

Flash Catalyst par l'exemple n°1

publié le 10 juin 2011

PRE-REQUIS

connaissance:

Une première expérience avec Catalyst est fortement conseillée. On considérera ainsi comme acquis la compréhension générale du fonctionnement de Catalyst et des différents panneaux qui le composent. Catalyst s'appuie massivement sur la notion d'état, il est donc préférable de connaître un minimum sur cette notion telle qu'elle est implémentée dans Flex/Flash Catalyst pour profiter pleinement de ce tutoriel.

niveau utilisateur:

Ce tutoriel s'adresse à des utilisateurs de niveau intermédiaire. Les concepteurs ont voulu un outil facile à prendre en main, c'est pourquoi je pense qu'un débutant curieux pourrait malgré tout s'en sortir. Son utilisation deviendra notamment rapidement très familier aux utilisateurs de la suite Adobe et en particulier Illustrator ou Photoshop. Au cours de ce tutoriel, nous ne nous appuyerons pas sur le code généré pour ne pas trop alourdir ce tutoriel. Je pense malgré tout que c'est un bon réflexe à prendre pour les devigners (comprendre développeur-designer) auxquels le logiciel s'adresse en particulier pour bien comprendre le fonctionnement de l'outil.

logiciels:

Adobe Flash Catalyst Cs5+
Adobe Illustrator CS5+ (optionnel)

fichiers sources:

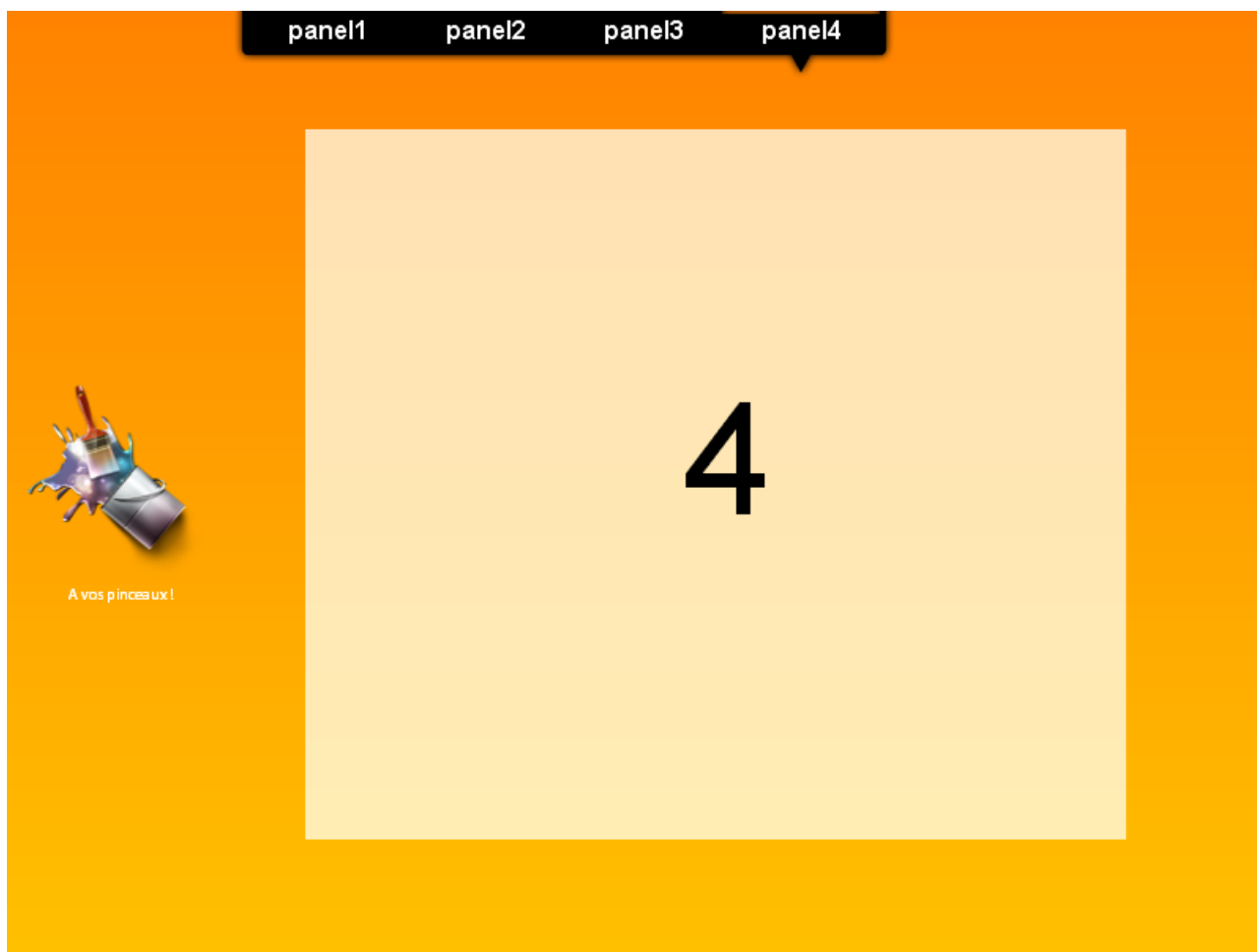
1. Présentation

En surfant sur le net, je suis tombé sur le site wokine.fr. Jetez-y un petit coup d'oeil. Le système de navigation implémenté est assez répandu mais efficace et agrémenté d'idées simples améliorant la perception des effets. Pour ma part, j'ai voulu voir si on pouvait le réaliser facilement avec Flash Catalyst sous-entendu sans la moindre ligne de code. On tâchera toute fois d'orienter nos choix d'implémentation pour ne pas se fâcher avec son collègue développeur qui sera chargé de reprendre le prototype pour réaliser la version finale.

Au cours de ce tutoriel, nous réaliserons les tâches suivantes:

- import d'un design réalisé sous Illustrator
- gestion des états d'un composant et de leur implémentation par Catalyst
- réalisation de composants génériques
- réalisations de transitions de type fondu, interpolation de mouvement ou de couleurs
- ajout d'interactions utilisateurs de type clic, rollover et rollout
- ajout d'effet de filtre

Pour se donner un peu de baume au cœur, jetons un petit coup d'œil sur l'application que nous allons réaliser.



2. Analyse de la structure du site

Cinq écrans sont facilement repérables:

- l'écran d'accueil: Il est composé d'un menu de type barre de boutons gérant la navigation à travers le site et d'un bouton, partagé par tous les écrans, qui permet un retour vers la page d'accueil

- 4 écrans similaires contenant un panneau semi-transparent affublé d'un numéro qualifiant sa position au sein du groupe et d'une barre de navigation reprenant la barre de bouton aperçu sur l'écran d'accueil.

Ces écrans ont été d'abord travaillés sur Illustrator CS5. Chaque écran est créé au sein d'un plan de travail (*artboard en anglais*) pour faciliter l'intégration au sein de Flash Catalyst. On y reviendra un peu plus tard.

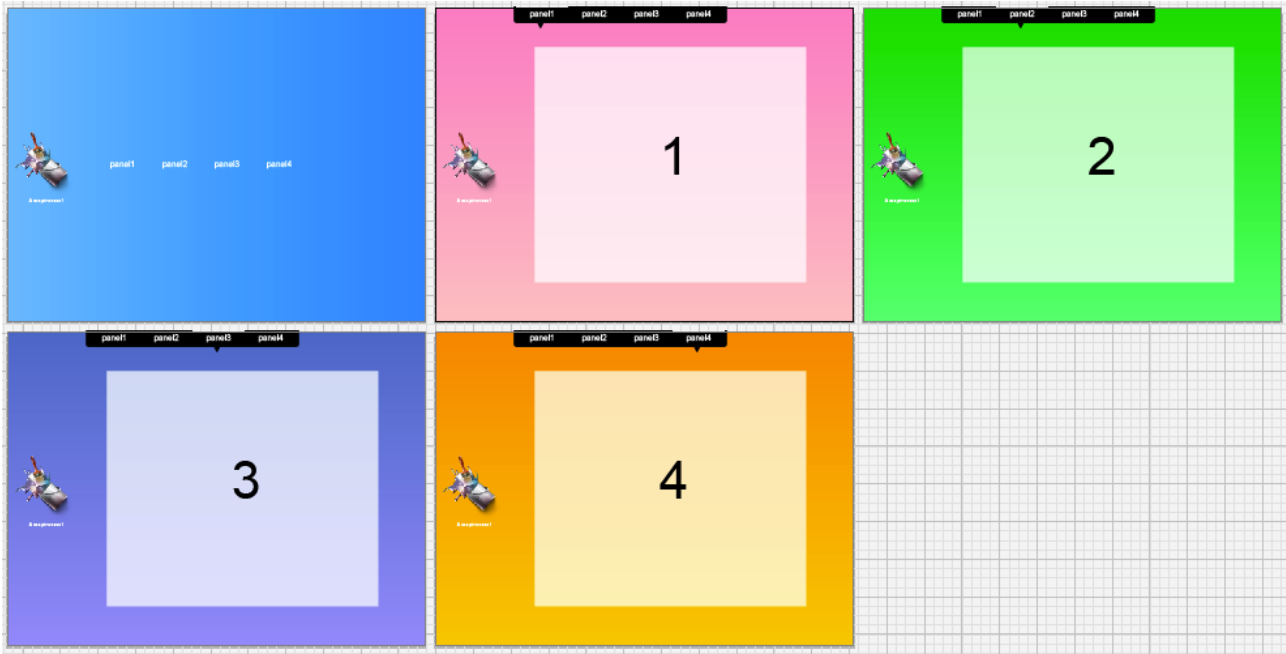


Illustration 1: les 5 écrans associés à 5 plans de travail différents au sein d'Illustrator

En terme de composants, on devra réaliser:

- un bouton permettant un retour vers la page d'accueil (état par défaut du site)
- une barre de boutons permettant la navigation entre les différents états créés
- une barre de navigation construite à partir de la barre de boutons précédente

3. A vos claviers

Nous allons créer un nouveau projet à partir du fichier Illustrator réalisé. Pour cela, à l'ouverture de Catalyst, sélectionnez l'option 'Depuis un fichier Adobe Illustrator AI...' et chargez le fichier nommé *screens.ai* que vous trouverez dans les sources jointes.

