Animate CC 2015 HTML5 Canvas - Symboles et clips

Sommaire [masquer]

► Tutoriels Flash

1 Introduction

Objectifs d'apprentissage

- Connaitre les trois types de symboles en Flash CS3/CS4/CS5/CC
- Créer des clips (Angl. *movie clip*)
- Arrêter/démarrer des clips imbriqués avec **JavaScript** (et non pas ActionScript !)
- Concevoir un fichier Flash comme un assemblage de petits clips imbriqués, c.a.d. des animations dans des animations.

Prérequis

- Univers Flash CC
- Flash CS3 - Créer et modifier les calques et les images
- Flash CS4 - Créer des dessins avec les outils de dessin
- Flash CS4 - Transformer des dessins (une partie)
- Flash CS4 - Arranger des dessins
- Flash CS4 - Interpolation de mouvement

Matériel (fichiers *.fla à manipuler)

- http://tecfa.unige.ch/guides/flash/cc-html5/ (répertoire)

Qualité et niveau

Ce tutoriel aidera les adeptes de la technique à démarrer. Le niveau de ce tutoriel est un peu haut pour les novices, mais peut servir comme fiches pratiques dans un atelier.

Prochaines étapes

- Plus de dessin (c.f. les prérequis)

....

Autres versions

- Flash CS4 - Symboles et clips

2 Les symboles Flash

Le mot *symbol* (angl.) ou *symbole* vient de l'ancienne terminologie Flash et ne fait plus beaucoup de sens. Un symbol est un objet Flash "lourde" qui contient des définitions pour des animations, boutons et animations. En informatique, un symbole correspond à une **classe**, c-a-d. il s'agit d'un objet générique.

Il existe trois types de symboles: une définition pour (1) un graphique, (2) un bouton ou un (3) "clip" (possibilité d'animation). Un symbol est réutilisable dans le même document ou dans un autre. Chaque symbole que vous créez ou utilisez est inséré dans la bibliothèque (*library*).

Le symbole n'est jamais utilisé directement dans un clip, il faut créer une **instance**, appelé aussi **occurrence**. Seulement les instances vont être utilisé sur la scène. Un symbole peut avoir plusieurs instances, mais une instance n'est reliée qu'à un seul symbole. Vous pouvez partager les symboles entre les documents en tant que ressources de bibliothèque partagées pendant la programmation et/ou l'exécution.

Chaque symbole de type *clip* (français) ou *movie clip* (Anglais) possède son propre scénario (timeline), scène et calques. Vous pouvez y ajouter des images, des images-clés et des calques ou encore d'autres symboles, exactement comme dans le scénario principal. Autrement dit, pour faire une animation, vous pouvez ...

- Juste utiliser la timeline (scenario) principale: Dans ce cas, vous n'utilisez qu'un seul clip avec animation, c-a-d le fichier Flash lui-même. Cette façon de procéder est cher aux designers Flash traditionnels et on ne la recommande pas dans un contexte de multimédia interactif.
- Créer une animation à l'**intérieur** d'un symbole de type clip (c.f. aussi ci-dessous): Vous pourrez ensuite créer des instances de ce clip, autrement dit, poser des objets animés dans une scène. Bien entendu, ce clip peut lui aussi contenir des clips. L'avantage de cette méthode est de pouvoir facilement réutiliser des composants. Son désavantage est qu'il faut apprendre un peu de JavaScript pour lancer et arrêter des animations.

2.1 Les trois types de symboles Flash

Flash comprend trois types d'objets génériques. Dans la plupart des cas, on utilisera le symbole de type clip.

1. Les **graphic symbols** (**symboles graphiques**) représentent des images statiques et sont réutilisables. Les symboles graphiques contribuent moins à la taille du fichier FLA que les boutons et les clips car ils n'ont pas de scénario (timeline) associé. Autrement dit, il s'agit de sortes de **briques graphiques** ayant une puissance limitée.
2. Utilisez des **button symbols** (**symboles de bouton**) pour créer des boutons interactifs qui réagissent aux clics, au "toucher" ou à d'autres actions de la souris. Un symbole de bouton contient tjrs une animation image par image (voir Flash CS4 - Boutons.
3. Les **movie clip symbols** (**symboles de clip**) sont faits pour créer des éléments d'animation réutilisables. Ils possèdent leur propre scénario et ils peuvent donc contenir des contrôles interactifs, des sons, voire des occurrences d'autres clips. En outre, les clips sont programmables à l'aide de code ActionScript.

Flash Pro inclut des symboles prédéfinis:

- Les boutons que vous retrouvez dans la *library - button.fla* de Flash Profession CS3/4/5/6. Ils n'existent plus dans Flash CC, mais on peut les réimporter.
- Les composants intégrés sont des clips programmés en ActionScript (voir: Flash CS4 - Composants), et ils ne marchent donc pas pour HTML5.

Note sur la traduction: En Anglais un symbole ayant son propre scénario s'appelle **movie clip**, en français juste **clip**. En français, *clip vidéo* (traduction de *movie clip* veut dire *autre chose*, c.a.d. se réfère à une vidéo importée ....)

2.2 Créer des symboles

Vous savez probablement déjà créer des symboles, mais on répète quand même la procédure:

(1) Créer un nouveau *symbol* vide

Faites une des opérations suivantes:

- Sélectionnez menu *Insert -> New symbol*
- Cliquez sur le bouton *New symbol*, en bas à gauche du panneau *library*
- Finalement, *New symbol* dans le menu d'options du panneau *library* (en haut à droite)

Ensuite, pour l'éditer, double-clic soit sur le symbole dans la library ou sur une instance qui se trouve sur la scène. C.f. ci-dessous.

