



INSTANT

Short | Fast | Focused

Axure RP Starter

Start prototyping your first Axure RP project the easy way

Amit Daliot

www.it-ebooks.info

[PACKT]
PUBLISHING

Instant Axure RP Starter

Start prototyping your first Axure RP project the easy way

Amit Daliot



BIRMINGHAM - MUMBAI

Instant Axure RP Starter

Copyright © 2013 Packt Publishing

All rights reserved. No part of this book may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, without the prior written permission of the publisher, except in the case of brief quotations embedded in critical articles or reviews.

Every effort has been made in the preparation of this book to ensure the accuracy of the information presented. However, the information contained in this book is sold without warranty, either express or implied. Neither the author, nor Packt Publishing, and its dealers and distributors will be held liable for any damages caused or alleged to be caused directly or indirectly by this book.

Packt Publishing has endeavored to provide trademark information about all of the companies and products mentioned in this book by the appropriate use of capitals. However, Packt Publishing cannot guarantee the accuracy of this information.

First published: January 2013

Production Reference: 1160113

Published by Packt Publishing Ltd.
Livery Place
35 Livery Street
Birmingham B3 2PB, UK.

ISBN 978-1-84969-516-9

www.packtpub.com

Credits

Author

Amit Daliot

Project Coordinator

Amigya Khurana

Reviewer

Panayiotis Karabetis

Proofreader

Mario Cecere

Acquisition Editor

Edward Gordon

Production Coordinator

Melwyn D'sa

Commissioning Editor

Ameya Sawant

Cover Work

Melwyn D'sa

Technical Editor

Charmaine Pereira

Cover Image

Melwyn D'sa

Copy Editor

Alfida Paiva

About the Author

Amit Daliot has vast experience in user interface and user experience design. He excels at working with companies to help mold their web services and applications in ways designed to give users a most engaging experience. During his 16 years of experience, he has defined, built, and launched new products while working from the U.S. and abroad. Due to his insatiable curiosity, Amit's career is quite diversified. Amit's wheelhouses include R&D, sales, and marketing; he holds an M.B.A. in Business Management and has expertise in creatively developing products. Amit is also an experienced software engineer with a B.Sc. in Chemistry. More information can be found on his website at www.UI-prototyping.com.

About the Reviewer

Panayiotis Karabetis is a partner at Vim Interactive in Baltimore, where he oversees IA and Ux efforts to ensure that user needs, business objectives, and technologies live happily ever after. As an Axure evangelist, he has been a presenter at the 2011 and 2012 AxureWorld conferences and is curator for Axureland, the place where Axure enthusiasts share free resources with the growing Axure community.

I'm thankful for the opportunity to contribute to a second Axure book for Packt Publishing, the first being *Axure RP Prototyping Essentials* by Ezra Schwartz. Appreciation also goes out to my team at Vim Interactive for their support and to Dr. Sarah Jalali for her patience during my reviews.

www.packtpub.com

Support files, eBooks, discount offers and more

You might want to visit www.PacktPub.com for support files and downloads related to your book.

Did you know that Packt offers eBook versions of every book published, with PDF and ePub files available? You can upgrade to the eBook version at www.packtpub.com and as a print book customer, you are entitled to a discount on the eBook copy. Get in touch with us at service@packtpub.com for more details.

At www.packtpub.com, you can also read a collection of free technical articles, sign up for a range of free newsletters and receive exclusive discounts and offers on Packt books and eBooks.

packtLib.packtPub.com

Do you need instant solutions to your IT questions? PacktLib is Packt's online digital book library. Here, you can access, read and search across Packt's entire library of books.

Why Subscribe?

- ◆ Fully searchable across every book published by Packt
- ◆ Copy and paste, print, and bookmark content
- ◆ On demand and accessible via web browser

Free Access for Packt Publishing account holders

If you have an account with Packt at www.packtpub.com, you can use this to access PacktLib today and view nine entirely free books. Simply use your login credentials for immediate access.



