

View Point



UI Prototyping with SketchFlow

Sakshi Sharma, Atul Gupta

Introduction

When developing applications, be it business critical, external facing or departmental applications, we usually have robust processes in place to manage the functional and non-functional requirements. One aspect that however gets neglected often is the user interface (UI) design, which gets handled as an afterthought.

Ensuring good UI ensures that users of the application are happy and comfortable using it. A typical process for UI design is to start with user interviews, create a UI prototype, get it reviewed by users, and finally pass to development team for implementation. The prototyping and user review may need to have multiple iterations to narrow down to something that works for all.

For more information, contact askus@infosys.com

There are multiple tools available for developing UI prototype. Some of these are listed below:

1. Images (using any standard imaging software)
2. Microsoft PowerPoint
3. Adobe FireWorks and Flash Catalyst
4. iRise
5. Axure
6. FluidIA
7. Visio stencils

For a detailed listing, refer to “GUI Prototyping Tools” (Reference 1).

The common tools that we have seen, however, are PowerPoint presentations, Visio or just plain simple images. These have their benefits and hence their prevalent usage. But we find there are some glaring gaps when it comes to effective prototyping. In our view, the key requirements that a UI prototype should fulfill are:

1. Quick and easy to build – should be able to develop the prototypes quickly without significant effort
2. Communicate ideas effectively – designing the UI, isn’t just about static screens, but also includes interactions, navigations, state changes, etc. and these should be easily demonstrable
3. Throwaway investment – a big need for UI prototyping is the ability to quickly work on a concept, validate it, and then discard it whether or not it works, as it is only a prototype and not production ready code
4. Reusable artifacts – while prototyping is mainly quick and non-standards based development; styles, resources, etc. would get built when trying out different look and feel. There should be some way to leverage these in the final implementation

For a more detailed discussion on this topic, refer to “The Art of UI Prototyping” (Reference 2).

With technologies like Windows Presentation Foundation (WPF) and Silverlight (SL), Microsoft has brought designers and developers closer. Expression Blend from Microsoft is a great product that allows designers and developers to work together as a team. Refer to “The New Iteration” white paper for more details on this (Reference 3). The work done by designers can be more easily integrated into final implementation by developers, thus, unveiling a new age of user experience (UX) design. UX is about how to make the UI of an application interesting to look at and work with so that the overall experience of the users is lot more satisfying.

Microsoft added a new product in their Expression Suite namely SketchFlow (Reference 4). SketchFlow is a UI prototyping tool and addresses some of the points we have listed above. In this document, we will talk about what SketchFlow is, how it is better than the earlier, more prevalent design prototyping tools, and also provide our viewpoints on its various features.

Target Audience

In this viewpoint paper, we will be focusing on SketchFlow 3 (as part of Expression Suite 3). This document is primarily intended for designers and developers involved in building design prototypes for applications. While SketchFlow is as such more suitable for WPF or SL application prototypes, but it could be equally well utilized to build prototypes for applications built on any other platform.

Overview of SketchFlow

Few of Microsoft’s goals for creating SketchFlow:

1. Be able to create fast, inexpensive, and dynamic UX prototypes
2. Disposable
3. Communicate the design intent
4. Collect and evaluate feedback
5. Deliver compelling UX proposals in cost effective manner

SketchFlow combines the power of Expression Blend and all aspects of WPF and Silverlight that designers have already learnt to create quick, interactive, and functional prototypes without having to code much. Prototypes built-in SketchFlow are not just static screens with some content layout, but are rich and interactive screens that allow user action on them, and thus give a better feel of how the particular screen or controls will behave in runtime. Using concepts like states, animations and cross screen navigations, you can build prototypes that are *almost* production ready. The sketch/wiggly styled controls make sure that the reviewers don't confuse the prototype with production ready application. Business users can not only independently review and navigate across the prototype via the SketchFlow player, but can use the same to provide in-place comments by inking at appropriate places. This feedback can be replayed by developers, in-line within Blend, to clearly understand what the comments mean and fix them quickly.

Prototyping cannot be completed without having some sort of data to display because prototypes are not just about screens but about the information to be displayed. SketchFlow leverages another important feature of Expression Blend namely, "dummy data". Dummy data of a variety of formats (integers, strings, web URLs, addresses, currency etc.) can be easily added and then displayed on the screens to show how users will interact with the application data.

In the next section we will deliberate over various features of SketchFlow in more detail and also provide our viewpoints on the same.

SketchFlow Features

This section highlights the main features of SketchFlow and our views on their suitability. Listed herein are also features that we deem can be added to SketchFlow in its upcoming versions to make the product more useful.

Note: The features below aren't in any particular order.