(2) Conversion d'éléments sélectionnés en symbole

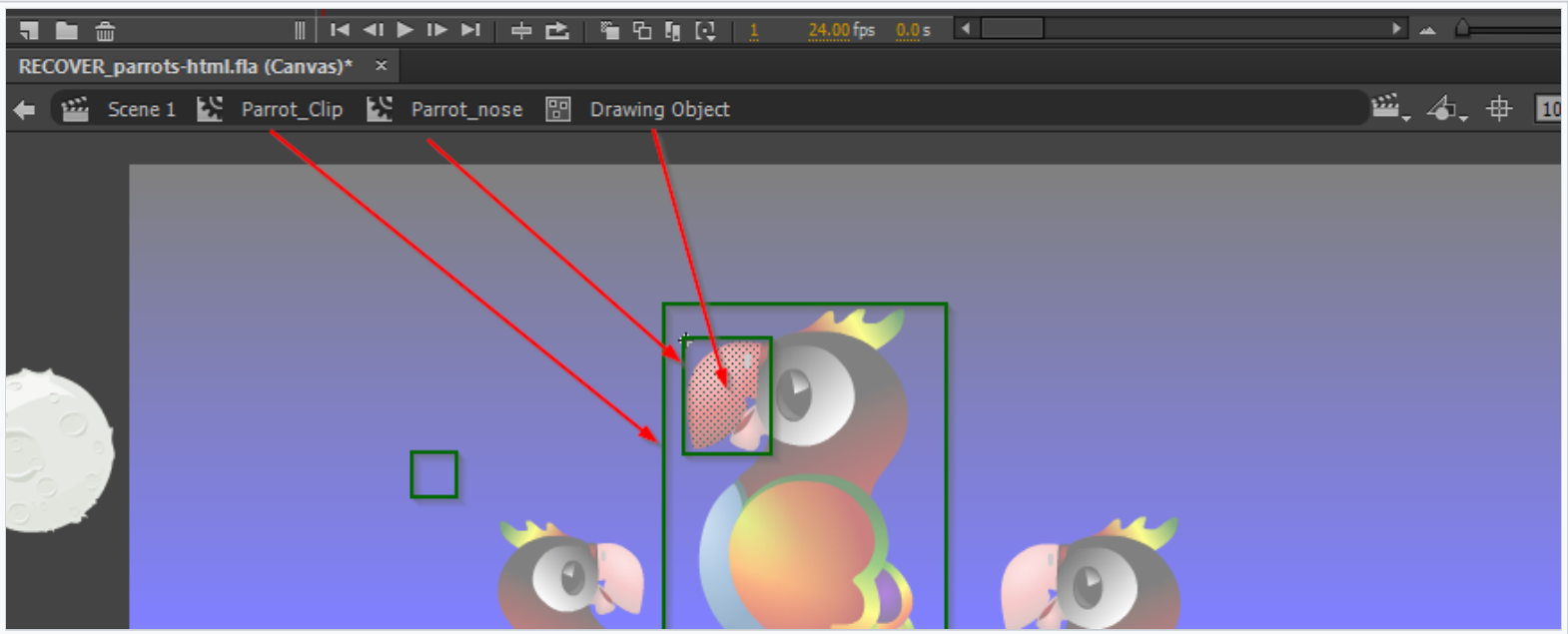
1. Sélectionnez un ou plusieurs éléments sur la scène, Effectuez l'une des opérations suivantes :
   - Cliquez du bouton droit (Windows) ou en appuyant sur la touche Contrôle (Macintosh), puis sélectionnez *Convert to symbol*
   - Menu *Modify* > *Convert to symbol* (ou utilisez le raccourci clavier F8)
   - Faites glisser la sélection dans le panneau *library* (bibliothèque)
2. Dans la boite de dialogue *Convert to symbol*, entrez le nom du symbole, puis sélectionnez son comportement, c'est à dire choisir entre *graphic symbol*, *button symbol* et *movie clip symbol*. Vous pouvez transformer un symbole plus tard, enfin si vous transformez un clip ou un bouton vous allez perdre son contenu interactif.
3. Cliquez dans la grille d'alignement afin de positionner le point d'alignement/alignement du symbole et que définit l'origine du système des coordonnés de l'objet. Par défaut (selon les conventions infographiques) il se trouve en haut à gauche. Nous préférons le centre pour les movie clip, et en haut à gauche pour les boutons et les graphiques.
4. Cliquez sur OK.

Flash ajoute le symbole à la *library*. La sélection sur la scène devient une occurrence (instance) du symbole.

2.3 Edition de symboles

Si ce n'est pas déjà fait, je suggère d'ajouter la barre d'édition: *Window-> Tool bars -> Edit bar*. Elle va vous montrer exactement à quel niveau vous éditez, par exemple au niveau de la timeline principale ou dans un symbol.

Dans l'image ci-dessous, vous pouvez voir une cascade de niveaux d'édition. Actuellement, nous sommes en train d'éditer le graphisme d'un clip appelé "Parrot_nose" !



Mode d'édition de symbole - Edition du graphisme d'un clip inséré dans un clip

Vous pouvez éditer un symbole de plusieurs façons:

- Double clic (Windows) ou *clic-droit->Edit* dans la bibliothèque (library). Cela ouvre une fenêtre d'édition où on ne voit que cet objet.
- Double clic sur une instance avec le selection tool. Dans ce cas vous verrez également les autres objets de la scène, mais ils sont "grisés", donc pas éditables. Clic droit sur une instance, puis *Edit in Place* fait la même chose
- Clic droit sur une instance. Ensuite choisir soit *Edit*, soit *Edit in place*, soit *Edit in New window* (CS5). La dernière opération ouvre une nouvelle fenêtre.

Pour s'exercer, vous pouvez ouvrir le *.fla suivant et remplacer la couleur du nez des perroquets.

- parrots-html.fla

Important: lors de la création de vos symboles (même si très basiques comme des carrés), vous devez impérativement placer les **points d'alignement de façon réfléchie**, surtout si vous voulez faire apparaitre des instances de cet objet via JaavaScript et les positionner finement. Deux stratégies sont conseillées: la première est celle de centrer le point par rapport à l'image (si un symbole fait 50x100px, pour le centrer vous devez le positionner en x: -25, y: -50, donc la moitié de la hauteur et de la largeur), l'autre stratégie est celle de déterminer un coin (soit en haut à gauche ou à droite; dans les deux cas on aura la position x: 0, et puis soit y:0, soit y:100, si la hauteur du symbole est de 100px). Garder une approche systématique lors de la création des symboles facilitera votre travail plus tard. Pour changer le point d'alignement, c.f. ci-dessous.

2.4 Attention

A tout moment veuillez bien faire attention à quel niveau vous éditez ! Editer un objet (par exemple ajouter des dessins) alors que vous croyez travailler sur la timeline (scénario) principale est assez désastreux ...

Pour revenir à un niveau supérieur, vous pouvez

- Cliquer sur le nom de la **scène** (séquence) dans la barre Modification.
- Cliquer sur la flèche "précédent" 
- Menu *Edit -> Modify document*

3 Les clips

3.1 Création d'un clip avec animation interne

Examinons maintenant une façon de créer des clips d'animation Flash dans les fichiers flash.

Pour créer un nouveau clip dans un fichier d'animation Flash, nous rappelons qu'on peut procéder comme suit:

- Menu *Insert-> New symbol*
- Selectionner *Movie Clip* et de donner un bon nom. On suggère de donner un nom comme "Clip XXX" à un objet qui contiendra une animation.

Sinon, vous pouvez aussi transformer un graphique existant en clip

- Dessiner un objet sur la scène
- *Clic-droit-> Convert to symbol*. Sélectionnez Movie Clip'

3.2 Point d'alignement et point de transformation

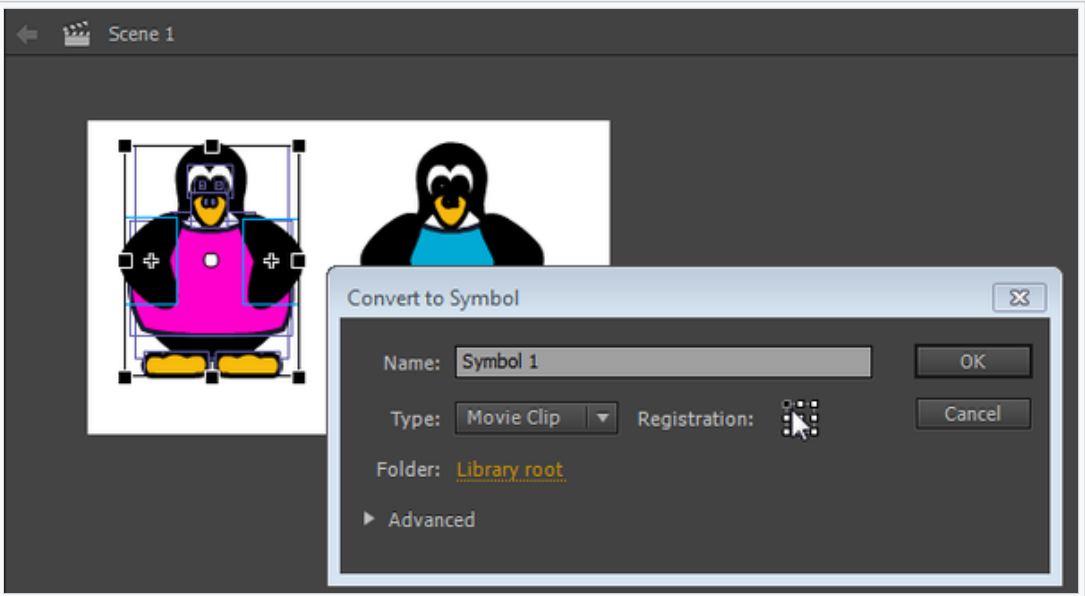
Les points d'alignement

Aussi appelé **point d'enregistrement** (traduction directe de l'Anglais ***registration point***)

