yurakchaga quyidagi skriptni yozamiz:

```

onClipEvent (load) {
    a=new Color (this);
    a.setRGB(Math.random()*0xFFFFFF);
}

```

Obyektni harakatlantirish texnologiyasi

Biz yuqorida Flash obyektlarini tugmalar, klaviaturadagi klavishlar orqali boshqarishni o'rgandik. Ammo foydalanuvchi uchun qulay yana bir texnologiya mavjud bo'lib, u **drag and drop** ya'ni ekrandagi obyektning sichqoncha orqali harakatlantirishdir. Bu bir qarashda oddiy bo'lib tuyilgan jarayon, Flashning kuchli vositalaridan biridir. Bu texnologiya: interfaol interfeyslarda, o'yinlarda va qiyin dinamik effektlar hosil qilishda qo'llaniladi. Bu jarayonni oddiy misollardan boshlab ko'rib chiqamiz. Oynada qizcha rasmi tasvirlangan **movie clip**, va uy tasvirlangan tugma berilgan. Sichqoncha orqali uyni harakatlantiring. Bizga ma'lumki, ekrandagi obyektning harakatlanishi sichqoncha tugmasi bosilgandan boshlanib, qo'yib yuborilganda to'xtaydi. Shuning uchun biz quyidagilardan foydalanamiz:

on (press) – obyekt ustida sichqoncha tugmasini bosgandagi holat.

on (release) – tugmani bosib qo'yib yuborilgandagi holat.

Harakatlanadigan obyektning esa tugma deb olish kerak. Uycha rasmi tasvirlangan tugmani va bitta **movie clip**ni kutubxonadan olamiz va ekranga joylashtiramiz. Dasturga chiroyli ko'rinish berish uchun, oynaga moviy osmon va quyoshni chizamiz. Yerni esa yashil rangga bo'yaymiz. Bu yerda quyosh, yer, osmon oynaga chizilgan rasm. Uycha rasmi tasvirlangan tugmani belgilab unga quyidagi skriptni yozamiz:

```
on (press) {  
  startDrag("");  
}  
on (release) {  
  stopDrag();  
}
```



34-rasm. Movie clipni sichqoncha orqali harakatlantirish

Bunda uycha ustiga sichqoncha tugmasini bosib turib harakatlantirsak, uycha bilan birga qizcha rasmi tasvirlangan **movie clip** ham harakatlanadi. Chunki, ularning ikkisi ham bitta vaqt shkalasida joylashgan.

Skriptning har bir qismiga to'xtalib o'tamiz.

```
on (press) {  
  startDrag(""); }  
on (release) {  
  stopDrag(); }
```

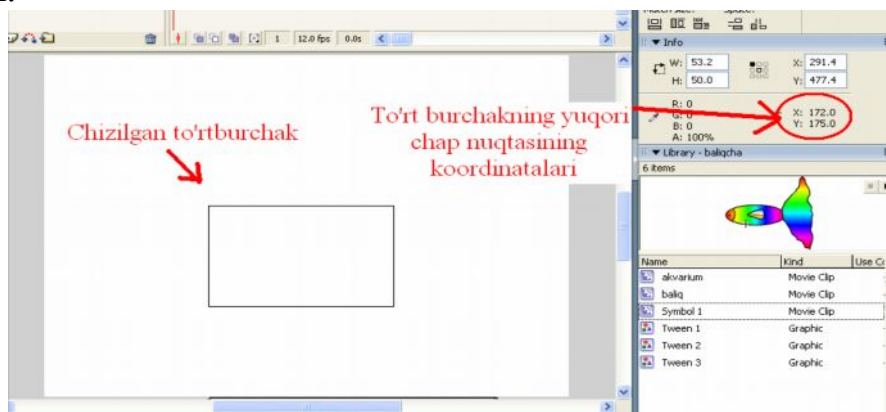
Uycha rasmi tasvirlangan tugma ustida sichqoncha bosilganda uychni sichqoncha ko'rsatgichi ortidan harakatlantiradi. Tugmani harakatlantirish uchun uning ixtiyoriy nuqtasi ustida sichqoncha tugmasini bosib siljitsak u siljiydi. Uning faqat markazidan to'rtganda harakatlanishi uchun unga yana bir parametрни qo'shishimiz kerak. Yuqoridagi skriptga o'zgartirish kiritamiz:

```
on (press) {
    startDrag("",true);}
```