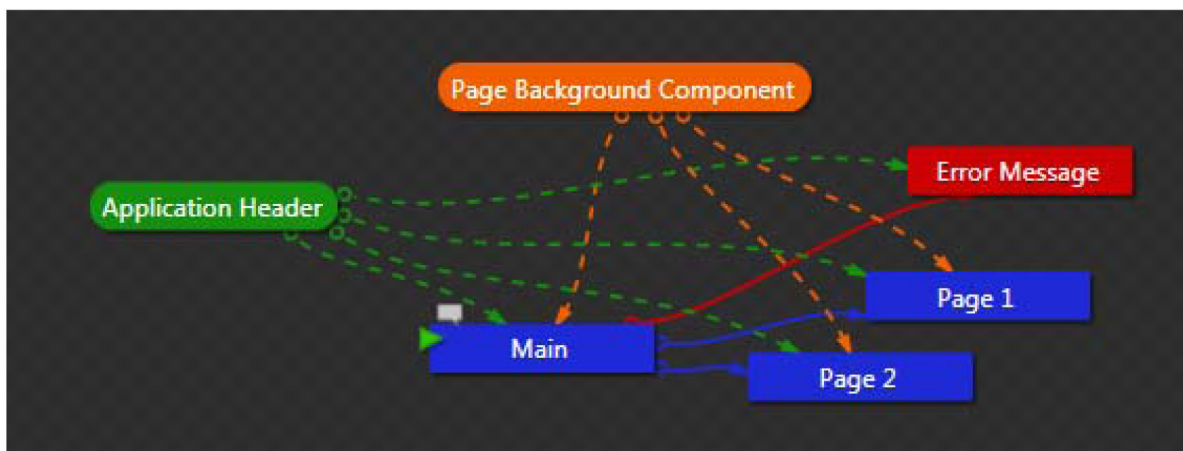
SketchFlow Map

Key to any design process is the flow maps, which can be treated similar to site maps that help show the navigation structure of a web site. Similar to this, SketchFlow provides a panel called SketchFlow Map where the flow map can be built very easily and quickly. The map gives a view on the screens in the prototype and also how they are connected to each other.

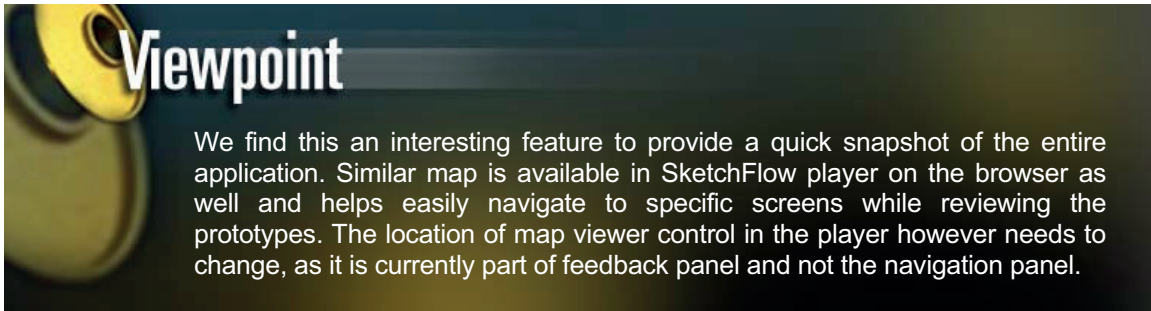
The flow map mainly has two items to display, Screens and Components. A Screen represents a unique page in the prototype and a component is a reusable element (say, a menu control), which can be used by many screens throughout the prototype. Visually, these are distinguished in the map view through a regular rectangle for screens and a rounded rectangle for components.

Using the SketchFlow Map designer can make connections between various screens and components to show the navigation flow and relationship involved. By default, screens are blue in color and components are green in color. However, one can color code these screens and components to help identify any grouping between various screens and components or any design convention. In the example flow shown below, blue nodes are the navigation pages or screens, red node is the error page, green node is the UI component, and orange node is the form component.

Figure 1 SketchFlow Map Panel



SketchFlow map also allows easy access to any screen/component. One can double-click on a node in the map to open that screen/component. With zoom in-out capability, one can easily view large and complex maps, though readability does become difficult in such case.

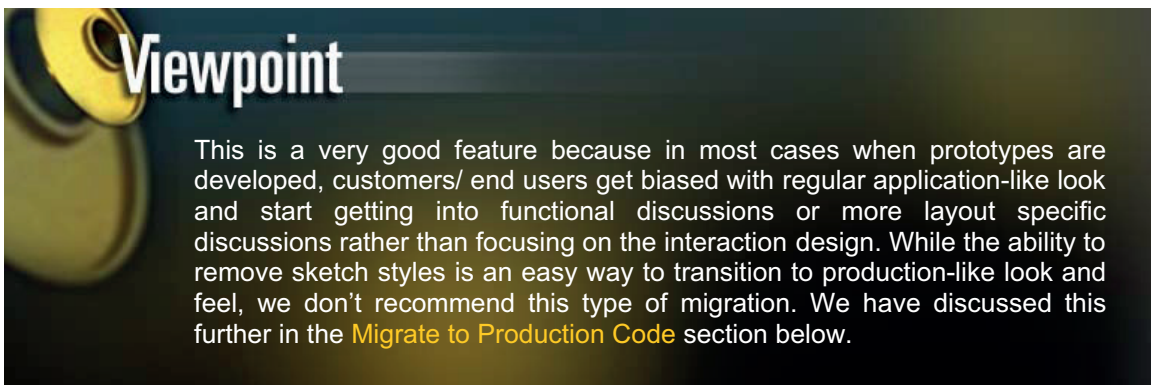
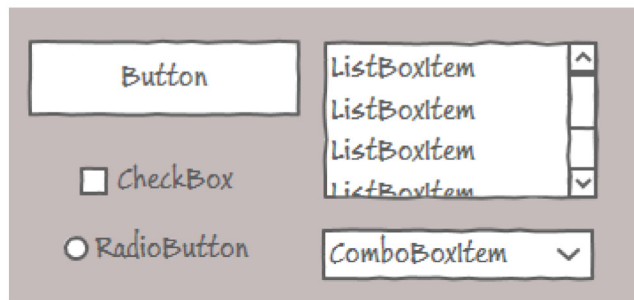