Comme on l'a déjà expliqué, chaque clip a un *point d'alignement* qui définit à quel endroit du dessin correspondent les points de coordonnées x=0 et y=0 (par exemple, supérieur gauche ou milieu). Lorsqu'on positionne un objet (par exemple dans une animation) on le positionnent par rapport à ce point. Ce point d'alignement est affichée par une petite croix. Vous pouvez changer ce point d'alignement en déplaçant le dessin à un autre endroit. Il suffit d'éditer le *symbol* (double clic sur une instance ou encore sur le symbole dans la *library*).

- Modifier l'objet dans **la librairie** (!). Puis le bouger ou mieux, utiliser le panneau "Properties": Selectionner le tout (CTRL-A) puis mettre x=0 et y=0 si l'alignement doit être en haut à gauche (standard pour tout graphisme 2D). La croix au centre (+) que l'on voit sur le graphisme représente le point d'alignement.

Note: Le point d'alignement (le centre de ses propres coordonnées) n'est pas la même chose que son *point de transformation* que vous pouvez déplacer avec le "Free Transform tool", c.f. en bas)



Convertir un groupe de graphiques en clip - le point d'alignement est au centre

Les points de transformation

Le cercle blanc que vous pouvez ensuite voir au milieu d'un clip est appelé le point de transformation ou point de rotation ou point cetnral (**Center Point**) et il a une fonction similaire. Il définit l'endroit où l'objet sera attaché à un guide de mouvement autour duquel il tourne. Vous pouvez le déplacer vers un autre endroit avec l'outil de transformation libre (*Free transform tool*). Par défaut, le point de transformation est au même endroit que le point d'alignement. Pour ramener un point de transformation au point d'alignement il faut double-cliquer dessus.

3.3 Modification d'un clip d'animation

Rappelons qu'il existe deux manières de modifier un clip:

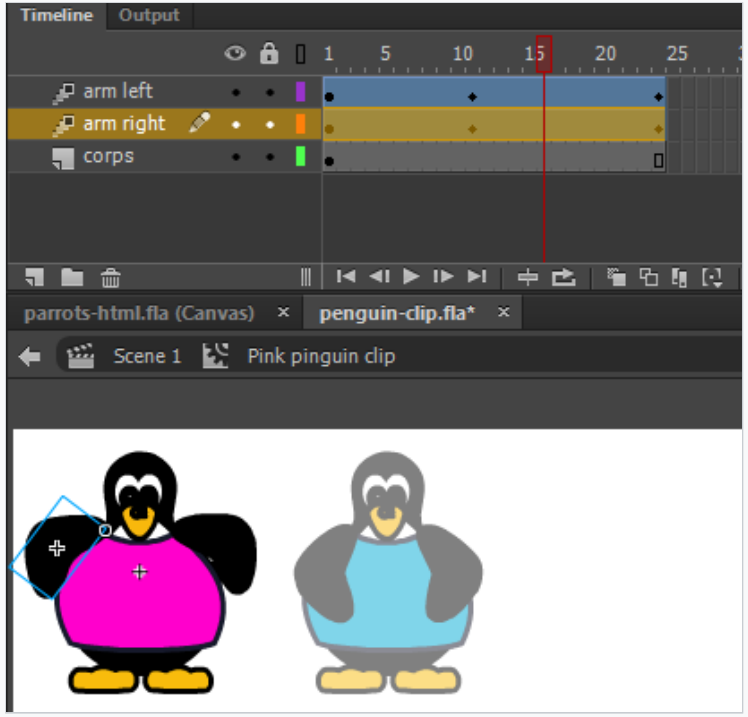
(1) En vue "stand-alone" vous pourrez voir uniquement les composantes du clip. Double-cliquez sur l'"icône clip" (pas son nom) **dans la bibliothèque**. Vous pouvez maintenant travailler avec sa chronologie (timeline) interne, donc créer une animation interne à l'objet.

(2) Modifier un objet avec la scène en arrière-plan. Si vous placez une instance du clip sur la scène et puis double-cliquez sur cette instance, vous pouvez modifier le clip tout en voyant les objets de la scène alors que vous éditez juste le clip. Ci-dessous est une copie d'écran d'une situation où nous éditons un symbole de clip dans un contexte. La scène elle-même est "grisée" (image et boutons).

Dans un clip vous pouvez faire ce que vous appris dans les tutoriels précédents, p.ex. dans le Flash CS4 - Interpolation de mouvement. En d'autres termes, les clips ont leur propre scénario comme vous pouvez le voir sur l'image ci-dessus. L'exemple ci-dessous implémente une animation de type **motion tween**

pour un pingouin, c'est à dire on fait bouger son bras droit. Dans la copie d'écran ci-dessous en est en train d'éditer le "Pink pinguin clip". Vous pouvez voir deux interpolations de mouvement. Le bras droit en train d'être édité a son **center point** en haut à droit pour pouvoir le tourner.

N'oubliez pas de revenir à la scène une fois que vous avez fini, par exemple en double-cliquant sur la "scène". Assurez-vous que vous savez à quel niveau vous modifiez et où placer les objets!



Pink pinguin clip arm movement animation

Fichiers:

- Animation (voir d'abord)
- penguin-clip.fla (fichier *.fla)
- http://tecfa.unige.ch/guides/flash/cc-html5/embedded-movie-clips/ (répertoire)

3.4 Insérer une animation existante dans un nouveau document

Il existe des situations où on désire juste copier un ou plusieurs calques (layers) et qui contiennent des animations. C'est facile à condition d'utiliser le menu contextuel (clic droit).

Créez un nouveau symbole de type "movie clip" (Menu *Insert->New Symbol* ou CTRL-F8).

Allez sur votre animation déjà réalisée et sélectionnez tous les *layers* dans la timeline. Pressez la touche "CTRL" avant de sélectionner avec la souris. Il est également possible de ne copier que certains frames. Ensuite clic-droit sur cette sélection et *copy frames* et coller tout dans votre nouvelle animation en utilisant de nouveau le menu contextuel (clic droit) *paste frames*. Puis glisser ce nouveau clip dans le layer/frame approprié du nouveau fichier Flash.

4 Pilotage de clips avec JavaScript

4.1 Pilotage d'un clip - survol

Créez un clip ou faites glisser un clip à partir de la bibliothèque sur la scène et donnez-lui tout-de-suite **un nom d'instance**. Rappelez-vous que les noms d'instance doivent être simple, par exemple books et *pas* quelque chose comme *animation géniale de livres*. Un nom d'instance doit commencer par une lettre, suivi par d'autres lettres, chiffres ou le signe "_" seulement.