Table of Contents

Instant Axure RP Starter	1
So, what is Axure prototyping?	3
Wireframe prototyping is nothing less than a revolution	3
The industry is migrating from static wireframe design to prototype design	4
Assimilating Axure into your existing design flow	4
Installation and setup	6
Step 1 – Downloading the right edition	6
Standard versus Pro editions	6
PC versus Mac editions	6
Step 2 – Getting some orientation	6
Step 3 – Gearing up with the best widgets	8
Installing the Better Defaults library	8
My top 5 Axure libraries	10
Quick start – creating your first prototype	11
Step 1 – Getting familiar with the Sitemap section	11
Things to do before proceeding to the next step	12
Step 2 – Using widgets and playing with styles	14
Understanding the widget basics	14
Things to do before proceeding to the next step	16
Step 3 – Generating a prototype	17
Things to do before proceeding to the next step	18
Step 4 – Learning how to use master pages	19
Things to do before proceeding to the next step	22
Step 5 – Configuring the menu navigation widget	22
Things to do before proceeding to the next step	24
Step 6 – Adding some interactions	24
Labeling widgets	25
Opening a window with a mouse click	26
Setting widgets to the selected state	26
Prototyping a login box	28
Things to do before proceeding to the next step	29

Step 7 – Adding some logic to the interactions	30
Things to do before proceeding to the next step	31
Step 8 – Understanding dynamic panels	32
Adding a form field error message	32
Creating a tooltip box	34
Understanding the Dynamic Panel Manager pane	35
Things to do before proceeding to the next step	35
Step 9 – Adding states to dynamic panels	35
Things to do before proceeding to the next step	36
Step 10 – Publishing the prototype	37
We are done!!	38
Three common design patterns and how to prototype them	39
Module tabs	39
Why use it?	39
Hipmunk.com is using it	40
Prototyping it	40
Carousel	45
Why use it?	45
Apple.com is using it	46
Prototyping it	46
Inline field adder and a spotlight effect	50
Why use it?	50
Kayak.com is using it	50
Prototyping it	51
People and places you should get to know	55
Developing your Axure skills	55
Finding inspiration	55
Participating in forums and subscribing to blogs	56
Using some valuable services	56
Reading other Axure RP books	56
About Packt Publishing	57
Writing for Packt	57

Instant Axure RP Starter

Welcome to *Instant Axure RP Starter*.

This book has especially been created to provide you with important information that you will need to get started with Axure. You will learn the basics of Axure, get started with building your first website, and explore how to prototype common UI design patterns.

This book contains the following sections:

So, what is Axure prototyping? – Find out what UI prototyping actually is, what you can do with it, and why it's so disruptive.

Installation and setup – Download the Axure software and gear up with the most productive libraries.

Quick start – creating your first prototype – This section will show you how to quickly set up your working environment and prototype your very first website wireframe.

Three common design patterns and how to prototype them – Here, you will learn how to prototype some commonly used UI design patterns using key features of Axure. By the end of this section, you will be able to work more efficiently with Axure and will be able to deliver detailed designs much faster.

People and places you should get to know – Every brilliant project is centered around a community. This section provides you with many useful links to Axure resources.

So, what is Axure prototyping?

Face it, expressing your UX design on a set of papers no longer works as it did before.

Axure prototyping enables you to create accurate, high fidelity mockups of applications and web services quickly and with no coding skills.

But before we rush into installing the application, configuring, sketching, and designing, there are a few things worth mentioning. Rapid user interface prototyping is not a substitute for any activity in the user experience design process. You still need to make a good analysis and a detailed design for the prototype, and then implement and deploy it. Prototyping tools are here to help us better convey our design concept and minimize ambiguities when delivering the final prototype to production.

Wireframe prototyping is nothing less than a revolution

Only a few years back, when you wanted to document your product's user interface, you had to visualize each possible page, transition, and interaction on numerous sets of plain papers. Each paper showed a single web page concept. Time passed, and wireframing tools improved. Instead of plain papers, designers started using MS PowerPoint slides, Visio wireframe stencil, OmniGraffle, and various other online sketching tools. These tools simplify the design work, enable a much clearer and more accurate design, introduce easy work collaboration and sharing features, and in certain cases, even some clickable interactions. But the concept remains the same, creating static (or semi-static) pages, never knowing for sure what is the best way to sort the pages, and what is the best way to simulate the entire product's interactions while flipping from one page to the other.