SketchFlow Styles

Blend 3 includes a set of skins for controls with sketch styles (wiggly styles), which gives them a hand-sketched look to be used with SketchFlow applications. Sketch styles, because of their unfinished look, can help reviewers to concentrate on the interactive design of the prototype instead of the visual design. These styles are available both under the Assets panel and on the Control tab. The following figure shows some of these styles.

When prototyping is complete and approved by the reviewer, a developer can easily remove these Sketch styling and all controls revert to standard Windows styling.

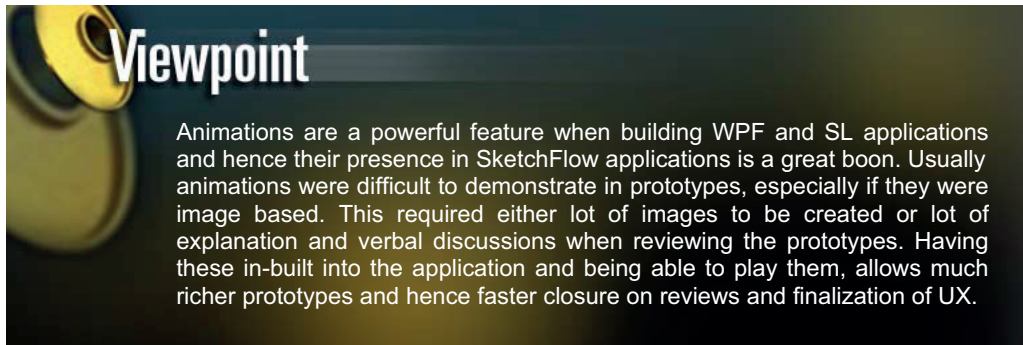
Figure 2 SketchFlow Styles



Animations

SketchFlow animation is used to add animated sequences to demonstrate the different states of a particular UI. They can also be used to hold different states for any specific screen. You can use keyframe animation tools of Expression Blend, States panel, or the SketchFlow animation panel to create a simple animated sequence.

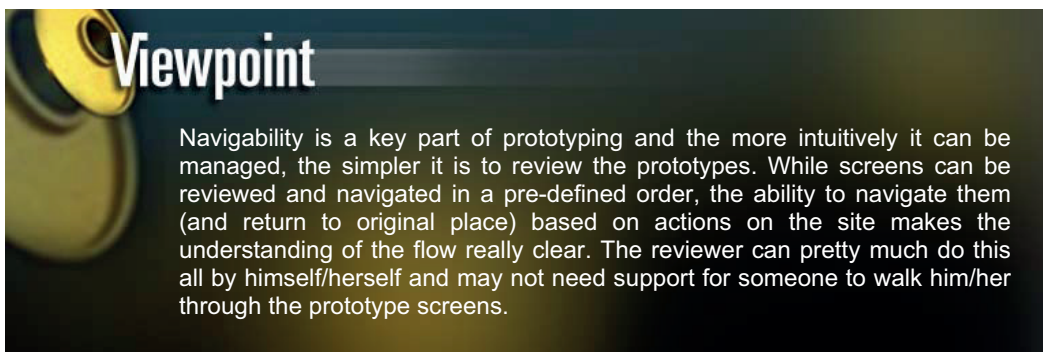
Animations are created using the SketchFlow animation panel to create a sequence of snapshots by adding a new frame to the panel and then adjusting the “scene” on the art board for that frame. You can easily adjust the transition times between frames and hold times for each frame. You can easily preview animations via the animation panel itself. An example of its usage is to demonstrate any drag and drop behavior on the site.



Navigation

No application will typically have a single screen and, hence, the ability to navigate between screens is important. The links as seen in SketchFlow map show the navigation path between screens. These are seen in the SketchFlow player and allow reviewers to go back and forth between screens. Another way to add navigation between screens is via the in-built “Navigate To” behavior. You can associate this behavior with any UI element or events on the screen and use to navigate across screens in any order. For example, you can link menu buttons to specific screens such that clicking them will take you directly to those specific pages.