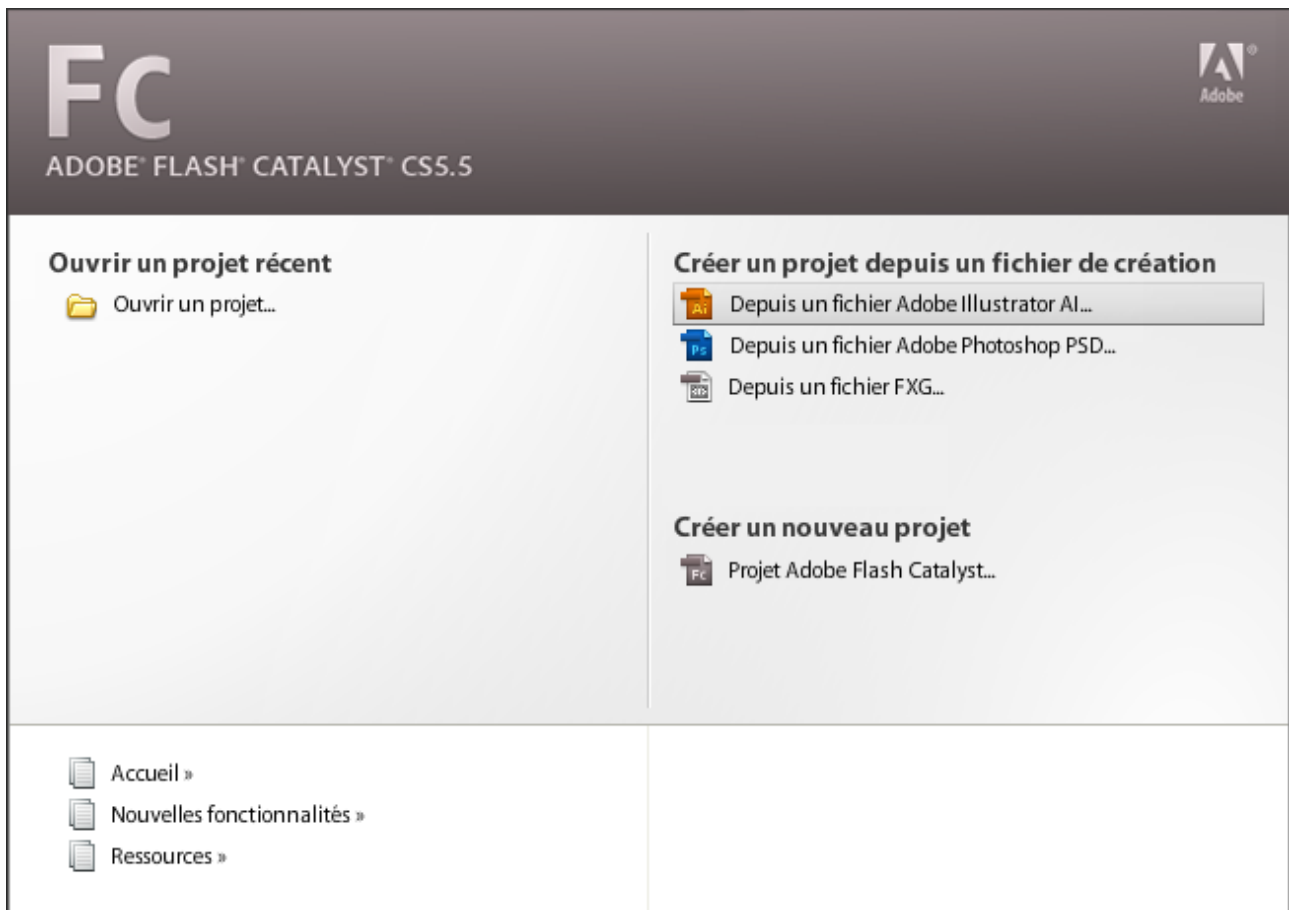


Illustration 2: Ecran d'accueil de Flash Catalyst

Une boîte de dialogue contenant les options d'importation apparaît à l'écran. Nous ne rentrerons pas dans les détails. Conservez juste les paramètres par défaut à l'exception de l'option « *redimensionnable* » que nous décocherons puis validez.

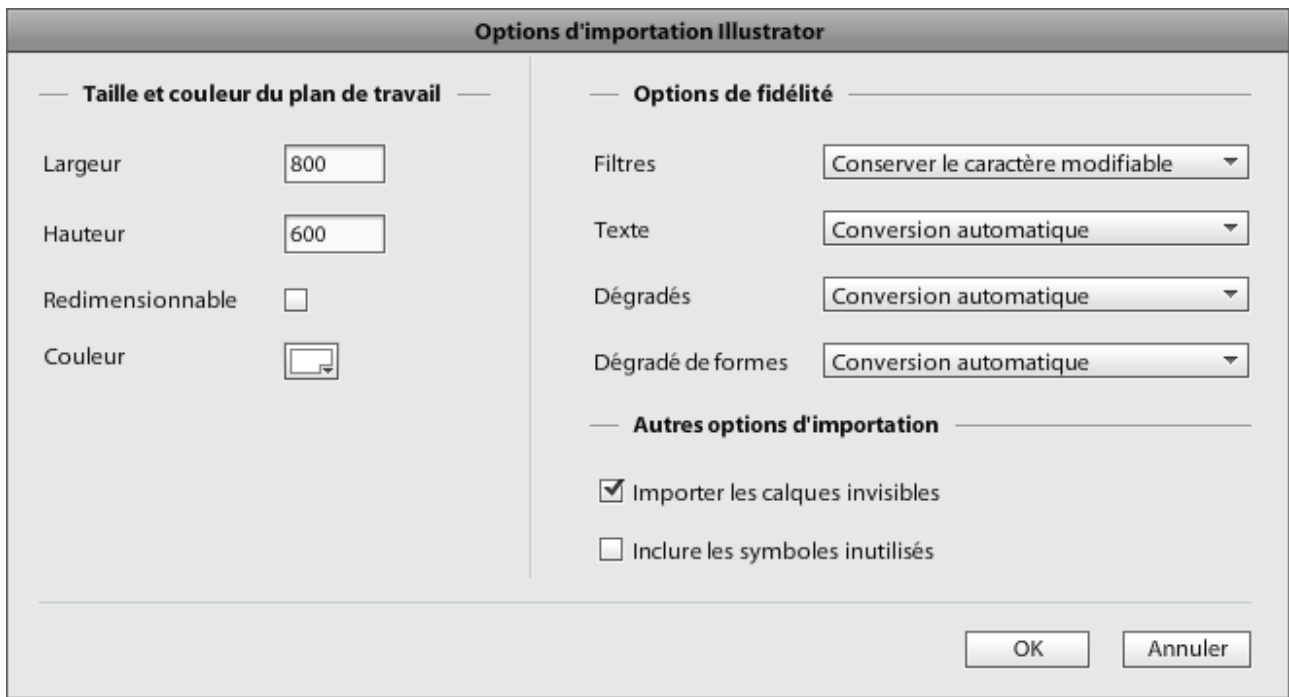


Illustration 3: Panneau d'options d'importation d'Illustrator

A partir de ces critères, le fichier *ai* est entièrement interprété. Chaque plan de travail créé sous Illustrator est reconnu par Flash Catalyst comme un état de l'application.

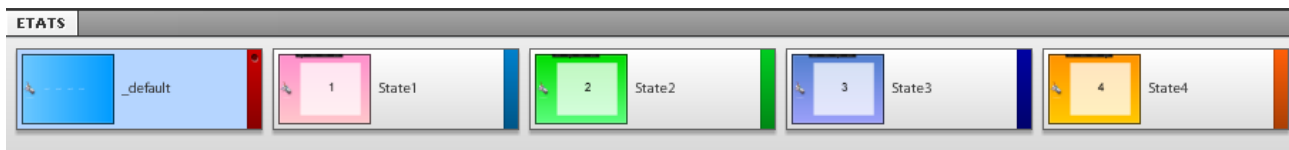


Illustration 4: Panneau Etats sous Flash Catalyst. La vignette marquée d'un point en haut de sa marge droite désigne l'état par défaut; ici l'état `_default`

On s'aperçoit également dans le panneau *Calques* que les éléments graphiques composant chaque plan de travail ont été regroupé à l'intérieur de calques différents. Le nom des calques est composé à partir du nom du plan de travail attaché. D'autre part, on constate que les groupes créés au sein d'Illustrator sont conservés.

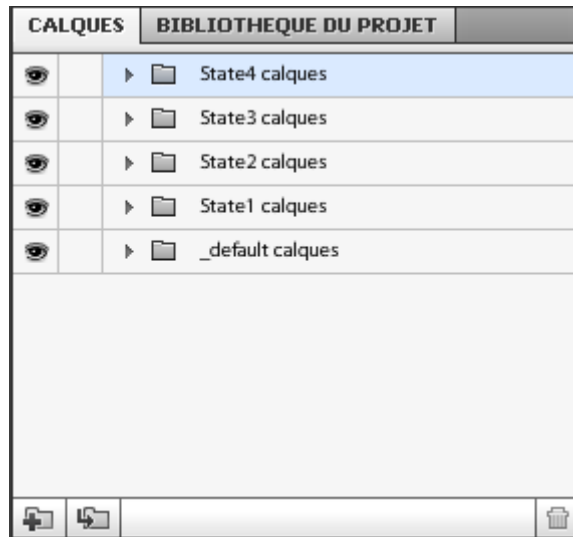


Illustration 5: Panneau Calques sous Flash Catalyst

La structure de base du site est en place. On peut maintenant s'attaquer à la réalisation des différents composants.

4. Le bouton de retour vers la page d'accueil

On se positionne sur le premier écran représentant l'état *_default*.

- (infos) -

Tout au long de ce tutoriel nous utiliserons le HUD diminutif de Heads-Up display. Le HUD est un outil d'aide listant les actions possibles sur la sélection courante.

1/ Sélectionnez le groupe nommé « HomeButton » contenant les éléments graphiques composant le bouton à savoir l'icône « pot de peinture » et le champ de texte « A vos pinceaux! ». Vous pouvez le sélectionner directement à l'écran ou depuis le panneau *Calques* sous les calques *_default calques*>layer 1.

A l'aide du HUD ou depuis le menu contextuel, convertissez le groupe en un composant de type bouton.

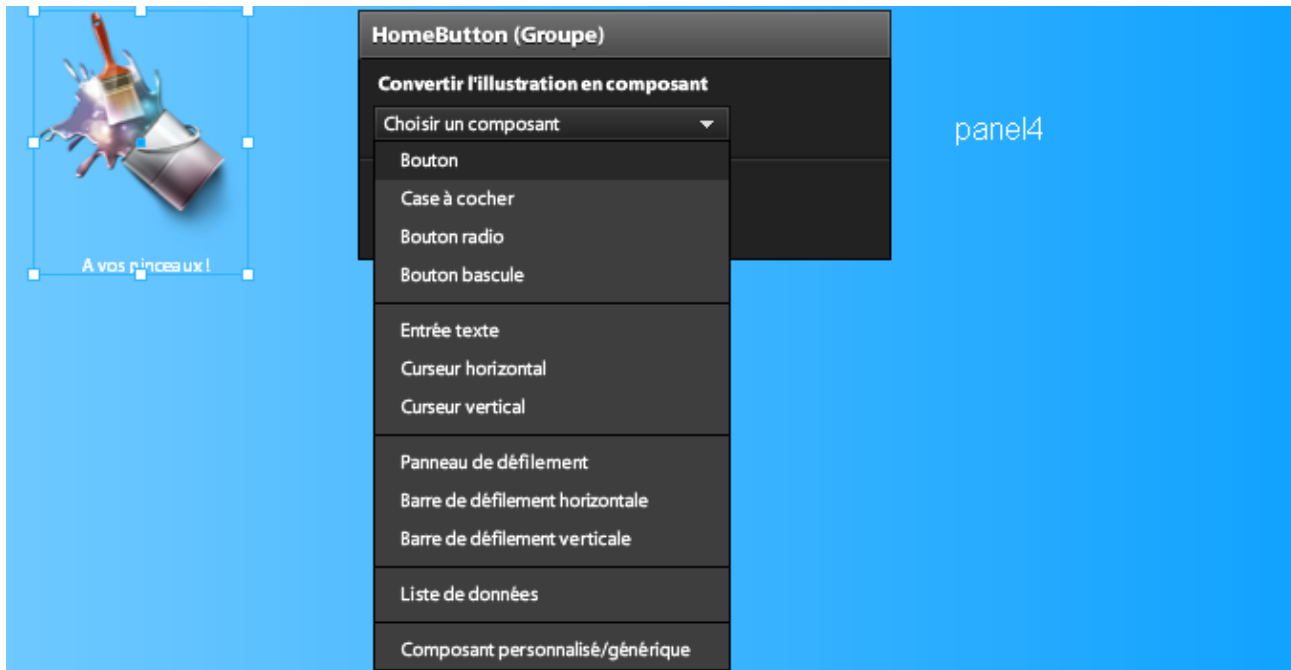


Illustration 6: Conversion des éléments graphiques sélectionnés en un composant de type bouton

2/ Une boîte de dialogue vous propose de choisir un nom pour la skin. Conservez le nom proposé.