From a client's perspective, evaluating a UI design was like being in a guided tour. First of all, you, the guide, had to be there. There was usually no way to understand the entire design without yourself being there to provide assistance. Your client had to follow your path, listen to your instructions, always stay in focus so that he doesn't get lost in the pile of wireframe pages, and feel the user experience of only the selected use cases that you have decided to expose him to.

Demonstrating your user interface design to your client usually sounded like—"imagine that when a visitor clicks here, he sees that..." and "when someone selects this option, he goes to this page...or that page...or that page..."—while quickly shifting from one wireframe page to another to help your client's imagination synchronize with whatever you had in mind.

The introduction of Axure, a sophisticated, fully interactive wireframing tool, disrupted the UI design industry, enabling for the first time an easy, reliable method for demonstrating product interactivity prior to its actual development.

The industry is migrating from static wireframe design to prototype design

Back in 2000, Malcolm Gladwell in his book *The Tipping Point: How Little Things Can Make a Big Difference*, described "that magic moment when an idea, trend, or social behavior crosses a threshold, tips, and spreads like wildfire."

UI prototyping reached that tipping point thanks to a very loyal community of designers, who managed to translate the idea from the specialized, static design world into a language that a majority of people can understand and easily grasp. The majority is a growing market of customers who understand the importance of a good user interface design and are no longer willing to compromise on static wireframes. A customer who is once introduced to a live prototype will no longer pay for a static one.

Axure has a rapidly growing community; thousands are using the company's website forum for exchanging ideas, solving design challenges, and interacting with the Axure development team. They are the translators, helping that shift in the market.

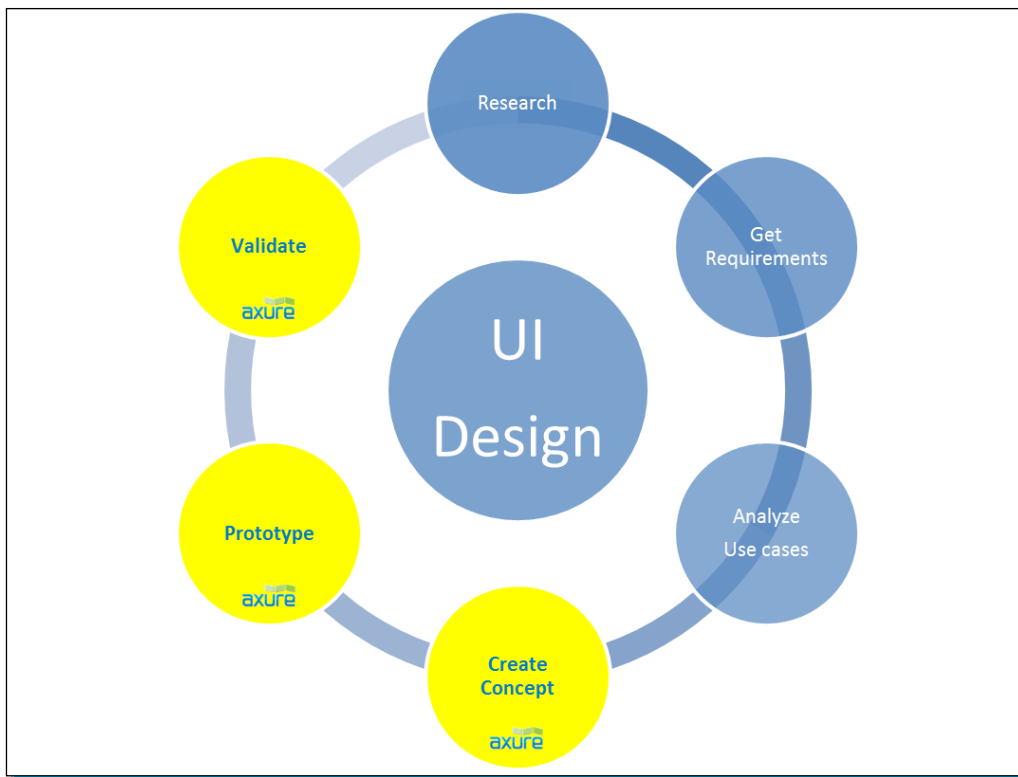
Assimilating Axure into your existing design flow