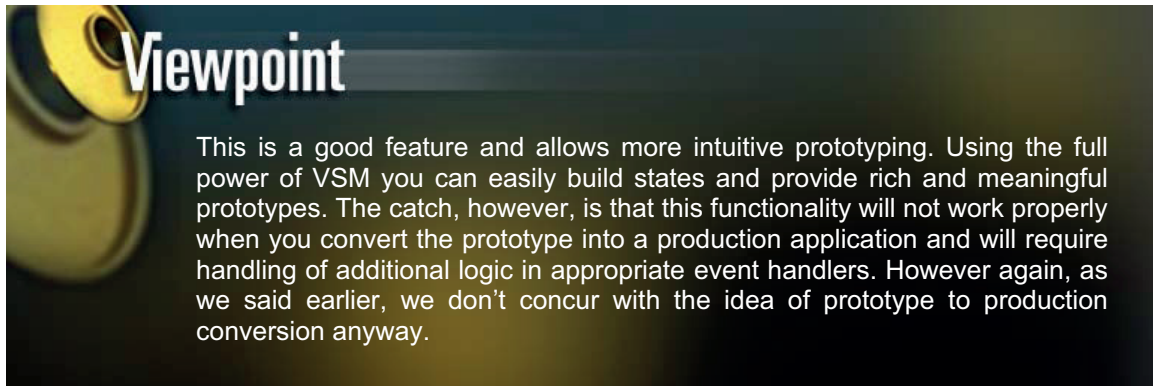
While viewing the prototype in the SketchFlow Player, we can navigate among the screens and can trigger state changes by using the Navigate panel. This functionality allows us to navigate through our prototype without adding any additional code or UI elements, even if it contains only few rough sketches or images. We can also quickly add behaviors to the UI elements (drawn on the art board) enabling us to navigate between screens or play animations.



State Changes

Another aspect in applications will be state changes. They essentially are changes happening on the same screen when interacting with elements on a screen. There are many scenarios in which creating multiple states of a single screen may be useful for creating a prototype. For example, on clicking on an element in a list of items, it needs to be added to a shopping cart.

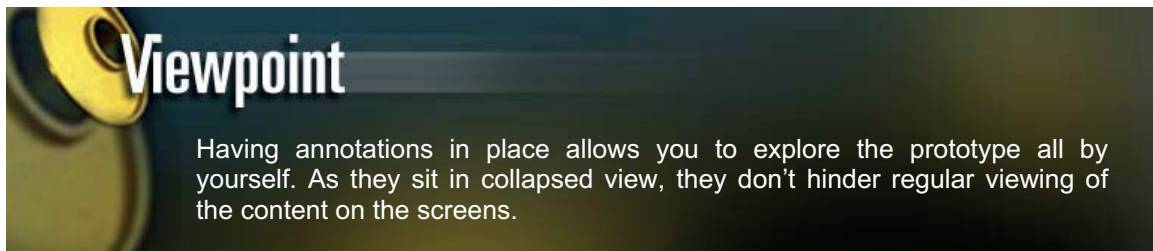
States are managed via the same VisualStateManager (VSM) feature as available with SL. The changes of states can then be connected via the “ActivateState” behavior available for SketchFlow applications. When viewing a prototype in the SketchFlow Player, you can navigate between screens and trigger state changes by using the Navigate panel. This functionality implies you can navigate through your prototype without adding any additional code or UI elements.



Annotations

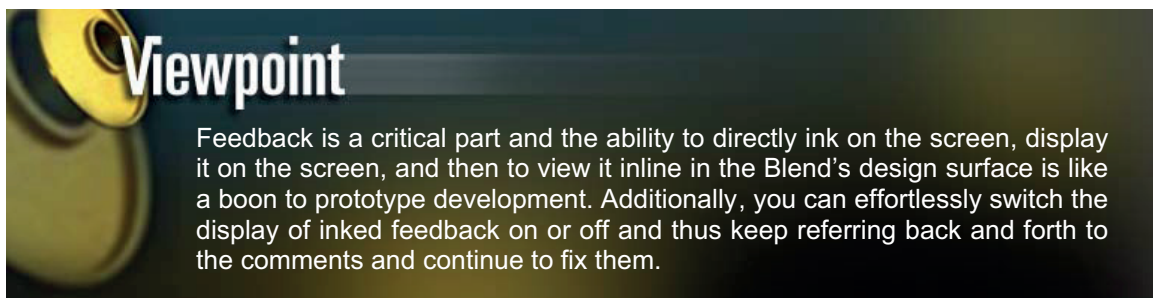
In addition to creating and modifying objects and controls on the art board, we can leave comments and notes by using collapsible annotations. These can be used to share comments with:

1. Self, acting more like TODO statements, reminding you of unfinished work
2. Each other during the prototype building phase within a team
3. The reviewer to explain some prototype specific decisions