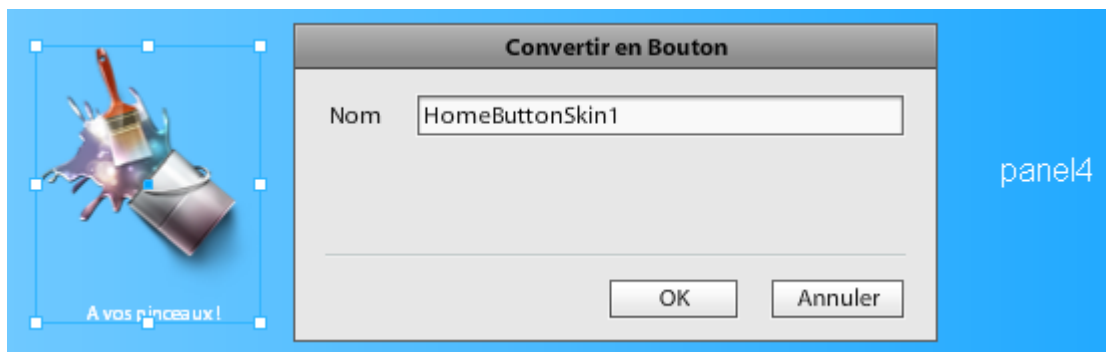


Illustration 7: Boîte de dialogue présentant le nom par défaut de la skin du bouton

Le bouton nouvellement créé apparaît sous le panneau «bibliothèque du projet» sous l'onglet «composants»

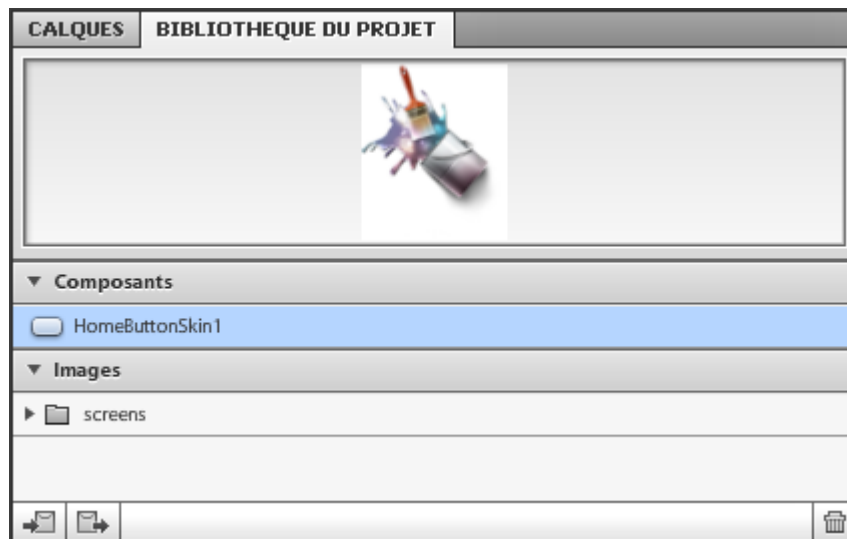


Illustration 8: Onglet Composants dans le panneau Bibliothèque du projet

Le HUD nous indique que le bouton possède les 4 états: Haut, Dessus, Bas et Désactivé.



Illustration 9: Liste des états existants du bouton présenté par le HUD

3/ Nous souhaitons rajouter un effet de *type* « lueur » lors du survol par la souris. A partir du HUD, cliquez sur le bouton « Dessus » pour accéder directement à l'état du même nom.

Une fois sur l'état « Dessus », dissociez les 2 éléments puis sélectionnez l'image. Dans le panneau *Propriétés*, sous l'onglet « Filtres », ajoutez le filtre de type « Lueur ».

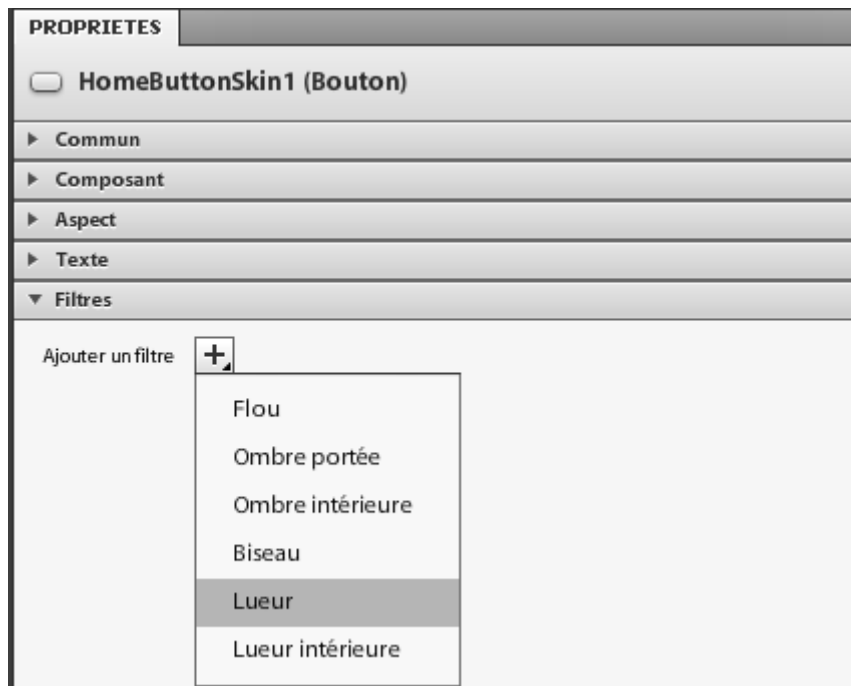


Illustration 10: liste des filtres disponibles au sein de Flash Catalyst

Conservez les paramètres par défaut à l'exception de la couleur que l'on choisira blanche #FFF.

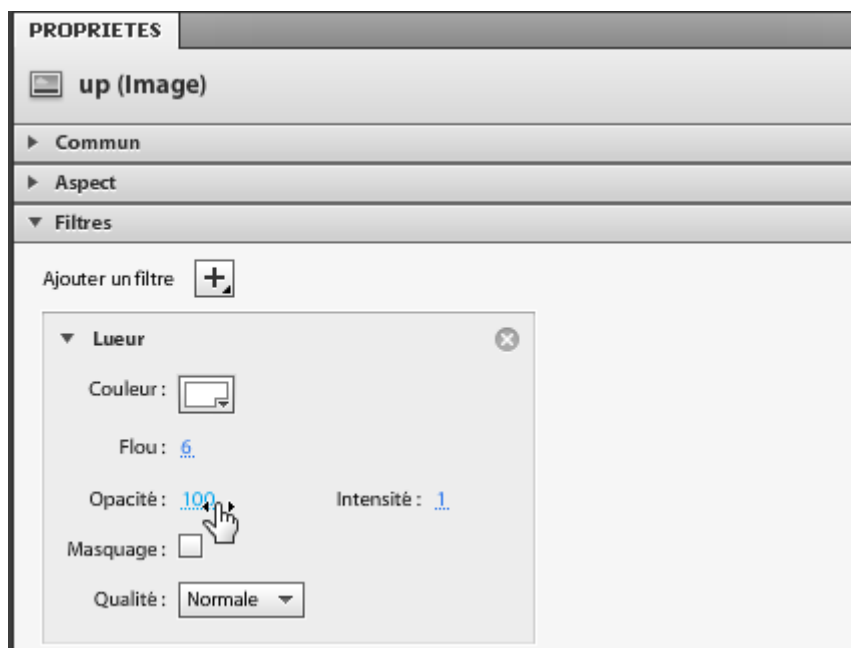


Illustration 11: Liste des paramètres pour le filtre de type lueur

4/ Retournez à la racine du document. Pour quitter le mode d'édition du bouton, cliquez dans le menu de navigation de l'espace de travail, sur le champ 'screens' qui est lien vers la racine du document.

- (infos) -
Catalyst reprend le nom du fichier pour nommer la racine du document. Dans notre cas, le projet est construit à partir du fichier screens.ai.

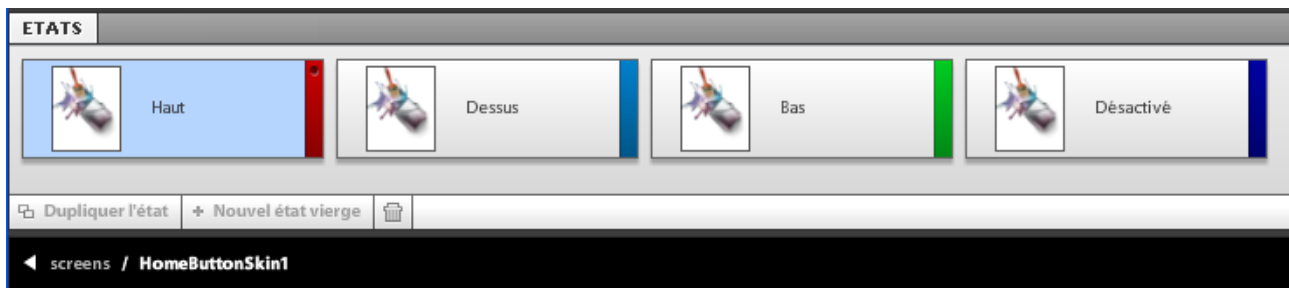


Illustration 12: Choisissez le bouton « screens » pour revenir à la racine de l'espace de travail

5/ Dans le panneau « Propriétés », sous l'onglet « Aspect » cochez la case « Curseur en forme de main » afin d'afficher l'icône main lors du survol du bouton par la souris.

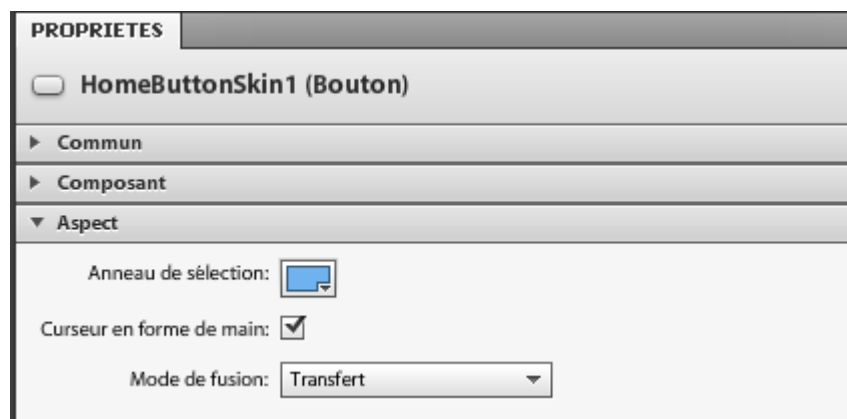


Illustration 13: Activez l'option « Curseur en forme de main » sous l'onglet « Aspect » du panneau « Propriétés »

6/ Nous souhaitons qu'à chaque clic sur ce bouton, l'application soit repositionnée sur l'état « *_default* ». Pour cela dans le panneau « Interactions », on ajoute un événement de type « lors d'un clic » ayant pour action de positionner l'application vers l'état « *_default* » quelque soit l'état dans lequel se trouve l'application comme présenté sur l'aperçu suivant.