Introducing Axure into your workflow enables you to:

- ◆ Easily create flowcharts and site maps
- ◆ Design static sketches and transform them into rich mockups
- ◆ Apply conditional logic and basic animations into every design
- ◆ Showcase the project and discuss it with coworkers and clients

The value added to your workflow is indeed significant. But let's examine the full picture—the user interface design should always start with a comprehensive solution analysis, market research, persona and use case definition, information architecture analysis, and very close interaction with the product owner who is familiar with the solution's actual limitations and capabilities. This first phase is out of the scope of the Axure tool.

The next phase is sketching. That's where you start drafting design ideas and try to reach a winning concept. Sketching with Axure is possible, but you may find it easier and faster to sketch on a piece of paper, especially when you are making the very first sketches.



It's time for Axure when:

- ◆ You already know which features are to enter the design and which ones are to be left aside
- ◆ You have already defined a flowchart describing all possible paths that one may take while using the solution
- ◆ You already have sketches of the prime concepts/patterns that fit the solution and you are ready to simulate them

Making changes in Axure is relatively easy, but when the main design concept changes or when product flow is unexpectedly redefined, that's when changes become risky and you may find yourself wasting time doing Axure tool modifications instead of doing pure UI design. So you should jump to Axure only once you feel comfortable enough with the design concept.

In the next section, we will install the Axure work environment and learn how to set it up with the most powerful components.

Installation and setup

In three easy steps, you can install Axure and get it set up on your system with the most recommended add-on libraries.

Axure is a downloadable software. You can purchase it or try it for free for 30 days at the Axure website (<http://www.axure.com/download>).

Step 1 – Downloading the right edition

Before downloading the software, you should determine what the right product for you is and in which platform you are going to use the licenses.

Standard versus Pro editions

The Axure product comes in two editions – Standard and Pro. Standard edition does not support work collaboration among a team of designers sharing the same project, and it doesn't support the capability to generate automatic MS Word specifications. Besides these two capabilities both editions are identical!

So, if you are an individual, freelancer, or a part of a small team focused on prototyping but not working on the same project at the same time, Standard is the right choice for you. Additionally, you can always start with Standard and, in case needed, upgrade to Pro. The upgrade price is the same as the price difference between both editions, there is no extra cost.

PC versus Mac editions

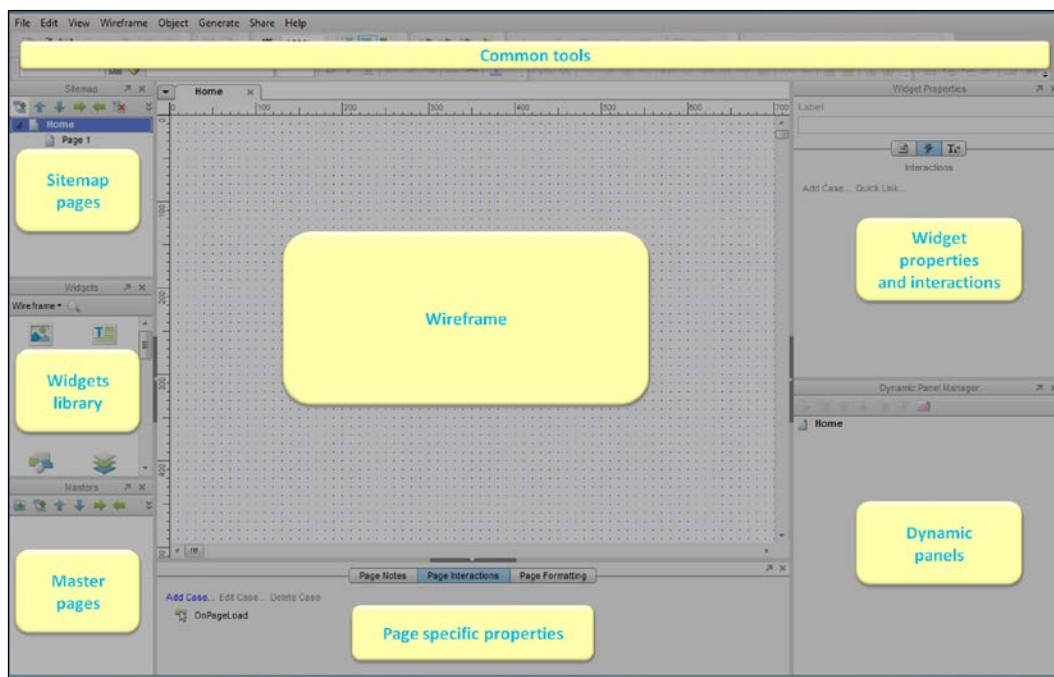
Being aware of the fact that UX designers interact with graphic designers (who are usually Mac users) as well as product managers (usually PC users), the Axure team decided to make life easier and designed the same license key to work on both, your Mac as well as your PC. It means that you can install the Axure version for Mac on one computer and install the Axure PC version on another—and the same license key will activate both.