Feedback

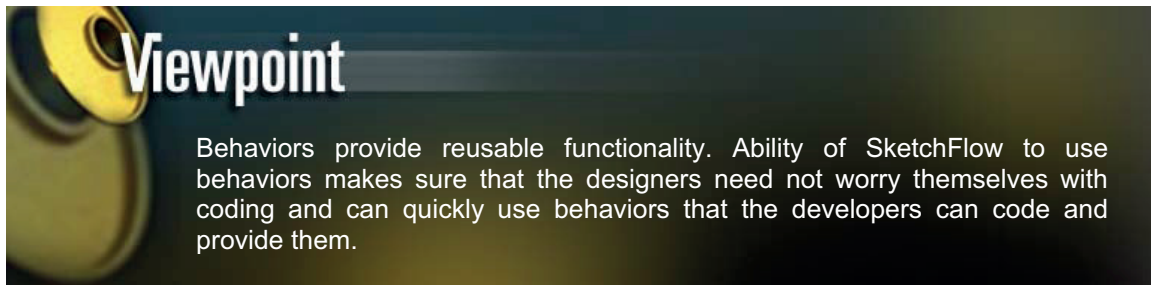
Prototyping isn't complete without discussion and feedback helps further refine the prototype before stepping into the final design and development for the application. Designer and the reviewer can share feedback for the prototype directly in the SketchFlow player using text or ink. They can also show or hide feedback for a clear view of the prototype. Feedback can then be saved as a file and later displayed in the Expression Blend design Surface. A place to type detailed feedback is also available.



Behaviors

Behaviors are built-in pieces of code that can be quickly applied on elements to create interactivity. These can be triggered on the occurrence of any event or can encapsulate multiple event triggers. Adding behaviors is similar to adding controls to the screen. These can be dragged from the asset gallery and dropped on controls for which we want to set the particular behavior. The control to which we add the behavior will behave in the same way as it would in the final application.

Blend provides us some standard behaviors which can be applied to the controls. The behaviors used most often in SketchFlow prototypes are “navigate to” and “activate state” behaviors. You can write custom behaviors as well, like drag and drop behavior.

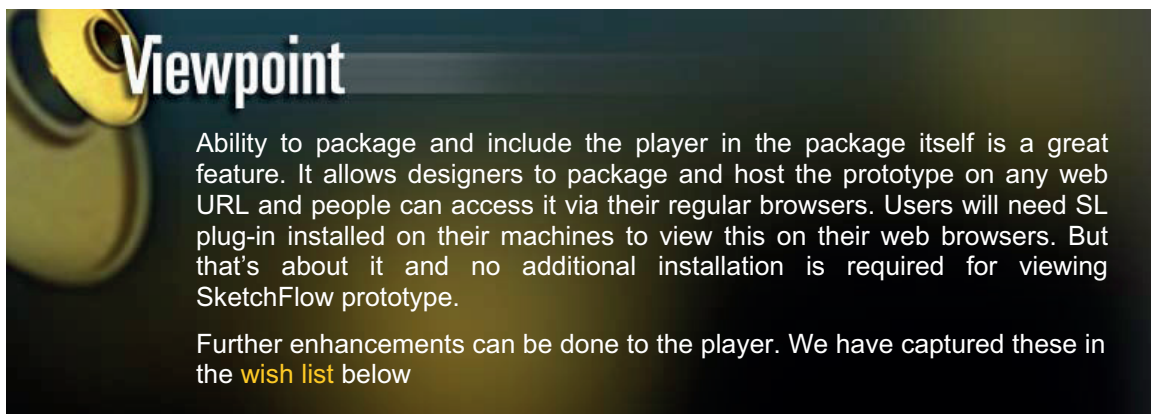


Packaging and Player

The way of distributing the prototype to the stakeholders for review and collect feedback is done by packaging the prototype. With Blend we can efficiently package our prototype. All the prototype files including the SketchFlow player exist in the package. They only need to be hosted on a website for others to access.

A simple way of doing it is to go to the File menu and select the Package (either Silverlight or WPF) project option, and give the desired name and path where the package needs to be saved.

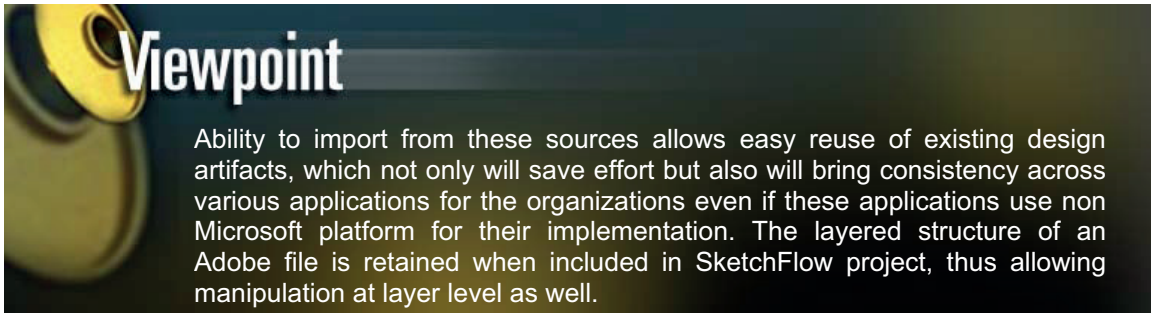
The player gets built into the package and runs in browser. It allows easy navigation of the various screens and also has a way to play the different animations that are built into the prototype. It allows users to provide feedback (discussed in Feedback section earlier).