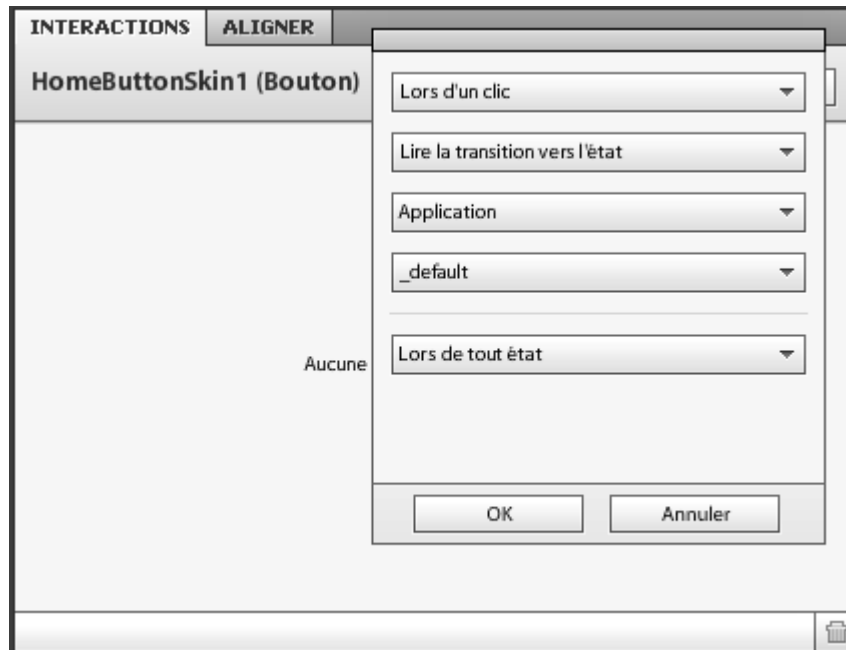


Illustration 14: Ajout d'un évènement "clic" sous le panneau "interactions"

Notre bouton est maintenant correctement configuré sur l'état « *_default* » de l'application. Il doit maintenant être partagé avec les autres états.

Pour cela, suivez les étapes suivantes:

1/ Un peu de ménage. Supprimez toutes les éléments graphiques représentant ce même bouton sur les 4 autres états. Vous pouvez vous aider pour cela du panneau « *Calques* » ou le réaliser directement à l'écran.

2/ Retournez ensuite sur l'écran « *_default* ». Sélectionnez le bouton et choisissez dans le menu contextuel l'option « *Partager avec l'état > Tous les états* ».

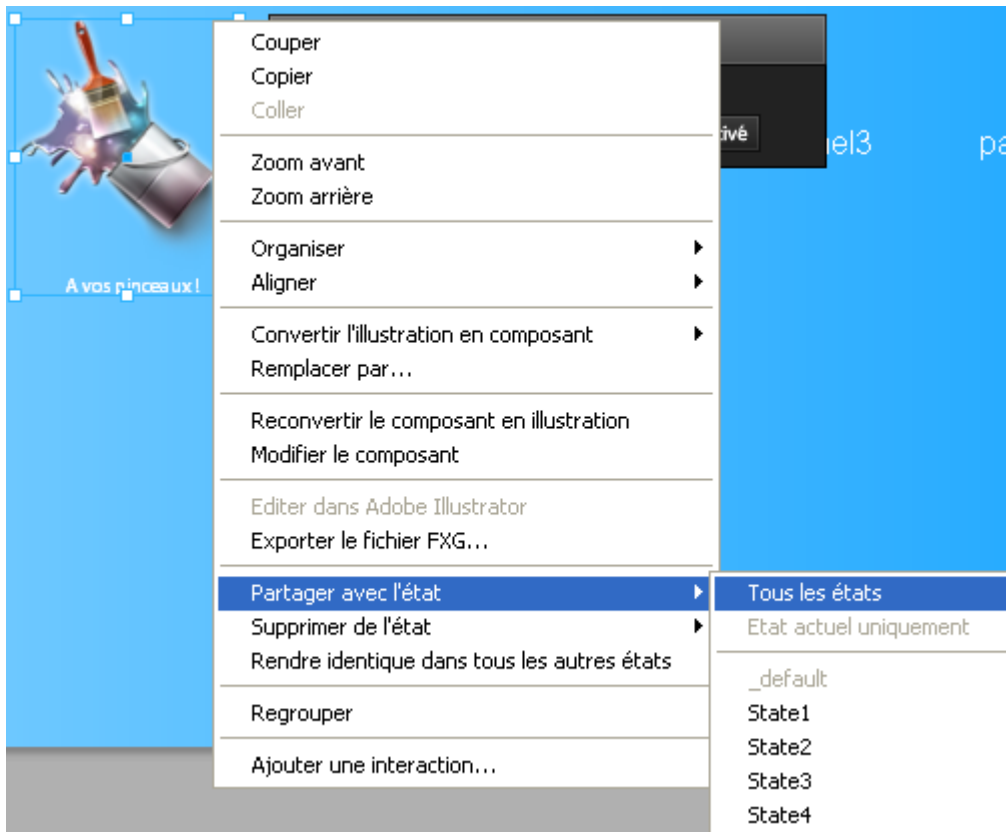


Illustration 15: Sélectionnez l'option « Partager avec l'état » puis « Tous les états »

Le bouton n'apparaît pas encore visuellement sur les autres états car le bouton est actuellement intégré au groupe lié au premier écran nommé « _default calques » qui est masqué sur les autres états.

3/ Sélectionnez le bouton dans le panneau Layer et glissez le à la racine de l'arborescence comme sur l'image suivante. Le bouton apparaît maintenant sur l'ensemble des écrans.

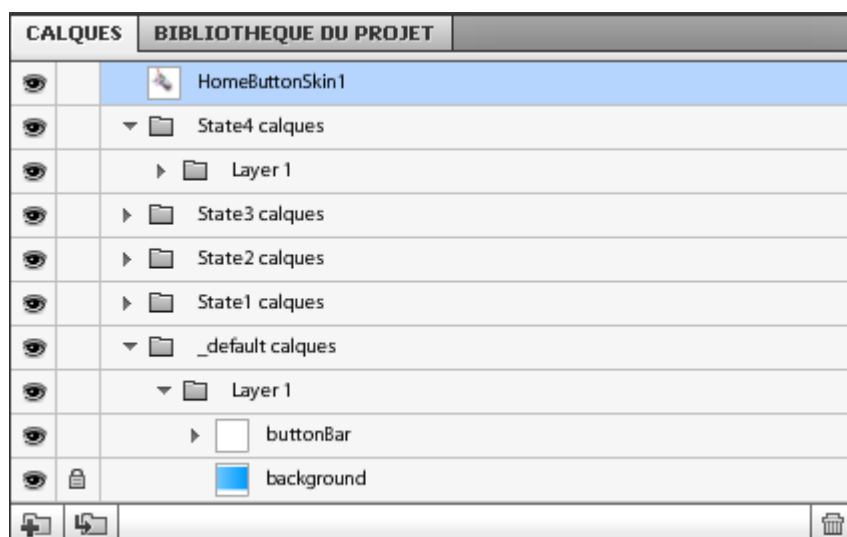


Illustration 16: Glissez la skin HomeBttonSkin1 à la racine de l'arborescence

- (infos) -

Il est important de suivre cette technique pour conserver une même instance au travers de plusieurs états et de pas utiliser la méthode de copier/coller qui multiplierait les instances à l'écran.

6/ Voilà c'est tout pour ce bouton. Testez.

- (infos) -

A tout moment vous pouvez regarder le résultat de vos changements en exportant une version dans votre navigateur à l'aide du raccourci CTRL+ENTER.

5. La barre de boutons

Avant de continuer, visualisez à nouveau le comportement implémenté dans l'aperçu final présenté en début de tutoriel. Certaines étapes déjà abordées ne seront pas aussi explicitées.

1/ Sélectionnez le groupe « buttonBar » présent sur le premier écran associé à l'état « _default » et convertissez le en un composant générique à l'aide du HUD.

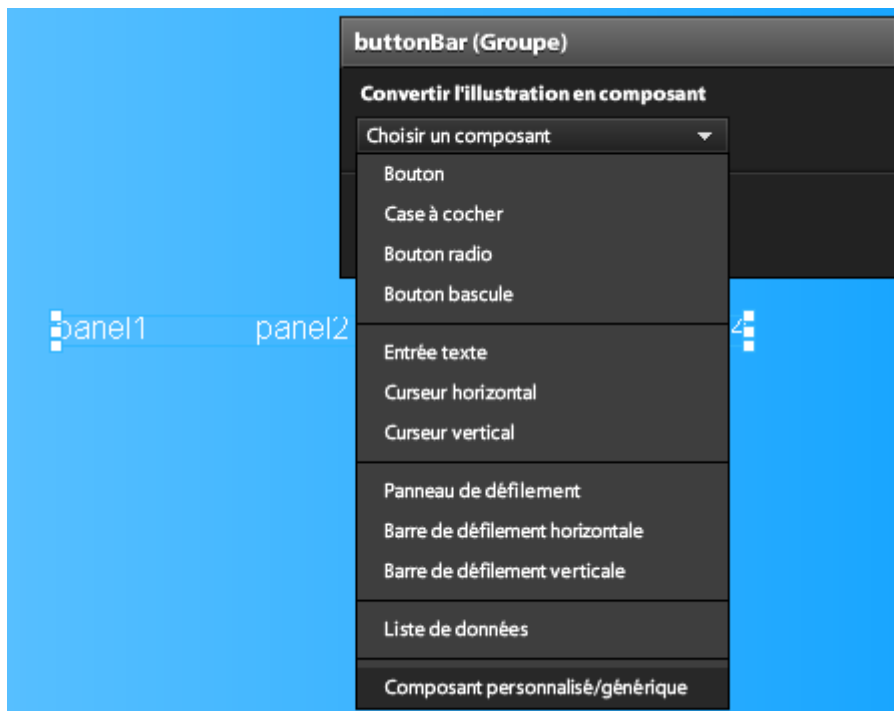


Illustration 17: conversion d'un élément graphique en un composant générique depuis le HUD

2/ Acceptez le nom par défaut proposé. Le composant est créé. Il apparaît dans la « bibliothèque du projet » sous l'onglet « composants ».

3/ Éditez le composant. Depuis le HUD, cliquez sur le bouton « State1 » qui permet d'éditer l'unique état du même nom.

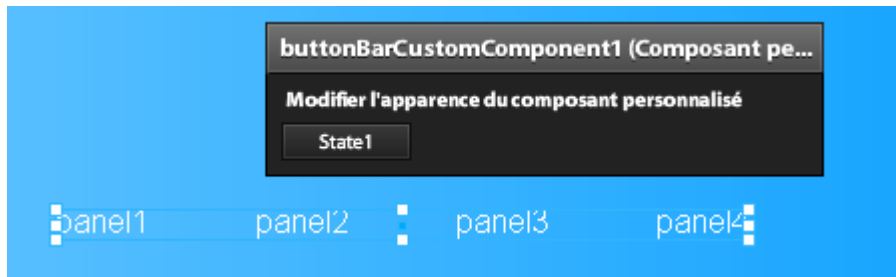


Illustration 18: Editez l'état "State1" en cliquant sur le bouton du même nom

La barre de bouton est composée de 4 éléments au comportement identique qui se différencient uniquement par leur label et les interactions qui sont attachées. le comportement de la barre de bouton est un peu particulier. Chaque élément actif du menu possède 2 états: un actif et non actif. Nous devons être capable de positionner lors du survol d'un élément les autres éléments sur l'état opposé. Ce n'est pas réalisable avec des composants de type bouton. Nous devons donc définir le comportement désiré dans un composant générique. Ce composant possède 2 états: un actif (texte de couleur blanche) et un inactif (texte grisé) .