Additionally, Axure prototypes are saved into a source file with the extension *.xrp; RP files generated in Axure for Mac edition will work on the Axure PC edition and vice versa.

Step 2 – Getting some orientation

Like many other professional tools, launching the Axure workspace for the very first time may be a bit intimidating. Numerous panels, buttons, and features will grab your attention and challenge your motivation. Hold on, let's have a quick orientation and soon all will be clear.

The following screenshot highlights the main sections of the work environment:



Your project will have pages (imagine a website with a page hierarchy). Pages are organized in the **Sitemap** tree.

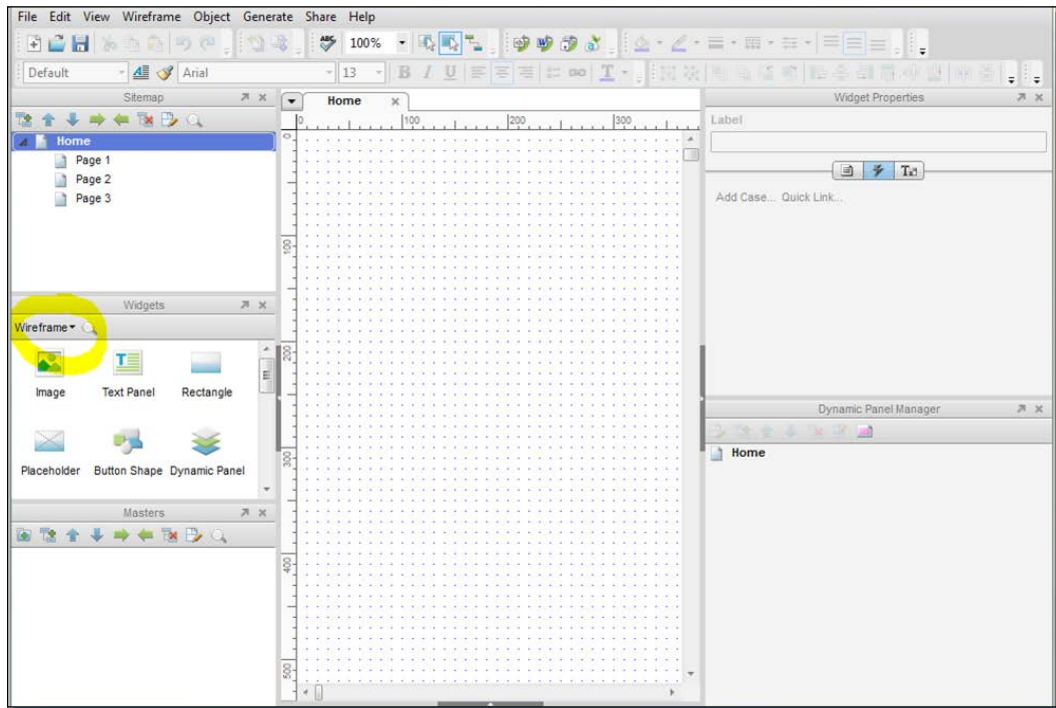
Wireframe is where you create the user interface design for each of the project's pages, and it is done by grabbing the selected UI elements from the widgets library and placing them in the relevant site map page.