Adobe and PowerPoint Import

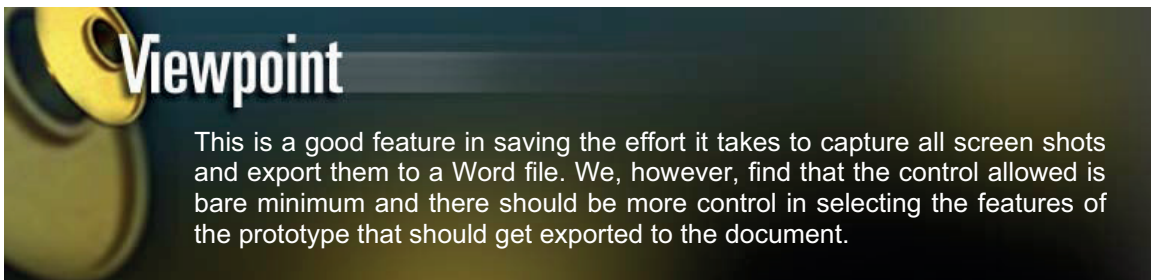
Many prototyping artifacts may get built as PowerPoint slides or as Adobe Photoshop or Illustrator files. SketchFlow allows you to easily import these files and then manipulate within Blend to build the prototypes. Blend has the ability to import a layered file, which could be an Adobe Photoshop file, an Adobe Illustrator or a Microsoft PowerPoint file. Each layer gets inserted as a separate Screen or UserControl, which on import acts just like an image. The advantage lies in the interaction designers, visual designers, and the UI developers working together seamlessly and also saving the time to recreate the control which can be simply showcased by a pictorial control.

For inserting any of the above mentioned layered file the designer just needs to click on File menu, then go to Import options, and select the desired file to be imported. On import, each screen automatically gets inserted in the SketchFlow Map and linked to the respective sequence with the main page.



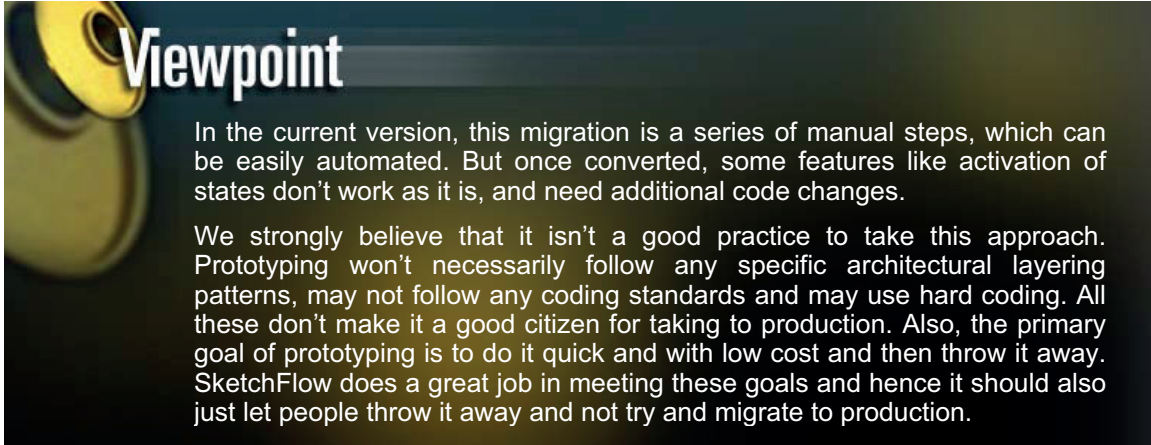
Export to Word

Another important factor in prototyping is the creation of design document for the application that can capture the flow and page level descriptions. SketchFlow has made it a trivial task for the designer. All you need to do is to export the application to Word and it will automatically create a document which includes a table of contents, a list of figures, SketchFlow Map, each screen in the application and all associated annotations.



Migrate to Production Code

SketchFlow allows you to convert the application to production code by doing some changes to the project files and changing some assembly references. Also, you can further remove the sketch styles from the control by resetting their Style property.



Viewpoint

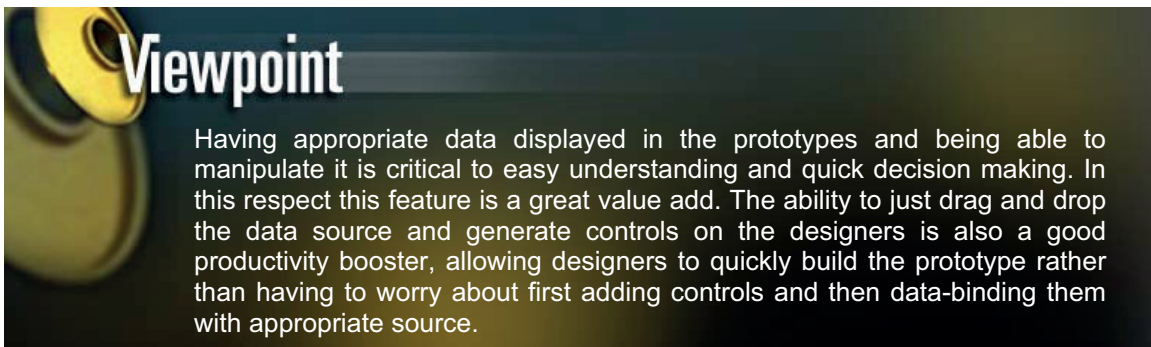
In the current version, this migration is a series of manual steps, which can be easily automated. But once converted, some features like activation of states don't work as it is, and need additional code changes.