- (infos) -

Catalyst ne permet pas d'exposer des propriétés au sein de composant générique. Par exemple, exposer la valeur d'un champ de texte à travers l'interface du composant. Nous devons donc créé 4 composants différents. Le développeur réunira ensuite ses 4 composants au comportement identique au sein d'un même et seul composant pour plus de concision.

Réalisons donc ce composant.

Les labels sont réunis au sein d'un groupe inutile. Choisissez l'option « dissocier » depuis le menu contextuel.

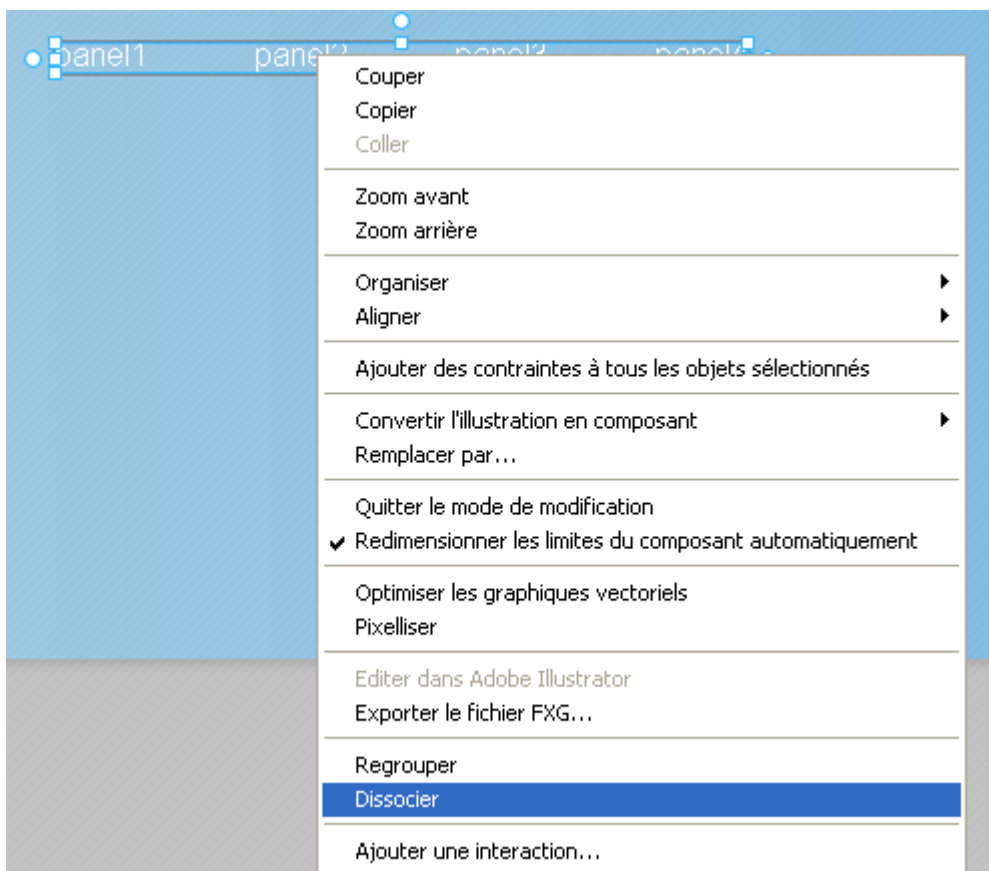


Illustration 19: Choisissez l'option "Dissocier" dans le menu contextuel

Nous détaillerons le procédé pour le champ de texte 'panel1' et dupliqueront le composant obtenu pour obtenir les autres composants jumeaux. En conséquence, supprimez dès maintenant les champs de texte labellisés panel2, panel3 et panel4.

Suivez les étapes suivantes:

1/ Sélectionnez le champ de texte restant, nommé panel1, et modifiez les propriétés comme sur le panneau ci-dessous.

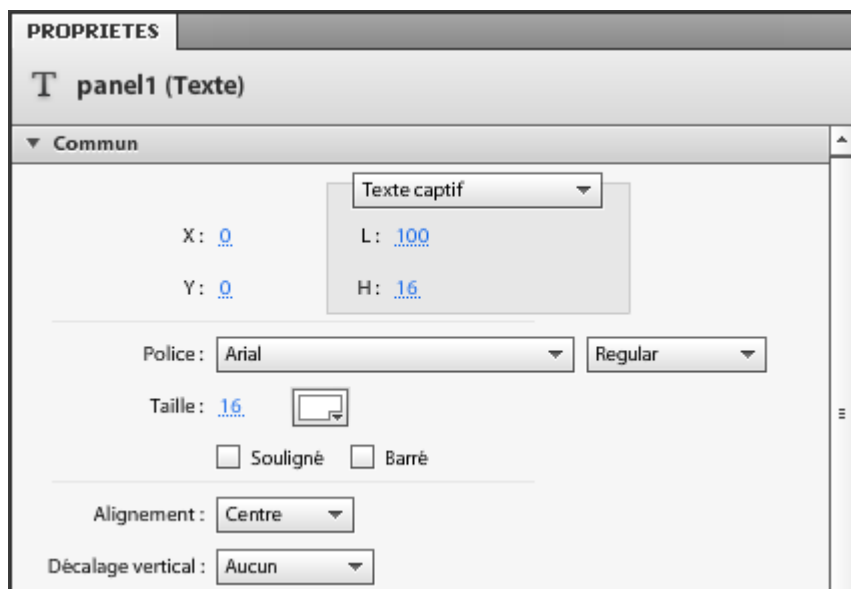


Illustration 20: onglet "commun" du panneau "Propriétés" listant les propriétés de l'élément texte

Concrètement le champ de texte est de type « texte captif » avec des dimensions fixes de 100x16 pixels et un alignement centré.

2/ Convertissez le champ de texte toujours sélectionné en composant générique en s'aidant du HUD,

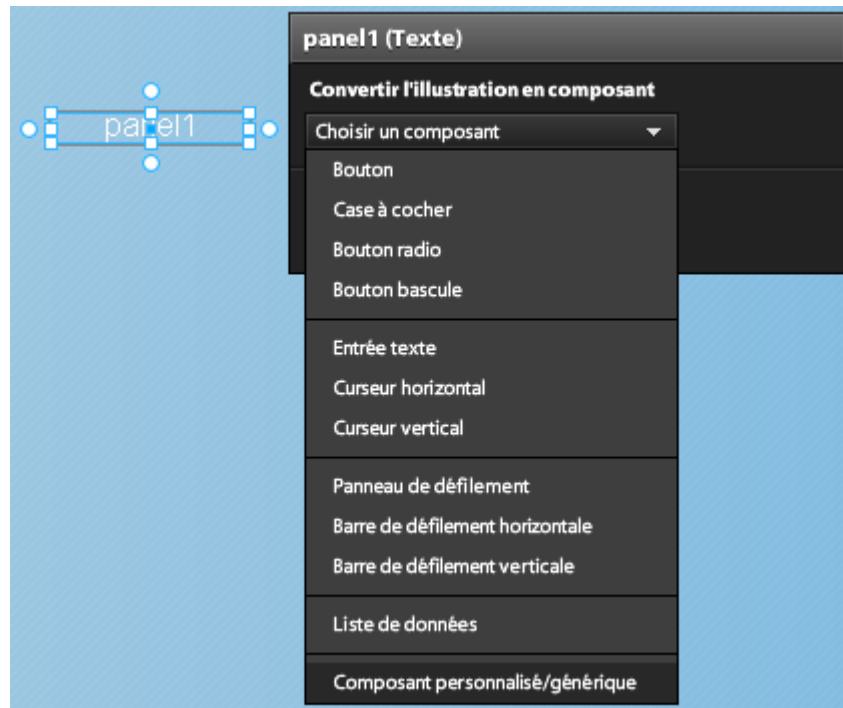


Illustration 21: Conversion du champ texte en composant générique

3/ Éditez le composant. Il possède 1 seul état nommé « State1 ». Cliquez sur le bouton « Dupliquer l'état » pour dupliquer l'état et renommez les états « State1 » et « State2 » respectivement en « normal » et « unactive ».

Dans l'état « unactive », changez la couleur du champ de texte en #CCC.

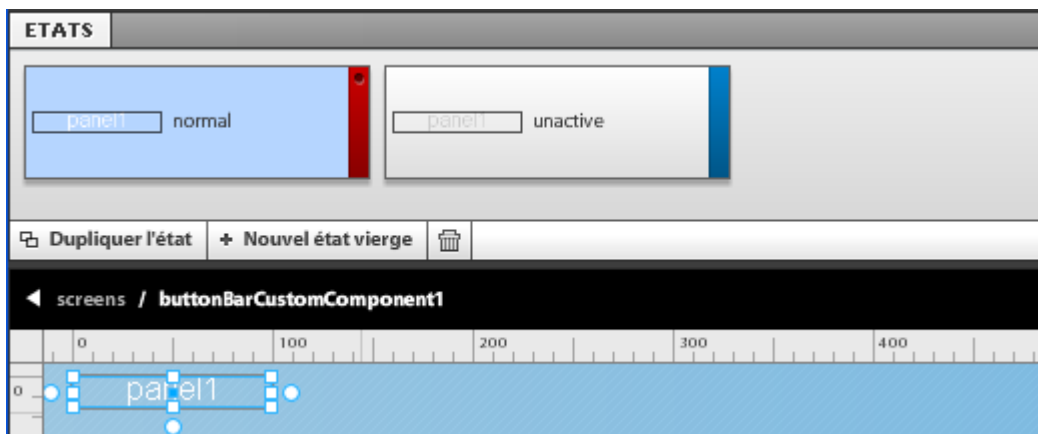


Illustration 22: Créez les états normal (#FFF) et unactive (#CCC)

Le composant est maintenant configuré.

4/ Retournez dans le composant parent représentant la barre de bouton nommé *buttonBarCustomComponent1*.

5/ Activez l'option « curseur en forme de main » dans le panneau « Aspect » pour le curseur de la souris.

5/ Répliquez 3 fois ce composant pour les « boutons » panel2, panel3 et panel4. Pour cela, sélectionnez le composant dans le panneau « bibliothèque du projet » sous l'onglet « composants » et sélectionnez l'option « dupliquer » dans le menu contextuel. Éditez les composants créés et ajuster les bons labels.

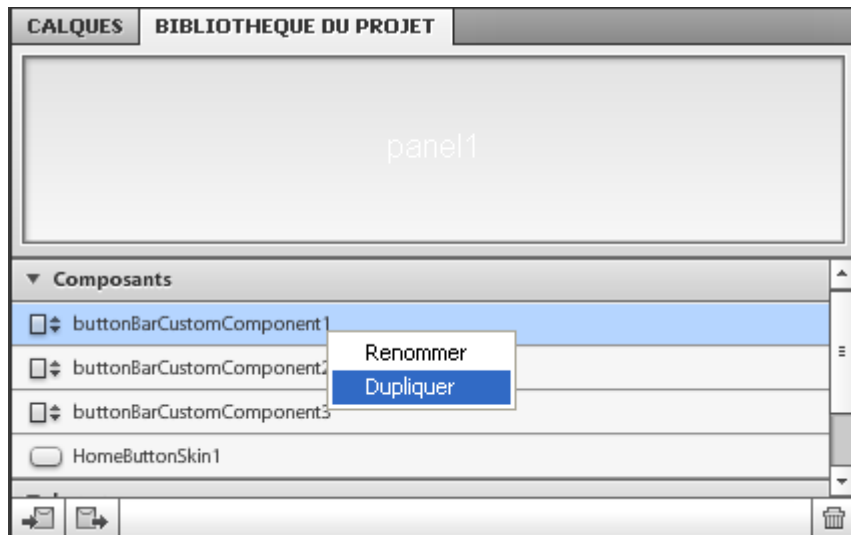


Illustration 23: Dupliquez le composant 3 fois dans le panneau "bibliothèque du projet" sous l'onglet « composants »

6/ Glissez les composants nouvellement créés dans l'espace de travail au côté du composant dupliqué sans la barre de bouton aux coordonnées voulues.

voilà ce que l'on doit obtenir.