Dasturni ishlatib ko'rsak, qiziq holat ro'y beradi. Uycha ustida sichqoncha bosilganda, u bilan qizchanning rasmi tasvirlangan **movie clip** sichqoncha ko'rsatgichidan ancha uzoqqa siljiydi va ko'rsatgich bilan birga harakatlanadi. Sichqoncha qo'yib yuborilganda ham tugma bilan **movie clip**, ko'rsatgich ortidan harakatni davom ettiradi. Chunki **on (release)** holatining bajarilishi uchun sichqoncha tugma ustida bosilib, so'ng qo'yib yuborilishi kerak. Rasm va **movie clip** ko'rsatgichdan uzoqda harakatlanadi, shuning uchun ko'rsatgich tugma ustiga kelolmaydi. Bunga sabab, tugma filmning asosiy shkalasida joylashgan bo'lib, uning boshlang'ich koordinatalari oynaning yuqori chap qismi bo'ladi. Tugma dastur ishga tushganda oynaning yuqori chap burchagida joylashmagan. Shuning uchun **startDrag** ishga tushganida, markazlash qismi yoqilganda ya'ni, **startDrag("",true)** bo'lganda, uycha rasmi tasvirlangan tugma oynaning yuqori chap burchagidan qancha masofaga joylashgan bo'lsa, sichqoncha ko'rsatgichiga nisbatan ham shunday masofaga siljiydi. Bu hol noqulayliklarga olib keladi. Bundan tashqari qizcha rasmi tasvirlangan **movie clip** ham tugma bilan birga harakatlanmasligi kerak. Bu noqulay effektlardan qutilish juda oddiy. Buning uchun uycha rasmi tasvirlangan tugmani asosiy diagrammadan olib tashlab, uni alohida **movie clip**ga joylashtirishimiz kerak. Tugmani **movie clip** ichiga joylashtirish uchun tugmani belgilab [F8] tugmasini bosamiz, shundan so'ng nomlab [ok] tugmasini bosamiz. Bunda tugma markazi, uning markaziy nuqtasi bilan ustma-ust tushadi. Uychaning ixtiyoriy nuqtasi ustida sichqoncha tugmasi bosilsa sichqoncha ko'rsatgichining uchi bilan uychaning markazi ustma-ust tushishiga e'tibor bering. Endi, yo'l qo'yilgan barcha kamchiliklar yo'qotildi.

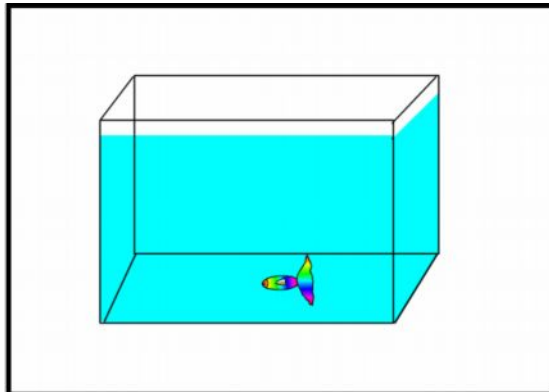
Akvarium rasmi tasvirlangan **movie clip** bilan baliq rasmi tasvirlangan **movie clip**lar berilgan. Baliqni sichqoncha orqali harakatlantiring. Bunda baliq akvarium chegarasidan tashqariga chiqmasin.

Qo'yilgan masala shartiga ko'ra, baliq ustiga sichqoncha tugmasi bosib harakatlantirilsa, baliq ham sichqoncha bilan birga harakatlanishi kerak. Agar sichqoncha tugmasi bosib turilib, ko'rsatgichi akvarium chegarasidan tashqariga chiqarilsa, baliq harak qilmasligi kerak. Buning uchun avval ekranga to'rtburchakni chizib olamiz. "Info" bo'limi orqali uning yuqori chap va quyi o'ng nuqtalarini koordinatalarini bilib olamiz. Chizilgan to'rtburchakning yuqori chap qismida sichqoncha ko'rsatgichi keltirilsa "info" bo'limida shu nuqtaning koordinatalari ko'rinib turadi.



35-rasm. Movie clipning harakatlanish sohasini aniqlash

Biz chizgan misolda to'rtburchakning yuqri chap burchagining koordinatalari x o'qi bo'yicha 171, y o'qi bo'yicha 177ga quyi o'ng burchakning koordinatalari esa, x o'qi bo'yicha 366, y o'qi bo'yicha 294 ga teng. Endi shu to'rtburchak ustiga akvariumni shunday joylashtiramizki, berilgan to'rtburchak uning ostiga qolib, ko'rinmasin. Baliq keyinchalik shu to'rtburchak ichiga harakatlanadi. Endi baliqni akvarium ichiga joylashtiraiz:



36-rasm. Harakatlanish sohasida joylashtirilgan movie clip.

Baliqni belgilab unga quyidagi skriptni kiritamiz:

```
on (press) {
    startDrag("",true, 171, 177, 366, 294); }

on (release){
    stopDrag(); }
```

Endi baliq akvarium ichida harakatlanadi. Chunki **startDrag("",true,171,177,366,294);** - baliqning harakatini to'rtburchak soha bilan chegaralaydi. Endi baliqni akvarium ichida harakatlantira olamiz. Baliq **movie clip**ini o'zini, yanada chiroyli bo'lishi uchun, harakatlantirsa ham bo'ladi. Masalan **movie clip**da uni dumini yuqoridan pastga harakatlantirish mumkin. Bunda baliq akvarium ichida muallaq suzib turganday animatsiya hosil bo'ladi.