We strongly believe that it isn't a good practice to take this approach. Prototyping won't necessarily follow any specific architectural layering patterns, may not follow any coding standards and may use hard coding. All these don't make it a good citizen for taking to production. Also, the primary goal of prototyping is to do it quick and with low cost and then throw it away. SketchFlow does a great job in meeting these goals and hence it should also just let people throw it away and not try and migrate to production.

Sample Data

When designing prototypes, data display and its manipulation is also relevant. From the styling and templating perspective also, availability of data is important. Sample data is useful when you are designing the appearance of controls that will display live data that you might not have access to at design time. Blend has made it easy for a prototype designer to create a data-driven prototype with very simple steps for a sample data source creation and then binding it to the controls. SketchFlow leverages the same functionality and allows sample data to be quickly inserted into the prototypes.

Sample data can be generated in various formats like integers, strings, web URLs, addresses, currency, images etc. Having created sample data from the Data tab in Blend, you can then drag and drop this directly on the designer surface to get appropriate controls created. Further styling can then be easily done again by using regular Blend styling and templating features.














































Viewpoint

Having appropriate data displayed in the prototypes and being able to manipulate it is critical to easy understanding and quick decision making. In this respect this feature is a great value add. The ability to just drag and drop the data source and generate controls on the designers is also a good productivity booster, allowing designers to quickly build the prototype rather than having to worry about first adding controls and then data-binding them with appropriate source.

Comparing Prototyping Techniques

There are many ways people do prototyping today. In the table below, we compare SketchFlow with some of the other often used prototyping techniques. There are prototyping tools available from other vendors like Adobe, but comparing with them is not in the scope of this document.

-  Feature full supported
-  Feature not supported
-  Feature partially supported

Feature	Images (Sketched Image)	PowerPoint	Dummy Application	SketchFlow
Visual layout	 Yes	 Yes	 Yes	 Yes
Visual design (colors, fonts, content)	 Possible, but difficult, especially to capture font and content specific details	 Yes	 Yes	 Yes
Navigation	 No Navigation Possible	 Multiple Screens can be viewed by Slide Show and links can be added to control the navigation	 Navigation Works same as in Production application. Coding is required for it	 Navigation is possible. Coding for navigating the screens is not required
Interactions	 No Interaction at all	 Page/Screen changes on mouse click	 Application is as interactive as a real production application could be. Requires tedious coding to be done	 Application is completely interactive with lesser coding required.
Animations	 No Animation Possible	 Page level standard animation is possible	 Possible. But is tedious to do	 Possible easily and quickly with an opportunity of lots of innovation
Feedback	 Have to be given as a separate document	 Have to be given as a separate document	 Have to be given as a separate document. But no	 Feedback can be given within the Blend tool and the SketchFlow Player itself. And can be viewed and packaged with it or can be shared as a separate document
Packaging	 Not possible	 It is a single PPT/PPTX document	 Can be simply Zippe Packaging possible	 Packaging is possible
Reusing other design assets	 If images, can be reused	 Can do if available as media that PPT can work with	 Can add to application and reuse, but requires effort	 Available
Document generation	 Has to be done manually	 Has to be done manually	 Has to be done manually	 Export to Microsoft word document is possible, which will have all the content table, list of figures and SketchFlow map
Simplicity to create	 Have to be drawn manually or using design tools. Requires expertise	 Simpler to hand drawn sketches	 Tedious, requires effort equivalent to a complete production application	 Simple and fast to create

SketchFlow Feature Wish list

SketchFlow in our opinion is a great product in the Expression Suite. It addresses the needs for a designer working on UI prototyping very well. Here's a list of additional features that should be added to SketchFlow.