Illustration 24: notre barre de bouton assemblée

7/ Définissons maintenant le comportement que l'on souhaite implémenté. Cette opération doit être opérée sur chaque composant soit 4 fois. C'est simple mais très répétitif.

Voici les 3 types d'événements que l'on souhaite implémenter:

- lors d'un clic --> positionne l'application sur l'état cible
- lorsque le curseur survole --> les 3 autres composants basculent vers l'état « unactive »
- lorsque le curseur s'éloigne --> les 3 autres composants basculent vers l'état « normal »

Ci-dessous, un aperçu du panneau « Interactions » pour le premier composant:

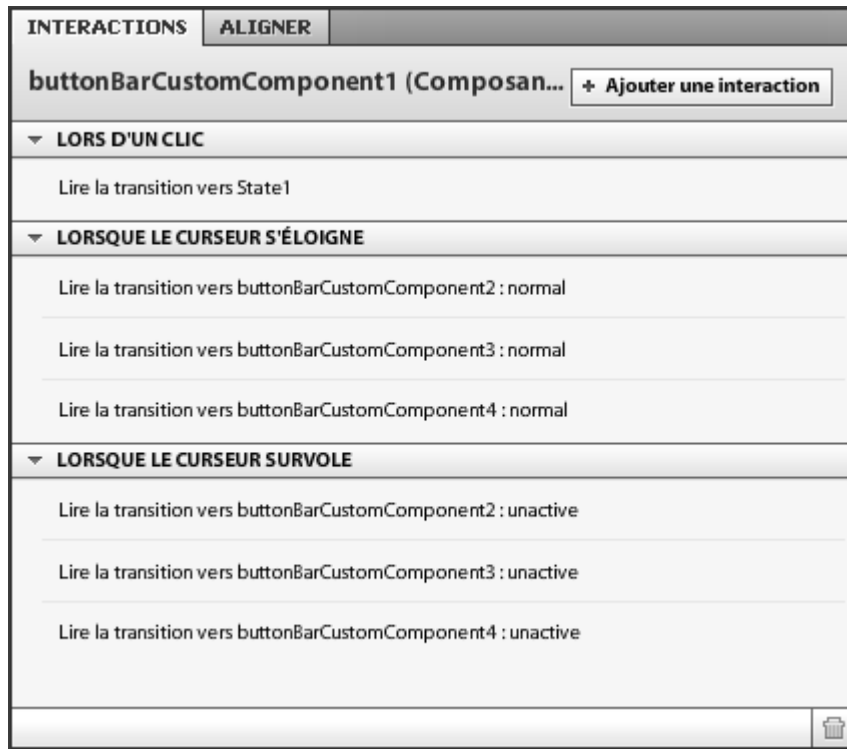


Illustration 25: Liste des évènements pour le composant "panell"

Faites de même avec les autres composants. Appuyez-vous sur le source si nécessaire.

9/ testez

6. La barre de navigation

Elle est visible sur tous les écrans à l'exception du premier. Positionnez-vous maintenant sur le 2eme écran représentant l'état State1.

L'item sélectionné est marqué par la flèche et un rectangle de hauteur moindre laissant apparaître la couleur du fond au dessus de lui.

1/ Sélectionnez l'élément graphique et convertissez le en un composant générique à l'aide du HUD. Acceptez le nom par défaut.

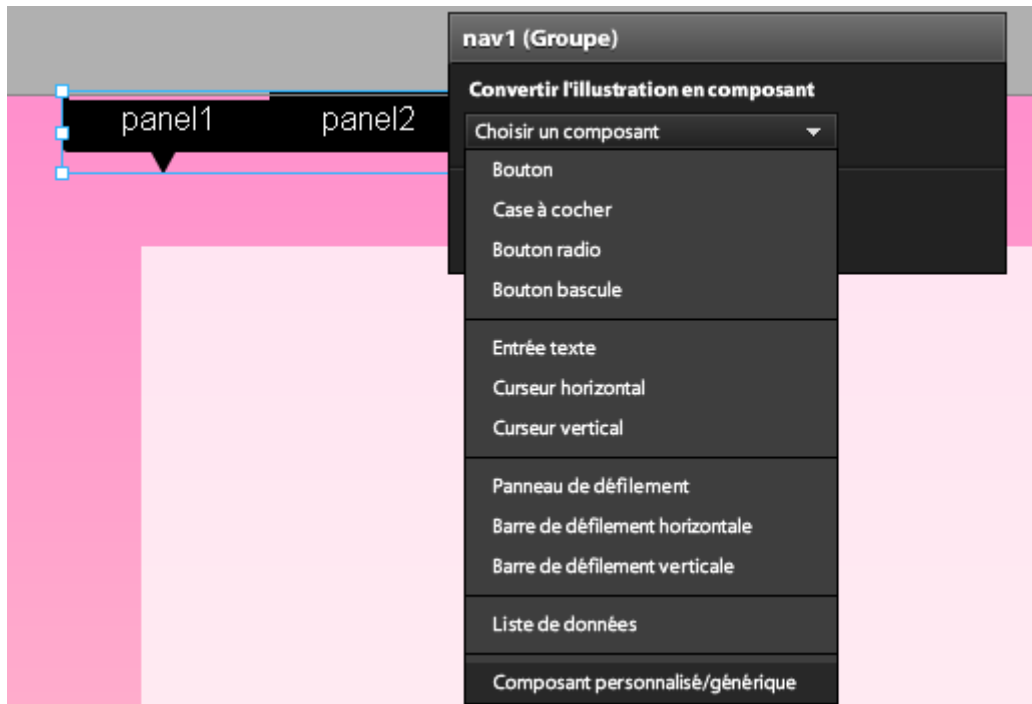


Illustration 26: Conversion de la barre de navigation en composant générique

2/ Éditez l'état « State1 » seul état existant en cliquant sur le bouton correspondant proposé par le HUD.

3/ L'ensemble des éléments est réuni à l'intérieur d'un groupe. Dissociez les éléments (même procédure qu'auparavant).

Ci-dessous un aperçu de la structure courante sous le panneau Calques:

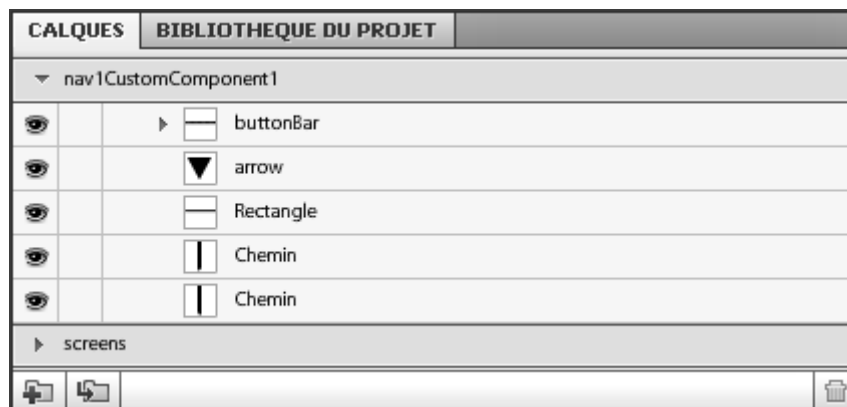


Illustration 27: Aperçu liste des calques

4/ Sélectionnez le groupe « *buttonBar* » et dégroupez jusqu'à obtenir une structure à plat.

5/ Supprimez ensuite tous les champs de texte qui seront apportés par la barre de boutons (composant) que nous rajouterons plus tard. Vous devez obtenir une structure semblable à celle ci.

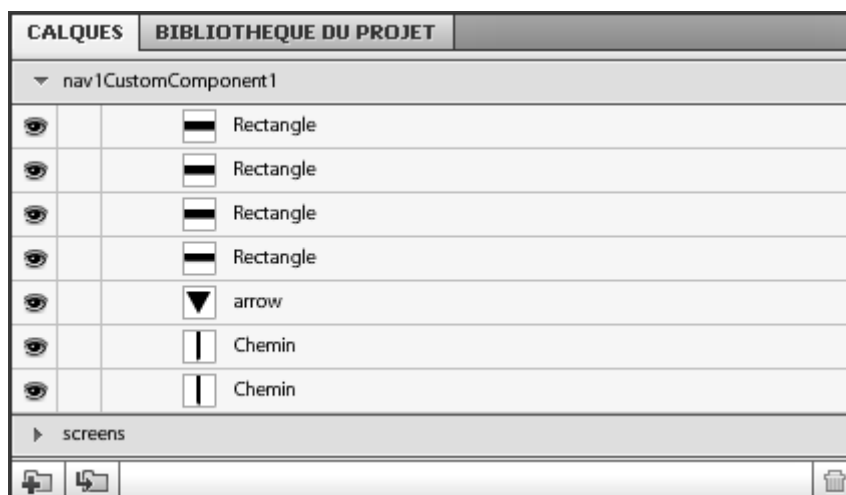


Illustration 28: Aperçu liste des calques après traitement

6/ Le composant possède actuellement un seul état nommé « State1 ». Nous allons créer 3 états supplémentaires nommés « State2 », « State3 » et « State4 ».

Reproduisez les règles suivantes en prenant soins à chaque fois d'aligner les rectangles par le bord inférieur.

State1:

1er rectangle en partant de la gauche: fixer hauteur 26px, centrez flèche horizontalement.
les 3 autres hauteur 31px.

State2:

2eme rectangle en partant de la gauche: fixer hauteur 26px, centrez flèche horizontalement.
les 3 autres: hauteur 31px.

Faire de même avec les états « State3 » et « State4 ».

Pour exemple ci-dessous, un aperçu de l'état « State3 »:

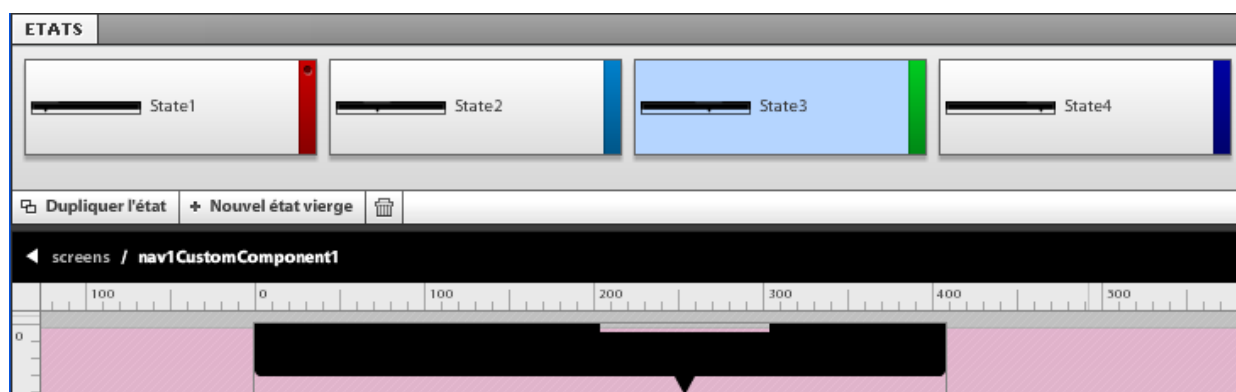


Illustration 29: La barre de navigation avec ses 4 états configurés

- (infos) -

Il est important pour ne pas rajouter de transitions supplémentaires de ne pas déplacer les rectangles d'un état à un autre mais de redimensionner uniquement le rectangle désiré.