Besides standard project pages, we can also use Master pages, which help us re-use design elements in multiple site map pages.

So far so good; things get a bit more complicated when we start talking about page-specific properties, widget properties, and dynamic panels. In the next section, we will discuss all of these until everything is clear.

Step 3 – Gearing up with the best widgets

Let's take a closer look at the **Widget** pane, located in the middle-left side of the tool. A strong Axure environment is one that is preloaded with the right set of widgets.



Widgets are predefined UI elements such as buttons, text fields, droplists, and checkboxes. While developing your design concept, you populate the **Wireframe** pane with the desired widgets and apply interactions between them.

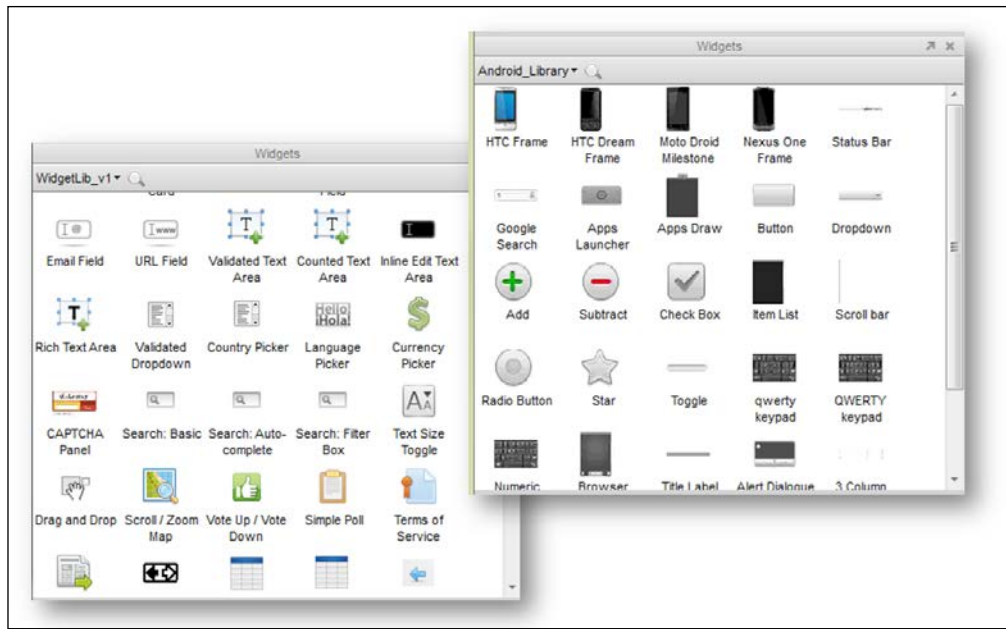
Having the right set of widgets can save time and make the prototyping work more efficient and scalable.

If there is a "must-have" widgets library for Axure, it is probably the one called Better Defaults. It is a free library created by Loren Baxter of *A Clean Design*, and unfortunately it does not come with an Axure clean install so you need to download and install it separately.

Installing the Better Defaults library

The must-have Better Defaults library can be easily found by searching the term Axure Better Defaults or directly from the following URL:

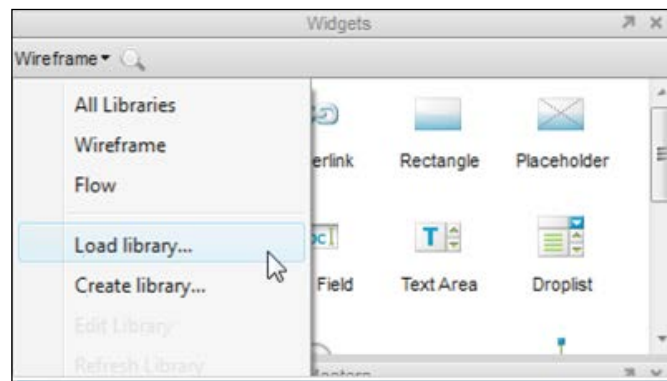
<http://www.acleandesign.com/2010/04/axure-better-defaults-library-v2/>



A widgets library is a file with the extension *.rp1ib. A library must be preloaded into the Axure environment in order to view and use its widgets.