- **Screen Deletion:** Nodes/Screens in SketchFlow Map can only be deleted by a right-click. Support of Delete key could be added.
- **Screen Deletion:** Deleting a screen from SketchFlow Map doesn't delete the same from the project as such. However, it's vice versa holds true. While we agree that deleting from map may not always mean an intention to delete from project as well, but there could be a message box providing an option to delete from project as well.
- **Adding new Screens:** While new screens can be added easily via SketchFlow map, ability to add them via File/New Item or context menu can also be provided. As of now, we can only add a new user control. An option to add a new SketchFlow Screen can also be provided.
- **Screen Naming:** Default name of the screen is 'Screen 1.xaml'. The blank space after 'Screen' in the name serves no purpose and can actually be confusing when trying to directly add "navigate to" type actions in XAML. Suggest this be set to "Screen1.xaml".
- **Screen Naming:** On similar lines when connected screens are drawn they derive their name from the previous one and we eventually end up with names like Screen 1_1.xaml, Screen 1_2.xaml. Instead, consider Screen1.xaml, Screen2.xaml, ScreenX.xaml, and so forth.
- **Screen Naming:** Renaming a screen in the map only renames the visual name of the screen, but doesn't change the file name or the class name. This can cause issues in understanding of screens as seen in the map against that seen in the project listing. This further causes issues if we try to add a "navigate to" type behavior directly in code. These visual names aren't seen in Visual Studio. Visual Studio shows only the filename. It will be good to have consistency between Blend and Visual Studio.
- **Visual Tags:** The ability to color the Screens in the SketchFlow map is fine, but we don't think the colors for connecting lines make lot of sense. These can be set independently without really having a need to match the screen colors. We recommend that these should be set automatically based on the color set for the connected screen i.e., the screen to which the line connects to.
- **Map Panning:** SketchFlow map can become pretty complex based on the size of the application and, hence, a panning control to pick and focus on specific area in the map will help. It will be useful for large and complex maps.
- **Ink and Comment Correlation:** Feedback is an important aspect and is provided in two ways –inking and typing comments. Usually the comments typed in will explain the inked part. But there is no easy way to relate the two in the current version of SketchFlow.
- **Feedback with State changes:** Similarly when there are states on a screen and some inked feedback is provided after playing some states; there is no way to figure this out when the feedback is seen in Blend. All inked feedback for a screen is loaded at once and, hence, some inked feedback may be out of place, till the time the specific state is played. There should be an ability to relate feedback with states as well.
- **Export to Word:** The export to word is a good feature, but if there is a way to select what we will like to export, it will be great. For example, annotations may mostly be relevant for the designer working in Blend or to the reviewer when viewing the application in the player and, hence, a choice to export it to the document must be facilitated. Additionally, given that the Word document may evolve into a design document, we would not always want the feedback to be embedded in the document. Finally, MS Word's hyper linking feature can be put to use by allowing cross-referencing between SketchFlow map and the individual screen details in the document for easy navigation.
- **Export Resources:** In our view, migration to production code isn't a good practice. However, when building the prototype, lot of useful resources like styles, templates, brushes, etc. may get created, which could be reused in production project. Hence, an ability to export these to a resource dictionary, which can then be included in the production project, would be a good idea.

- **Map and Player:** SketchFlow player has an option to overlay the SketchFlow map on the screen. This allows the user to pick any screen to quickly navigate to. Right now, it is a part of the feedback panel. We recommend relocating this to be a part of navigation panel.
- **Control Conversion:** While designing, Blend provides an option to change the Layout type i.e., from Grid to Canvas. Similarly, we should be able to change the control type as well. This will enable the prototype designer with an option to create a wireframe of the prototype in the beginning, in which we can denote placeholders symbolically (say by a rectangle), and later change to specific controls like say text box or list box (via context menu) as and when the design and the requirements evolve.

References

1. GUI Prototyping tools
<http://c2.com/cgi/wiki?GuiPrototypingTools>
2. The Art of UI prototyping by Scott Berkun
<http://www.scottberkun.com/essays/12-the-art-of-ui-prototyping/>
3. The New Iteration by Karsten Januszewski and Jaime Rodriguez
<http://windowsclient.net/wpf/white-papers/thenewiteration.aspx>
4. Overview of SketchFlow
http://www.microsoft.com/expression/products/Sketchflow_Overview.aspx
5. SketchFlow documentation
[http://msdn.microsoft.com/en-us/library/ee341458\(Expression.30\).aspx](http://msdn.microsoft.com/en-us/library/ee341458(Expression.30).aspx)
6. Prototyping with Blend and SketchFlow
<http://bradsramblings.com/blog/2009/09/prototyping-with-blend-3-and-sketchflow/>

Conclusion

SketchFlow is a great new tool in Expression Suite and will help a lot in building fast and more intuitive prototypes. In its initial version itself it offers lots of goodies and helps designers build interactive prototypes quickly. The effort to build these is minimal and we strongly recommend everyone to check this product out, if not already done so, and start using it in their day to day work.

About the Authors

Sakshi Sharma

She is a Systems Engineer with Product Competency Center (PCC) in Microsoft Technology Center (MTC). She has spent more than 2 years with Infosys. She has expertise on Windows Presentation Foundation and Silverlight technologies. Her current focus is user experience design and works closely on creating innovative designs with Expression Blend and Expression Design for application front end.

She can be reached at Sakshi_Sharma03@Infosys.com.

Atul Gupta

He is a Principal Technology Architect with Product Competency Center (PCC) in Microsoft Technology Center (MTC). He has spent more than 14 years with Infosys and has been working on Microsoft Technologies all along. He leads a team working on user experience creation using technologies like Windows Presentation Foundation and Silverlight. He blogs on latest technologies at: <http://blogs.infosys.com/microsoft>. Working on latest technologies is his passion and this has helped him bag the Microsoft's Most Valuable Professional (MVP) award six times in a row.

He can be reached at AtulG@Infosys.com.



For more information, contact askus@infosys.com

About Infosys

Many of the world's most successful organizations rely on Infosys to deliver measurable business value. Infosys provides business consulting, technology, engineering and outsourcing services to help clients in over 30 countries build tomorrow's enterprise.

For more information about Infosys (NASDAQ:INFY), visit www.infosys.com.