En cas de soucis, appuyez-vous sur le source.

7/ Glissez maintenant depuis le panneau « *Bibliothèque du projet* », la barre de boutons (*composant*) créée précédemment aux coordonnées: x:0px , y:9px sur l'état « State1 » de la barre de navigation.

8/ Sélectionnez la barre de boutons et à l'aide du menu contextuel, choisissez l'option « *Partager avec l'état* » puis « *tous les états* » pour le partager avec tous les états.

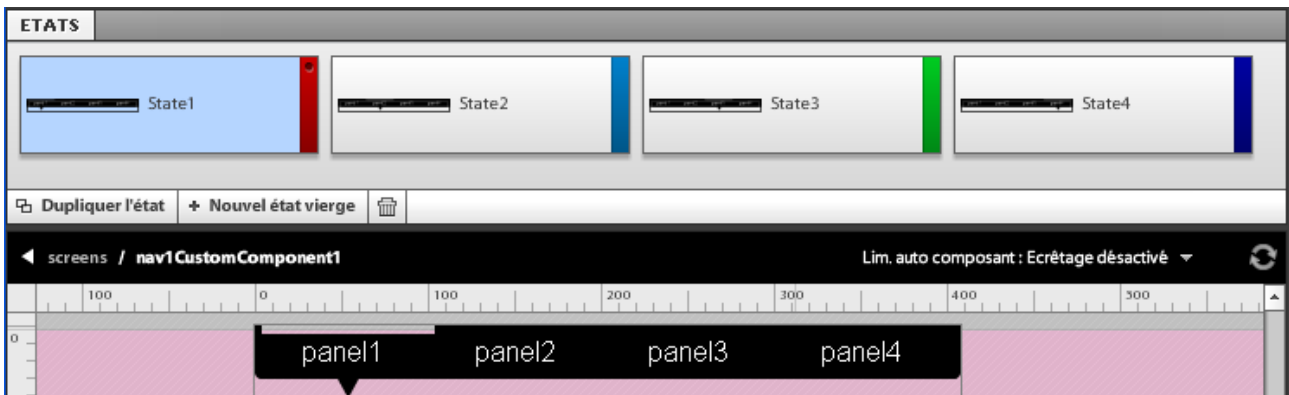


Illustration 30: Aperçu de la barre de navigation assemblée

7. Connexion de la barre de boutons avec la barre de navigation

1/ Éditez le composant représentant la barre de bouton.

2/ Ajoutez pour chaque bouton composant la barre de bouton un événement clic ayant pour action de positionner le composant barre de navigation sur l'état approprié (panel1->State1, panel2->State2...)

Pour exemple, suivez les étapes ci dessous pour configurer le bouton « panel1 ».

a/ Sélectionnez le bouton « Ajouter une interaction'» depuis le panneau « Interactions »

b/ attachez l'événement « *lors d'un clic* » à la cible correspondant à la barre de navigation. Suivez les étapes suivantes pour sélectionner la cible.

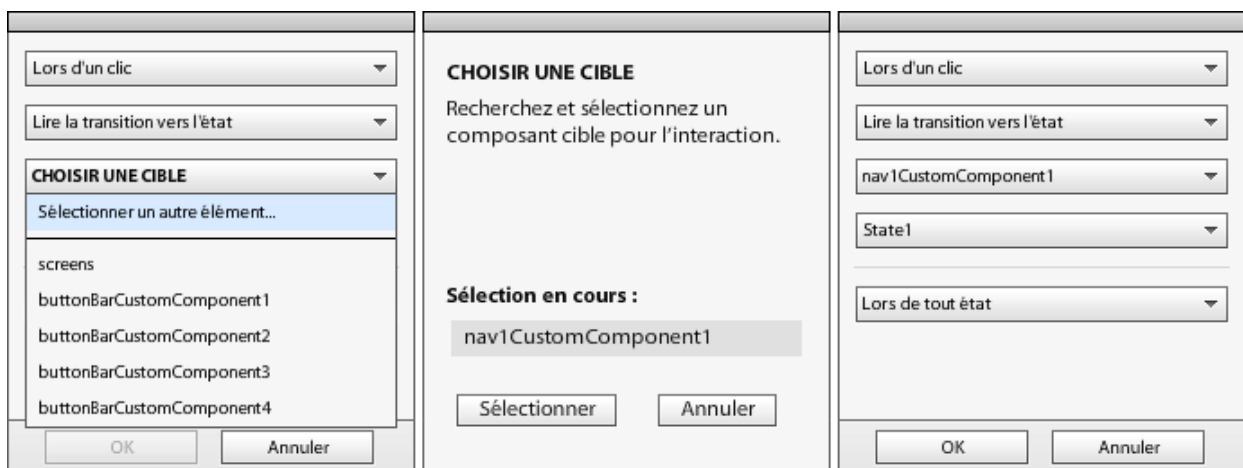


Illustration 31: Étapes successives entrant dans la création d'une nouvelle interaction

c/ validez et vérifiez que l'événement créé est bien ajouté à la liste des interactions comme ci-dessous souligné en bleu.

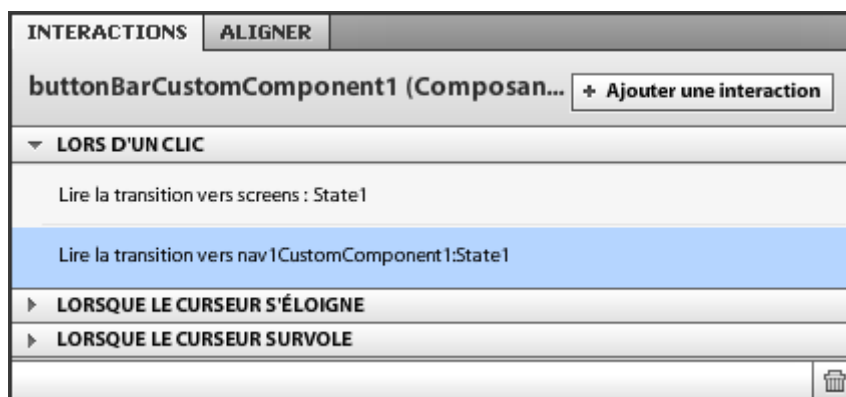


Illustration 32: Liste des interactions pour le bouton "panel1"

3/ Dans le panneau « Scénarios », sélectionnez l'ensemble des transitions listées et cliquez sur le bouton « Transition régulière ».



Illustration 33: Aperçu états des transitions sans transitions régulières

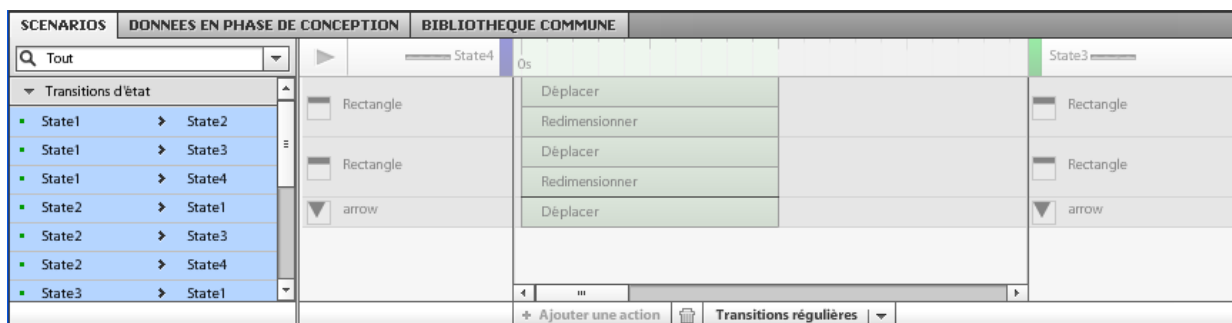


Illustration 34: Aperçu états des transitions avec transitions régulières

Par défaut, la durée des animations est fixée à 0.5s. Vous pouvez modifier cette valeur depuis le menu option accessible via la flèche à collée.

8/ Un peu de ménage. Revenez à la racine du document et supprimez sur les autres états toutes références aux éléments graphiques se rapportant à notre barre de navigation. Partagez le composant à travers tous les états.

9/ Sélectionnez le composant depuis le panneau « Calques » et glissez le à la racine de l'arborescence comme sur l'aperçu ci-dessous:



Illustration 35: Aperçu du panneau Calques

6/ Dans l'état1, déplacez le composant hors des limites de l'écran en jouant uniquement sur les coordonnées verticales.

7/ A la racine du document, sélectionnez l'ensemble des transitions listées et cliquez à nouveau sur le bouton « Transition régulière ».

8/ Testez l'application.

8. Transitions entre les états - glissement des panneaux

État « State*i* » vers états « State*j* » avec *i* et *j* de 1 à 4 et $i < j$

Dans cette partie, on va s'attacher à réaliser le défilement des panneaux du bas vers le haut. Le travail sera expliqué pour la transition de l'état « State1 » vers l'état « State2 » et devra être reproduite dans les autres cas de manière identique.

Si on s'attarde un peu sur le panneau « Scénarios » et plus particulièrement les éléments « panel1 » et « panel2 », on remarque que des transitions de type fondu sont déjà présentes. Cela s'explique tout simplement par le fait que « panel1 » est présent sur l'état initial « State1 » mais pas dans l'état final « State2 ». Une transition de type fondu est donc attaché pour faire disparaître progressivement l'élément de la scène. Le raisonnement inverse explique les transitions pour l'élément « panel2 ».

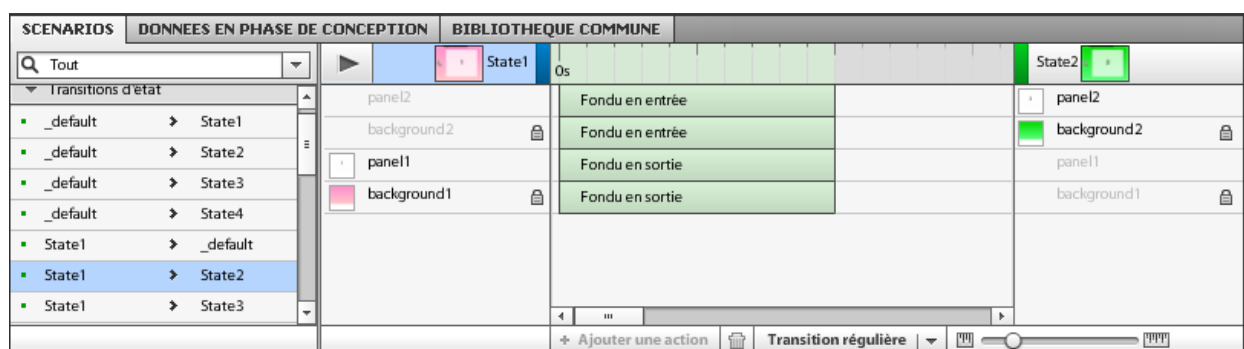


Illustration 36: Panneau "Scénarios" pour la transition "State1" vers "State2"

Pour faire apparaître l'effet de glissement, on va joindre à ces transitions des transitions de type « Déplacer » en respectant les critères suivant:

State1 --> State2: « panel1 » passe des coordonnées (x:190, y:75) à (x:190, y:600)

State1 --> State2: « panel2 » doit être repositionné hors champ aux coordonnées (190, 600) puis passer aux coordonnées (x:190, y:75) pour le replacer au centre de la scène.