Ensuite, il faut créer un bouton. N'importe quel symbole peut faire office de "button", y compris l'objet animé lui-même (lisez le tutoriel Animate CC 2015 HTML5 Canvas - Boutons. Le bouton sera utilisé pour déclencher une fonctions une fois que l'utilisateur clic dessus:

```
this.monBouton.addEventListener("click", startAnimation.bind(this));
```

Ensuite il faut définir la fonction `startAnimation`. Supposons que le nom de l'instance d'un clip soit **books**. Vous pouvez maintenant utiliser du code JavaScript pour manipuler ce clip depuis cette fonction. Par exemple:

Jouer un clip

```
this.books.play();
```

Pour arrêter un clip

```
this.books.stop();
```

Rendre visible ou invisible un clip

```
this.books.visible = false;
this.books.visible = true;
```

Ci-dessous, il s'agit de code exemple. Donc si votre instance s'appelle pingu, il faut changer, par ex:

```
this.pingu.play ();
```

Astuce: Les clips vidéo commencent à jouer dès qu'ils se trouvent dans l'image (**frame**) courante. Par exemple si vous les mettez dans dans un frame qui s'étale sur plusieurs frames, ils vont continuer à jouer jusqu'à ce que l'exécution du scénario principal tombe sur une autre image clef (**keyframe**) dans le même layer. Si vous préférez que le clip soit arrêté lors du chargement:

- Modifier le clip (double clic)
- Ajouter un calque "Actions" si nécessaire
- Ajouter cette méthode JavaScript dans le frame 1.

```
this.stop ();
```

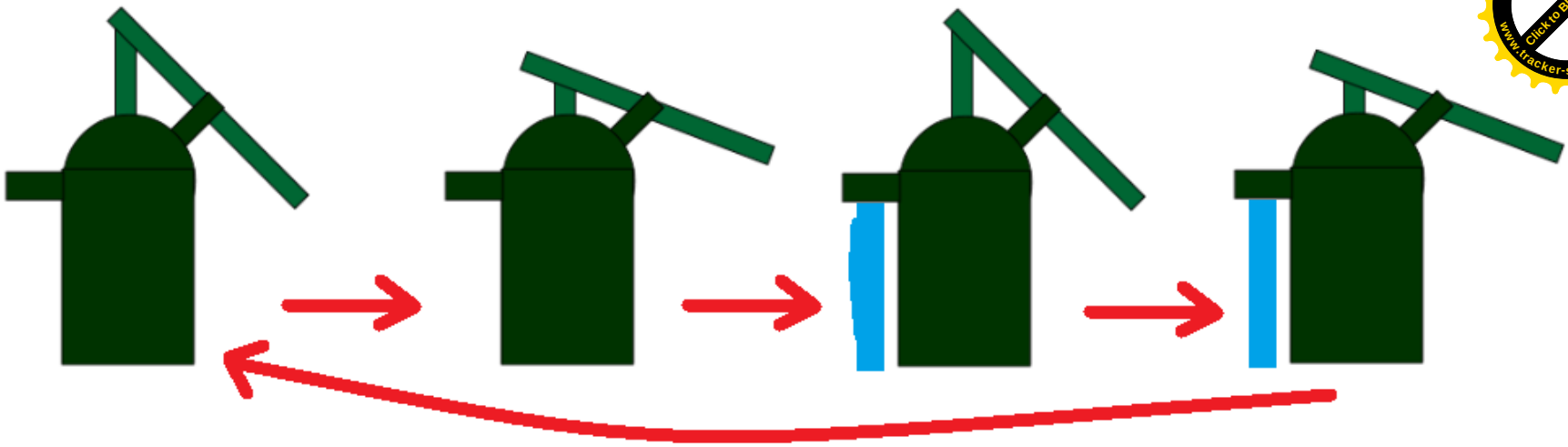
Pour plus de détails, veuillez consulter les exemple (y compris leur code source) ci-dessous.

4.2 Lancer un clip après x temps

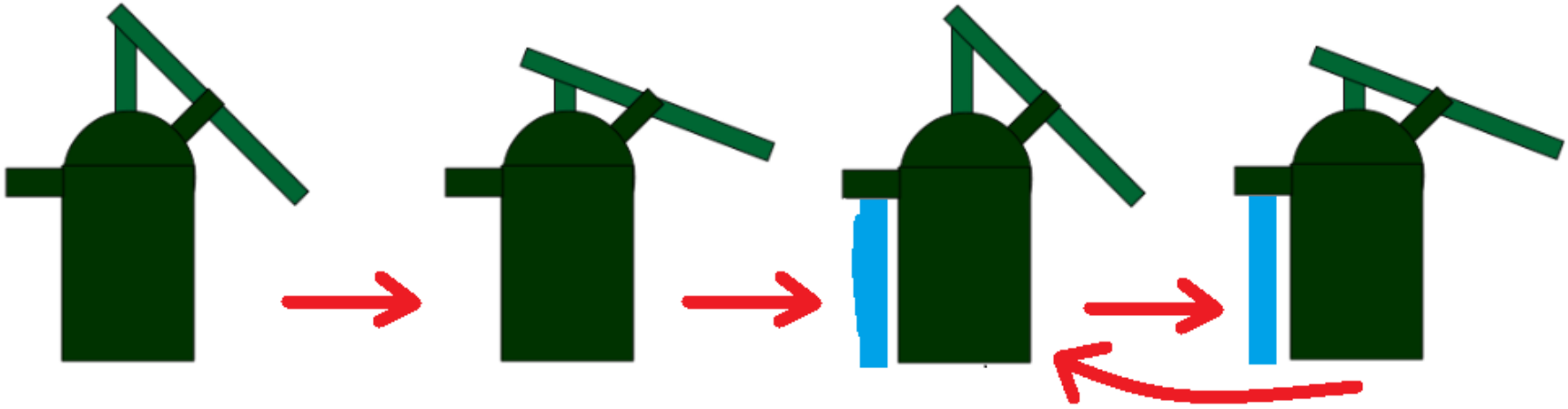
Lorsque vous avez des animations qui commencent à devenir complexes comme des cinématiques il peut devenir difficile de la faire exécuter sur un seul clip. Il est alors parfois plus simple de construire des bouts d'animation séparément et de les intégrer par la suite. Imaginons que nous voulons faire l'animation d'une fontaine à levier. Nous voulons que de l'eau sorte un moment après avoir commencé à actionner le levier. En résumé la cinématique se passe en boucle en deux étapes:

1. la pompe est actionnée (l'eau ne sort pas).
2. la pompe est toujours actionnée et l'eau sort.

Si on faisait ça dans un seul clip alors ces deux étapes seraient répétés en boucle et donc dans l'animation il y aurait des moments où l'eau s'arrête de sortir.



Or nous voulons qu'une fois que l'eau a commencé à sortir cela ne s'arrête plus d'où l'utilité d'avoir fait deux animations (une pour la fontaine et une autre pour l'eau).



Il va donc falloir faire tourner les deux animations en boucle mais que l'animation de l'eau commence à tourner quelques secondes plus tard après que l'animation de la fontaine ait commencé.

Procédure:

- Créer l'animation (clip) de la fontaine
- Créer l'animation de l'eau
- Insérer les deux animations dans la scène principale et leur donner un nom (fontaine et eau).