Catalyst ne laisse pas la possibilité de spécifier dans une transition les critères initiaux. Cela nous contraint à doubler à chaque fois les transitions. une première brutale pour positionner le panneau lorsqu'il est encore hors champ suivi d'une deuxième pour le replacer au centre de la scène.

Sélectionnez la cible (« panel1 » ou « panel2 »), puis cliquez sur le bouton 'Ajouter une action' et choisissez dans le menu, le type de transition « déplacer ».

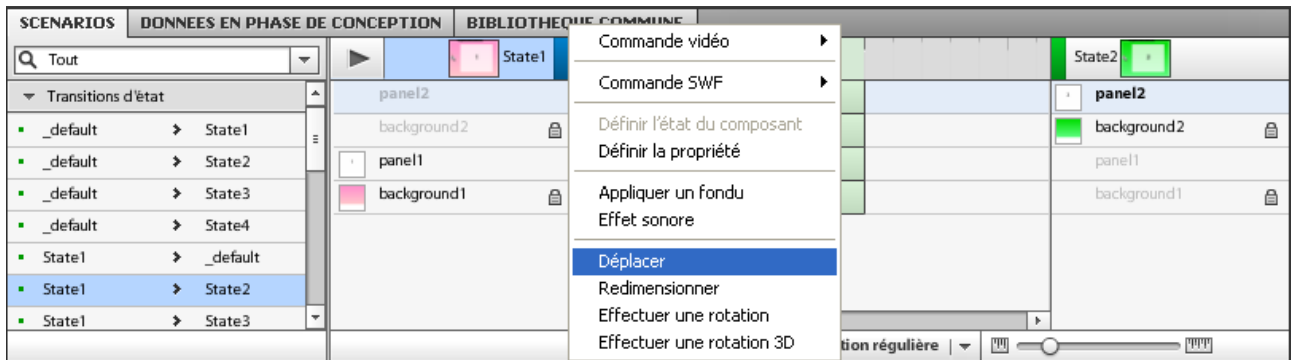


Illustration 37: Sélectionnez l'option « Déplacer » dans la liste des transitions

Configurez les paramètres de la transition dans le panneau « Propriétés »



Illustration 38: Paramètres de la transition sous le panneau Propriétés

Au final la transition de l'état « State1 » vers « State2 » doit ressembler à la photo ci-dessous.

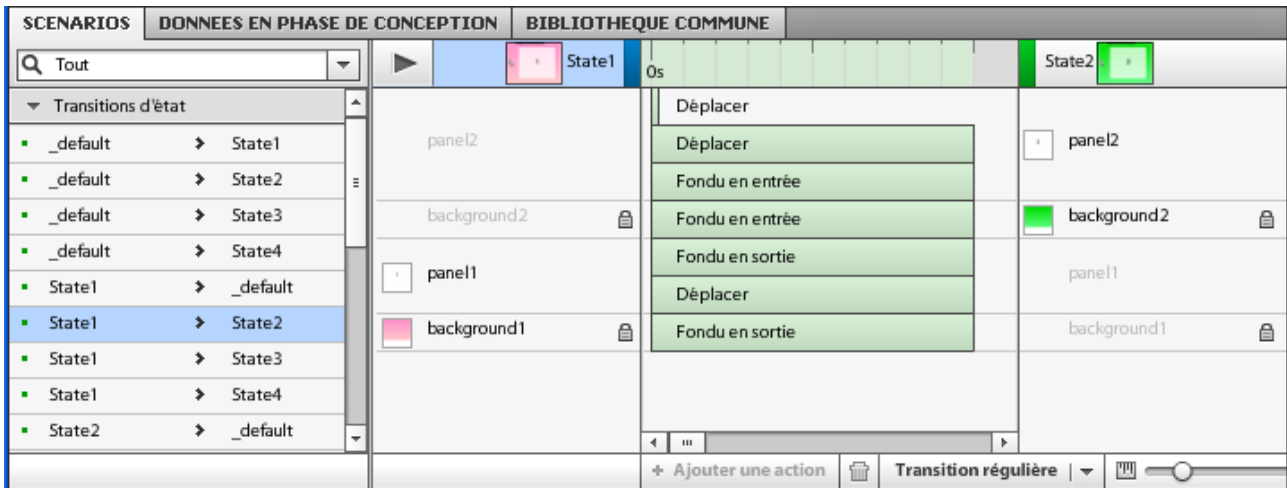


Illustration 39: Aperçu transitions pour le panneau 'panel1'

Les transitions étant verticales on ne s'intéressera ici qu'à la variable y.

- Le panneau 1 doit disparaître de l'écran en glissant du haut vers le bas. Le document ayant une hauteur de 600 pixels, la position finale sera fixée à -600.
- Le panneau 2 doit apparaître à l'écran en glissant du bas vers le haut. Le panneau doit être positionné hors champ donc sans animation en bas de l'écran avec une ordonnée de 600 pour permettre l'animation voulue. L'animation aura ensuite pour action de le centrer à l'écran avec une ordonnée de 75.

Reproduisez la même procédure pour chaque transition en prenant simplement soin à chaque fois de bien identifier les panneaux entrants et sortants. La procédure est simple mais répétitive et bien moins souple que si elle était implémentée via la programmation car on est obligé ici de passer en revue tous les états. Le programmeur gagnera à simplifier le code généré pour plus de clarté.

État « _default t » vers états « Statei » avec i de 1 à 4

On reprend la même procédure mais cette fois-ci il n'y a qu'un panneau cible à gérer.

9. Le fond d'écran

Cette partie est optionnelle mais conseillée en vue d'éclaircir le code, le panneau "Scénarios", la nature et la qualité de la transition.

Actuellement, le fond d'écran est composé d'un élément graphique différent pour chacun des 5 états. La transition entre chacun de ces fonds est donc actuellement un fondu qui permet de passer d'un élément à un autre. Ce serait plus naturel de posséder un seul élément graphique en guise de fond et ensuite ajuster son dégradé sur chaque état.

Mettons en place cette nouvelle structure.

1/ à l'intérieur du panneau « Calques », sélectionnez le fond sur l'état « _default » par exemple et glissez-le à la racine de l'arborescence tout en bas comme sur l'aperçu ci-dessous.

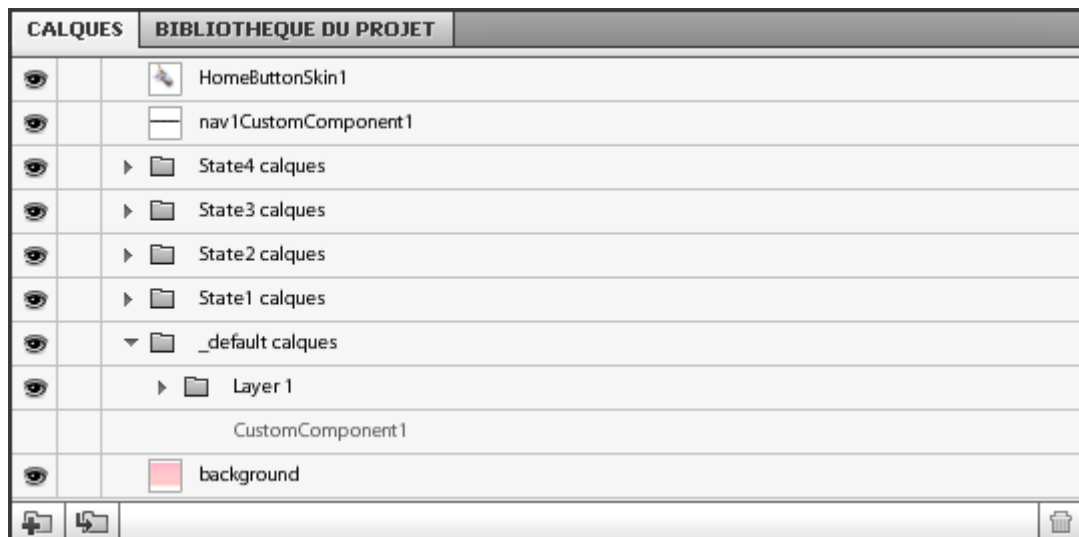


Illustration 40: Aperçu panneau "Calques"

2/ Sélectionnez le fond et partagez le avec l'ensemble des autres états.

3/ Supprimez les autres fonds après avoir pris soins de reproduire leur dégradé sur le fond maintenant commun

4/ En vous appuyant sur le panneau "Scénarios", réajustez manuellement la durée de la transition en glissant la poignée . Notez que la nature de la transition est passé d'un fondu à une interpolation de couleur entre 2 gradients.

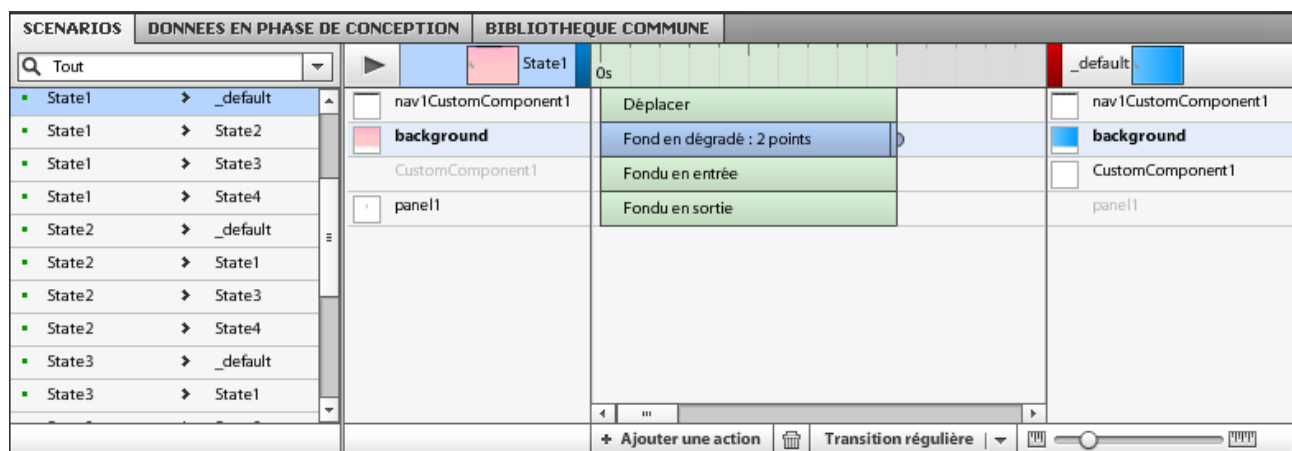


Illustration 41: Aperçu transition pour le passage de l'état "State1" à "_default"

5/ Testez l'application

- (infos) -

L'orientation des dégradés est important car elle accentue ainsi l'effet de glissement verticale. Ce sont les détails qui font la différence à la fin et c'est très bien rendu sur le site wokine.

10. Au final

Flash Catalyst met l'accent sur la gestion des états. Il serait donc primordiale de simplifier l'intégration des transitions entre ses états pour limiter les manipulations et implicitement le code généré. L'idée derrière Catalyst reste de simplifier et d'accélérer le flux de travail entre designer et développeur. Il y a toujours plusieurs façons d'obtenir un même résultat. Il ne faudrait pas que Flash Catalyst fournisse la

mauvaise. C'est un outil récent et j'attends avec impatience les évolutions.

11. Ou aller maintenant

En guise d'exercice, je vous laisse finaliser le travail pour notamment ajouter les transitions attachées à la barre de boutons et coller un peu plus à l'implémentation faites sur le site wokine par exemple. Maintenant que vous avez compris le principe rien ne vous empêche de vous éloigner un peu du modèle et implémenter vos propres effets. Une rotation 3D ?

Ce tutoriel étendue des transitions de la barre de boutons sera bientôt disponible dans un tutoriel vidéo présenté sur le site tuto.com.