Une fois ces trois étapes exécutés nous allons devoir ajouter du code afin de lancer l'animation de l'eau après x secondes. Il faudra donc écrire du code (f9 ou fn+f9). Voici la fonction qu'il faudra insérer au début de votre code:

```
fairePause = function(symbole, visible, secs) {
    symbole.stop();
    if(visible == false){
        symbole.visible=false;
    }
    var intID = setInterval(lancer, secs * 1000, symbole);
    function lancer(symbole) {
        clearInterval(intID);
        if(visible == false){
            symbole.visible=true;
        }
        symbole.play();
    }
};
```

Cette fonction permet de lancer une animation après x secondes et de dire si l'animation doit rester invisible ou pas tant qu'elle n'a pas été lancée. Ensuite il suffira juste de lancer l'animation de la fontaine et celle de l'eau en utilisant la fonction fairePause. Il faudra plus que copier à la suite le code suivant à la suite:

```
this.fontaine.play();
fairePause(this.eau, false, 3);
```

Ici l'animation de l'eau sera lancée après 3 secondes et ne s'affichera pas avant les 3 secondes. Si on veut qu'elle s'affiche dès le chargement de la page il faudra mettre "true" à la place de "false".

4.3 Répéter un clip x fois

Par défaut, lorsque l'on créé une animation celle-ci se répète en boucle. Il serait parfois utile de définir combien de fois nous voulons répéter une animation.

Procédure:

- Créer votre animation (appelée rectangle dans l'exemple)
- Insérer votre animation dans la scène principale

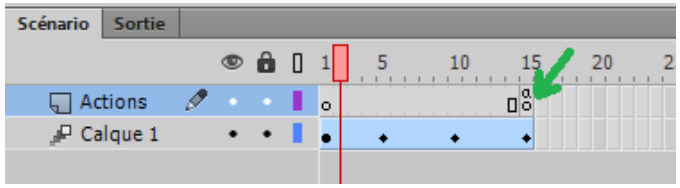
Cas 1: Vous voulez que votre animation se répète en boucle Tout est bon, vous n'avez plus rien à faire.

Cas 2: Vous voulez que votre animation se répète une seule fois

- Ouvrir votre animation
- Créer un nouveau calque (appelé Actions dans l'exemple)
- Insérer le code suivant dans le nouveau calque (f9 ou fn+f9):

```
this.stop();
```

- Placer ce code à la fin de l'animation:



Quand l'animation arrivera au frame 15 alors le this.stop(); s'exécutera et donc l'animation ne sera plus répétée. Au final l'animation sera répétée qu'une seule fois.

Cas 3: Vous voulez que votre animation se répète x fois

- Sur votre scène principale, créer un nouveau calque (appelé Actions dans l'exemple)
- Insérer le code suivant dans le nouveau calque (f9 ou fn+f9):

```
compteurAnimationRectangle = 0;
//Sert à dire combien de fois on va répéter l'animation
NbFoisAnimationRectangle = 3;
this.rectangle.play();
```

Il faudra créer une variable compteur et NbFois pour chaque animation que vous voudrez répéter un nombre particulier de fois. La variable compteur doit être insérée au début de votre code et devra toujours valoir 0. La variable NbFois vaudra le nombre de fois que l'on veut répéter l'animation (3 dans l'exemple). Cette variable sera placée avant l'instruction this.play(); de l'animation en question (rectangle dans l'exemple).

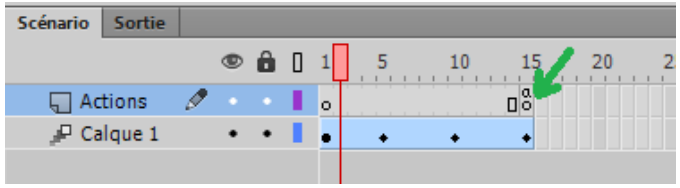
- Ouvrir votre animation
- Créer un nouveau calque (appelé Actions dans l'exemple)

- Insérer le code suivant dans le nouveau calque (f9 ou fn+f9):

```
if(compteurAnimationRectangle >= 0){
        compteurAnimationRectangle++;
}


if(compteurAnimationRectangle == NbFoisAnimationRectangle){
    this.stop();
    //this.visible=false;
    compteurAnimationRectangle = 0;
}
```

- Placer ce code à la fin de l'animation:



A chaque fois l'animation arrivera au frame 15 alors le compteur compteurAnimationRectangle sera incrémenté. Lorsque celui-ci vaudra la valeur que nous avons indiqué que nous voulons que l'animation soit répétée grâce à notre variable NbFoisAnimationRectangle alors this.stop(); sera exécuté et donc l'animation ne sera plus répétée. Au final l'animation sera répétée NbFoisAnimationRectangle fois. Si vous voulez qu'à la fin des NbFoisAnimationRectangle répétitions votre animation disparaisse vous pouvez enlever les "//" devant "this.visible=false;".

4.4 Faire un effet de mouseover

L'exemple de code ci-dessous montre comment faire un effet de mouseover. Lorsque le curseur de la souris passe sur une forme, dans notre cas il s'agit d'une forme jaune (jaune1 dans le code), un textbox s'affiche contenant du texte faisant référence à cette forme. Il faut tout d'abord dire à la scène d'activer le mode mouse over, puis ouvrir un évènement (addEventListener) sur la forme souhaitée. Cette méthode va récupéré l'action de l'utilisateur lorsqu'il passe la souris sur la forme et appeller une fonction qui rend visible ou invisible le textbox.

```
// activation du mode mouseove
stage.enableMouseOver(30);
//application du eventListener sur la forme jaune pour appel à la méthode qui permet de faire appraître le texte
contenu dans le textbox
```

```
this.jaune1.addEventListener("mouseover", seeTextJaune.bind(this));
//application du eventListener sur la forme jaune pour appel à la méthode qui permet de masquer le texte cont
dans le textbox
this.jaune1.addEventListener("mouseout", hideTextJaune.bind(this));
```

```
// déclaration de la fonction qui joue avec l'attribut visible du textbox pour le rendre visible
function seeTextJaune() {
 this.txtJaune.visible = true;
}
// déclaration de la fonction qui joue avec l'attribut visible du textbox pour le rendre invisible
function hideTextJaune() {
 this.txtJaune.visible = false;
}
```

4.5 Transformer une animation de la timeline principale en clip

Supposons que vous avez une animation intéressante dans un fichier *.fla que vous désirez utiliser comme un clip intégré.

Étape 1 - Créez un nouveau symbole de clip dans un nouveau fichier

(voir ci-dessus)

- Supposons que le fichier est appelé clips.fla
- Menu *Insert-> New symbol*
- Select *Movie Clip* et donnez un bon nom

Vous devriez maintenant voir une scène vide, car vous êtes en mode d'édition de ce nouveau symbole.

Étape 2 - Copier les images à partir du fichier *. fla

- Maintenant ouvrez le fichier FLA avec la timeline animée. Supposons que le fichier est appelé anim.fla
- Sélectionner tous les calques et les cadres que vous désirez récupérer (par exemple *avec un clic dans le premier layer/frame* et un *shift-clic dans le dernier layer/frame*). Si vous voulez récupérer le tout, c'est simple: Shift-clic sur tous les layers (à gauche de la timeline)
- Ensuite *Clic droit* quelque part dans cette sélection et **Copy Frames**.
- Maintenant retournez au fichier clips.fla ouvert (vous devriez toujours être dans le mode édition de symbole).
- Cliquer dans la frame 1 du calque 1 de ce clip encore vide
- Puir clic-droit et **Paste Frames**.

Voilà, vous avez maintenant un clip avec une animation. Retour à la scène....

4.6 Exemple flying kite (CC 2014.2 / HTML)

Cet exemple montre comment jouer et arrêter un clip embarqué avec un bouton. Regardez d'abord l'application kite pour avoir une idée.

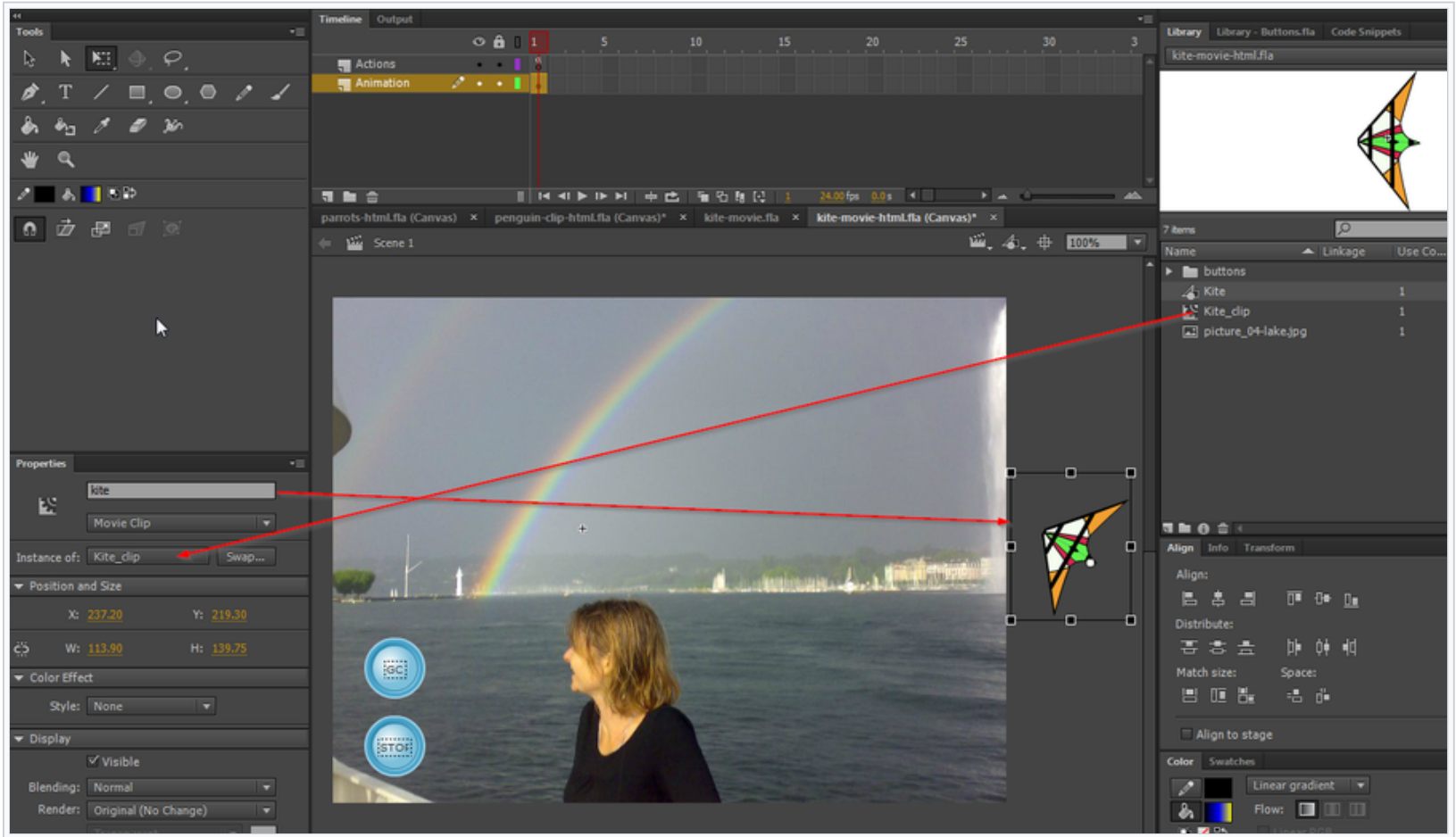
Étape 1 - Créez le clip qui va bouger

- Créez un clip (CTRL-F8, c.f. aussi ci-dessus). Dans l'exemple il s'appelle `Kite clip`. Il est important de ne pas créer une animation dans la main timeline !
- Insérez un dessin dans ce nouveau clip.
- Transformez le dessin dans le clip en symbole (clip ou graphisme). Dans l'exemple c'est un graphisme qui s'appelle `Kite`. Sinon, on ne peut pas l'animer.
- Donc, pour finir, vous avez un clip pour l'animation et un symbole (graphisme ou autre clip) inséré dans le premier clip.

Étape 2 - Si nécessaire, glissez le clip sur la scène et donnez lui un nom d'instance

- De la bibliothèque, glissez le clip sur la scène
- Dans le panneau de propriétés, donnez-lui un nom, par exemple `kite`

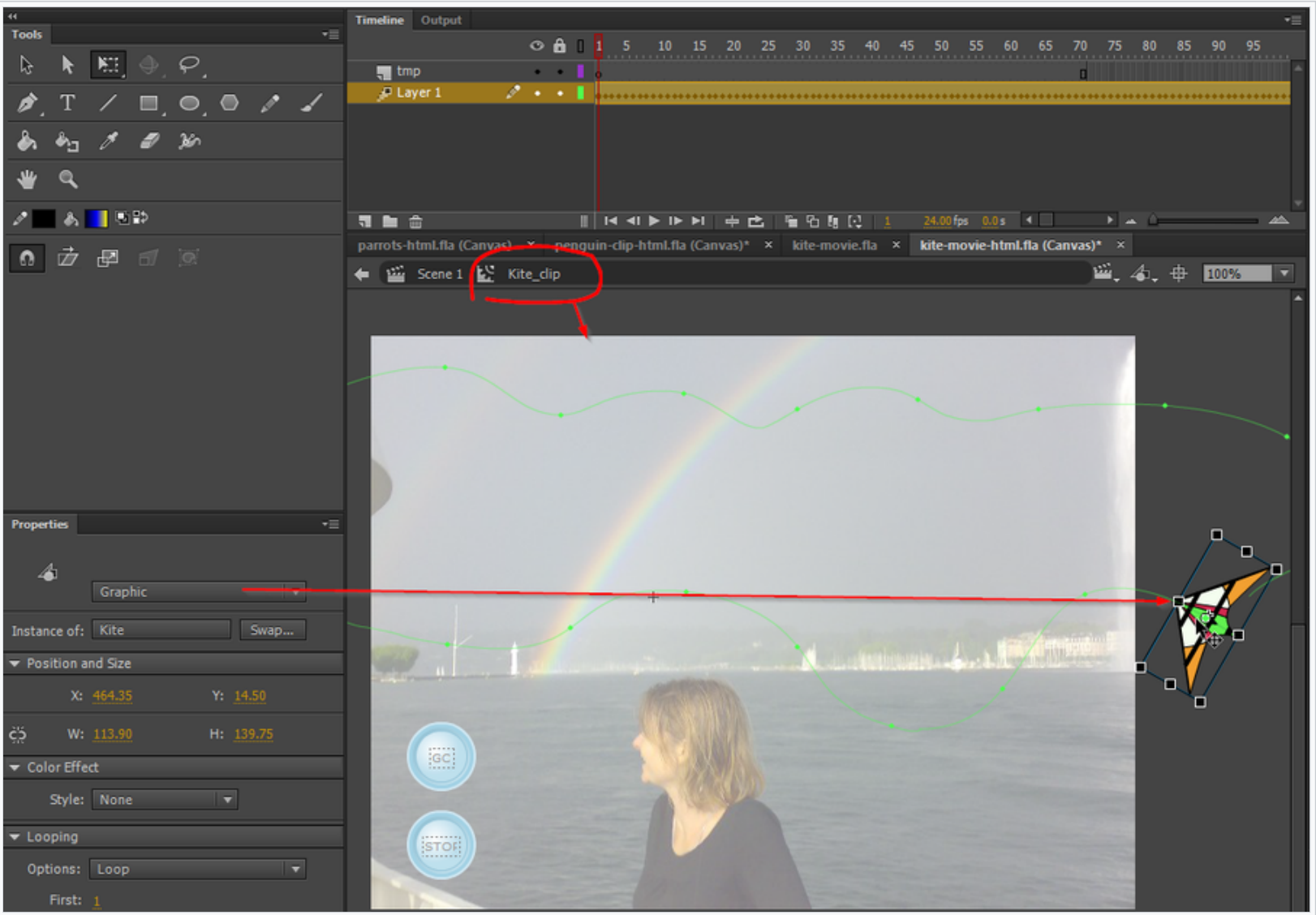
Vous devriez voir quelque chose comme sur la capture d'écran suivante:



Un clip "kite" sur la scène

Étape 3 - Créer une interpolation de mouvement guidés pour le clip

- Double-cliquez sur l'instance du clip
- Ensuite, procédez comme expliqué dans les tutoriels sur l'animation, c-a-d. ajoutez une animation de mouvement *dans la timeline du clip* (et non pas sur la timeline principale).
- etc..



Edition d'un un clip "kite" sur la scène

Étape 4 - Ajouter des boutons

- Retournez à la scène
- Nous utilisons deux boutons
- Chacun de ces boutons a un nom d'instance: *stopBtn* et *goBtn*

Étape 5 - Ajoutez le code JavaScript

- Au début, on arrête le clip avec `this.kite.stop();`. Sinon le clip pour le cerf-volant sera automatiquement démarré.
- Un clic sur le goBtn lancera la fonction startIt lorsque l'utilisateur clique dessus. Cette fonction va alors exécuter *this.kite.play ()*.
- Le bouton stopBtn est programmé de la même manière.

```
// Stop the kite clip from running
this.kite.stop();

// registers the startIt function to be called when user clicks on goBtn
this.goBtn.addEventListener("click", startIt.bind(this));

function startIt()
{
    // Start the kite clip
    // uncomment following line for debugging
    console.log ("goBtn pressed" + ", kite=" + this.kite);
    this.kite.play();
}

// registers the stopIt function to be called when user clicks on stopBtn
this.stopBtn.addEventListener("click", stopIt.bind(this));

function stopIt()
{
    // Stop the kite clip
    // console.log ("goBtn pressed");
    this.kite.stop();
}
```

Exemple

kite-movie-html.html

Source: kite-movie-html.fla

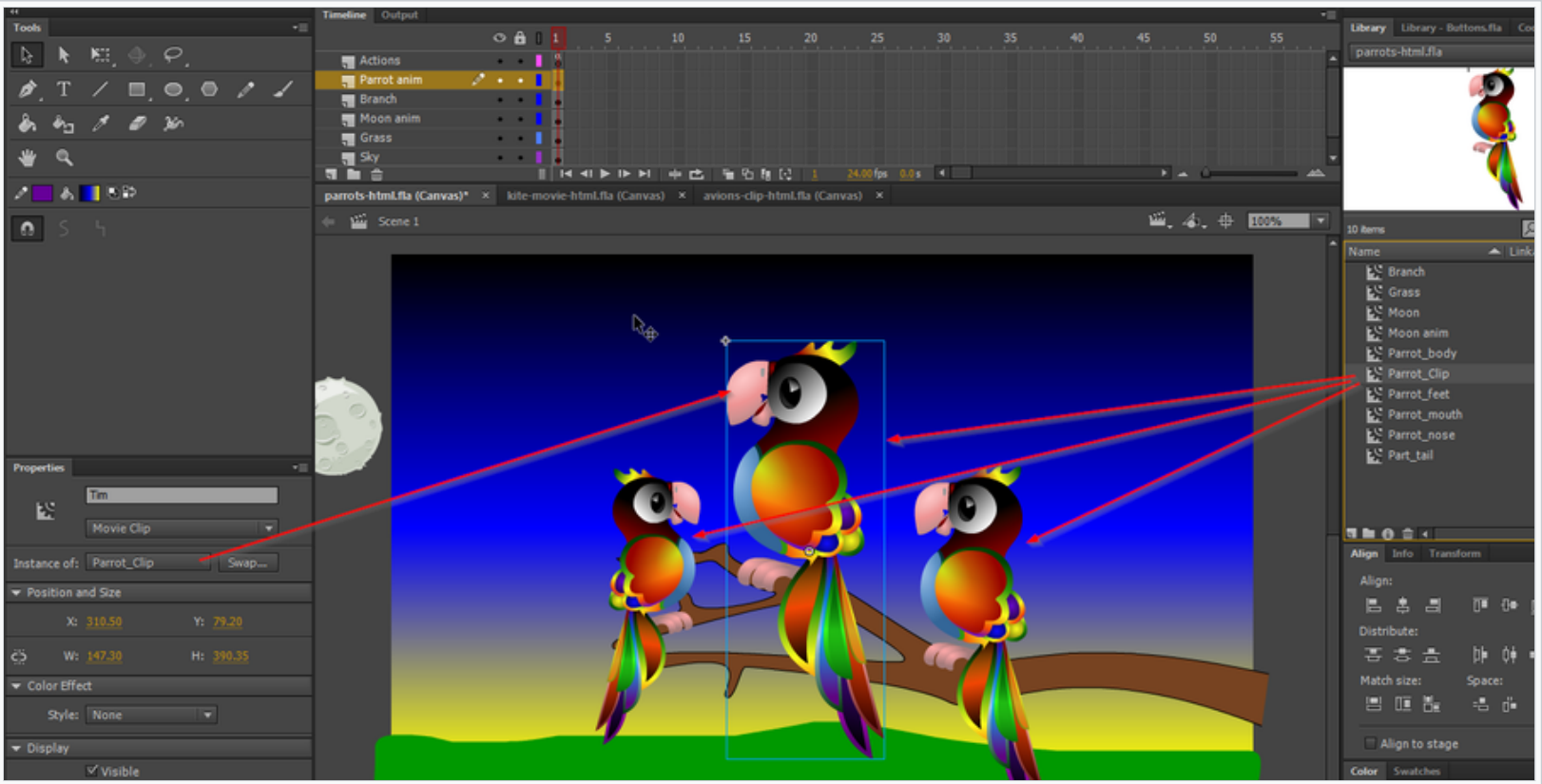
Répertoire: http://tecfa.unige.ch/guides/flash/cc-html5/embedded-movie-clips/

Notice: Il existe aussi une version Flash.

Exercice: Ajouter une autre animation clip. Par exemple un alien volant.

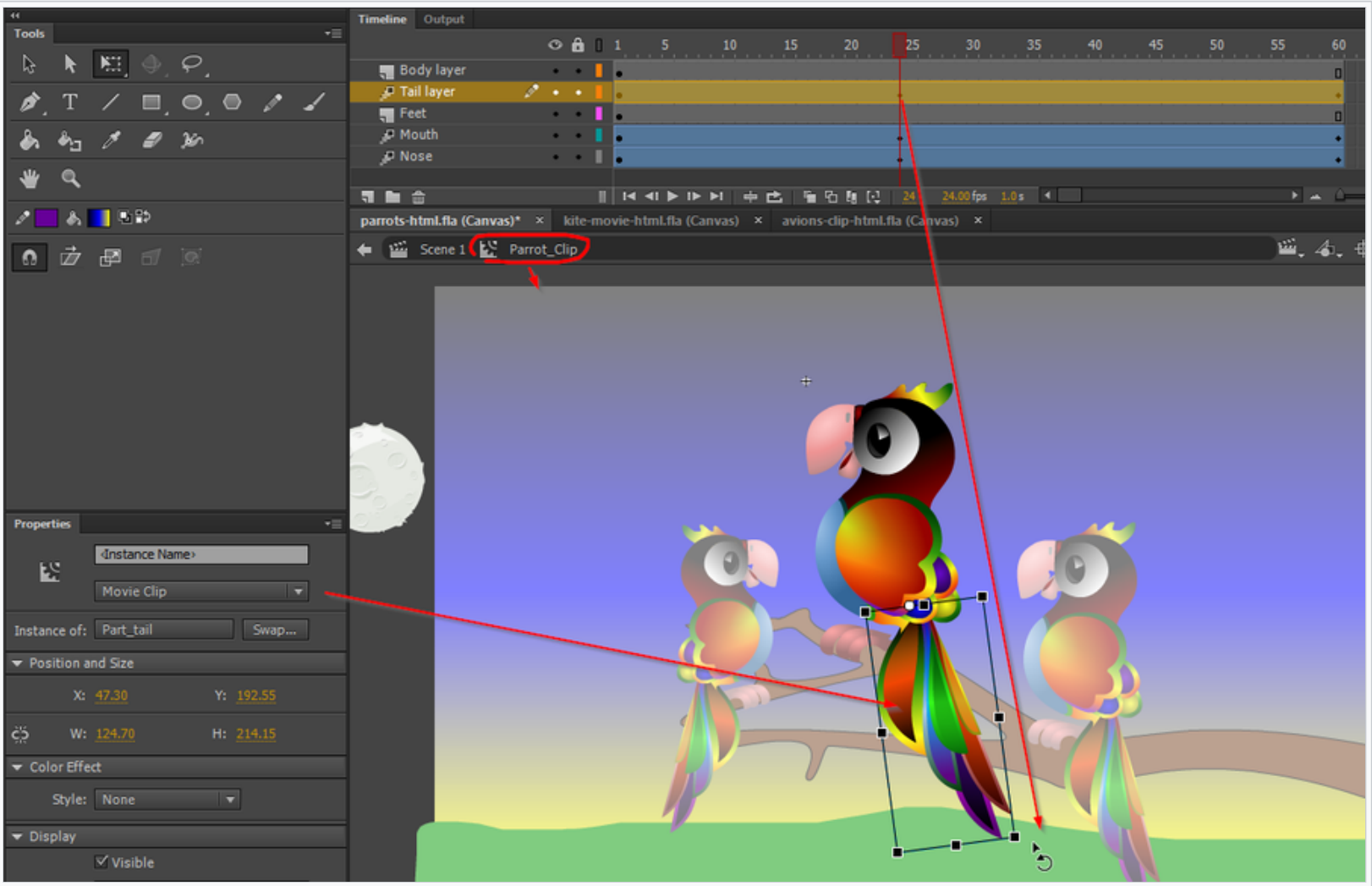
4.7 Exemple Perroquets

- Dans cet exemple, on crée 3 instances du même clip.
- Ensuite pour lancer les animations, on transforme chacun en bouton. Donc en cliquant sur un peroquet on le reveille.



3 clips perroquets

Le clip perroquet a trois animations de mouvement, c'est-à-dire 2 rotations pour la bouche et une pour la queue



3 clips perroquets

Ces animations sont déclenchés par des clics sur les instances. On utilise deux méthodes différentes.

Fichiers:

- parrots-html.fla
- parrots-html.html

Sources clip art sur OpenClipart.org

- https://openclipart.org/detail/193812/parrot-remix
- https://openclipart.org/detail/190243/doves-on-a-branch-for-v-day

4.8 Exemple avion avec 2-3 niveaux d'imbrication de symboles

Le principe est simple: On crée une bibliothèque de clips animés. Ensuite on peut combiner de différentes manières. Par exemple: (a) créer des bouts d'animation, (b) créer des clips qui les utilisent, (c) créer une instance dans la timeline.

Exemple:

- Clip A = Dessin hélice
- Clip B = Dessin avion
- Clip C = Clip hélice qui tourne
- Clip D = Avion qui contient B et 2 fois C
- Clip E = Animation avion qui bouge et qui contient deux fois D

Sources:

- avions-clip-html.fla (source CC clip)
- avions-clip-html.html

4.9 Aide à la programmation avec les code snippets

Animate CC dispose d'une aide pour programmer les interactions selon le langage employé. On peut accéder à ces aides depuis le menu `Window` > `Code Snippets`.

Pour illustrer ces aides, nous allons arrêter une animation à un frame :

1. Positionez-vous au frame où vous souhaitez arrêter l'animation
2. Puisque nous concevons une animation en HTML5/JS, aller sur le menu `HTML5 Canvas` > `Timeline Navigation` > `Stop at this frame`.

5 Créer des instances de clip avec JavaScript

Je ne sais pas s'il existe un mécanisme similaire à celui de ActionScript (voir Utilisation d'un clip de la library dans AS3 - Survol du langage ActionScript)

En attendant que je trouve une solution plus élégante, voici une procédure qui marche:

En résumé:

- Placez une instance sur la scène (Angl **stage**) et mettez la en dehors de la plage d'affichage. Cela force Flash à exporter le symbole en tant que classe, lorsqu'il génère le code JavaScript pour CreateJs. Mais **attention**, ne rendez pas l'objet invisible car il ne sera pas exporté. Je répète, évitez `this.myInstance.visible = false;`. A mon avis il s'agit d'un bug.

- Ensuite dans le code ActionScript, vous trouvez la classe à instancier comme propriété de la variable `lib`

```
var newObj = new lib.myClip();
```

par exemple
`var personne = new Joe();`

Dans l'exemple suivant, on stocke le résultat - une nouvelle instance de `lib.Parrot_Clip` - dans une propriété de this.

```
this.newBird = new lib.Parrot_Clip();
```

Exemple générateur de Perroquets (cliquer sur la branche):

- File: parrot-creation-html.html (exemple live)
- File: parrot-creation-html.fla (source)

```
// Clipart
// Parrot: https://openclipart.org/detail/193812/parrot-remix
// Branch: https://openclipart.org/detail/190243/doves-on-a-branch-for-v-day

// It seems that library objects cannot be exported.
// I therefore created an instance and had a look at the JS code.
this.Joe.visible = false;

this.createBtn.addEventListener("click", createBird.bind(this));

var counter = 1;

function createBird(ev)
{
    //Notice the lib.something for creating an instance !!
        this.newBird = new lib.Parrot_Clip();
    // position the thing
    this.newBird.x = 180 + counter * 30;
    this.newBird.y = 60 + counter * 8;
    // add an event handler for hiding
    this.newBird.on ("click", hideBird)
    // add it to the stage
```

```
    stage.addChild(this.newBird);

    //console.log ("newBird=" + this.newBird + " at x=" + this.newBird.x);

    counter++;
}

function hideBird ()
{
    this.visible = false;
}
```

PDF-XChange Editor
Click to BUY NOW!
www.tracker-software.com