In order to load a new library, click on the droplist in the Axure **Widgets** pane and select the **Load library** option.

Browse and find the *.rp1ib file, and load it. The new library will be accessible immediately, and will remain in your Axure environment until you decide to unload it (from the same droplist).



Other useful libraries include smart phone objects and templates, tablets, design patterns, and icons. The most comprehensive set of libraries can be found on the Axure website (<http://www.axure.com/download-widget-libraries>). Almost all libraries are free of charge.

More resources for great widget libraries can be found in the *People and places you should get to know* section of this book.

My top 5 Axure libraries

It is very tempting to load every possible library you find into your Axure environment. I can promise you though, that you will eventually use only 2-4 libraries, while the rest will only stay there, making your Axure boot up slowly.

The following is a list of libraries I commonly use:

- ◆ **Better Defaults:** It is a must-have library. It can be downloaded from this link: <http://www.acleandesign.com/2010/04/axure-better-defaults-library-v2/> (refer to the preceding paragraph for the full description).
- ◆ **Flow Library** (built in the Axure installation): We use it prior to designing the prototype itself. It is used for detailing the flow of the solution, and among other things, it is a very powerful tool for defining the scope of work between you and your client (whatever is included in the flow diagram will be included in the prototype, and nothing more).
- ◆ **WidgetLib_V1** (by Ari Feldman): It is a large set of common, highly structured widgets; most of them are fully dynamic. It can be downloaded from the following link: <http://www.widgetworx.com/widgetworx/widgetlib/>
- ◆ **iPhone-UI** (by Paul Sizemore): It contains a large set of iPhone UI widgets that does a good job of covering the common elements. It can be downloaded from this link: <http://paulsizemore.com/axure-iphone-widgets-and-library/>.
- ◆ **MyLib:** This is my personal custom widget library. It is easy to make one of your own; see the instructions on the Axure website (<http://www.axure.com/custom-widget-libraries>) and some guidelines for creating a professional grade custom library at http://axureland.com/axure_blog/entry/axure_widget_library_best_practices.



Once you complete a basic training for Axure, it usually takes no more than 50 hours of prototyping to feel like you are ready to abandon all other tools and start delivering shiny prototypes to your clients. No need to panic, 50 hours is just one (small) project workload.

Quick start – creating your first prototype

By the end of this section, we will have a basic website prototype. We will use the most common widgets, play with styles and positions, and define some interactions. Throughout these steps you will gain enough knowledge to prototype much more complicated design concepts.

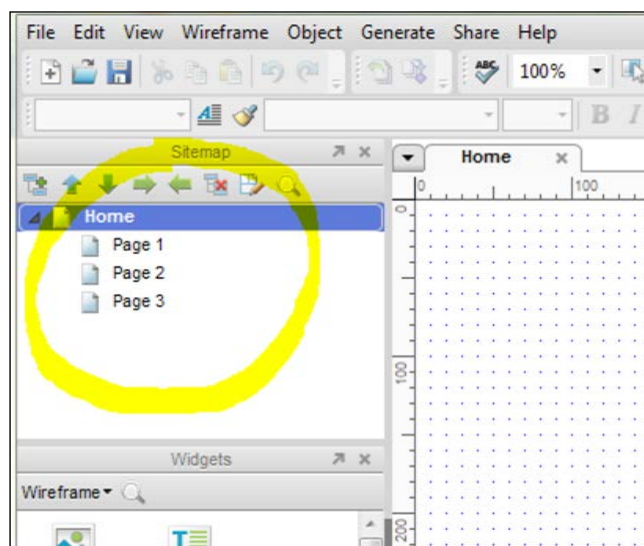
It is worth mentioning that each step is based on the successful completion of the previous one. So no shortcuts please.

Step 1 – Getting familiar with the Sitemap section

The **Sitemap** pane is where you organize the web page hierarchy. I have to say that this is the most intuitive pane in Axure, so step 1 will be easy to complete.

You can drag and drop pages in the site map or use the arrow buttons in the **Sitemap** toolbar to reorder pages. Go ahead and explore these controls.

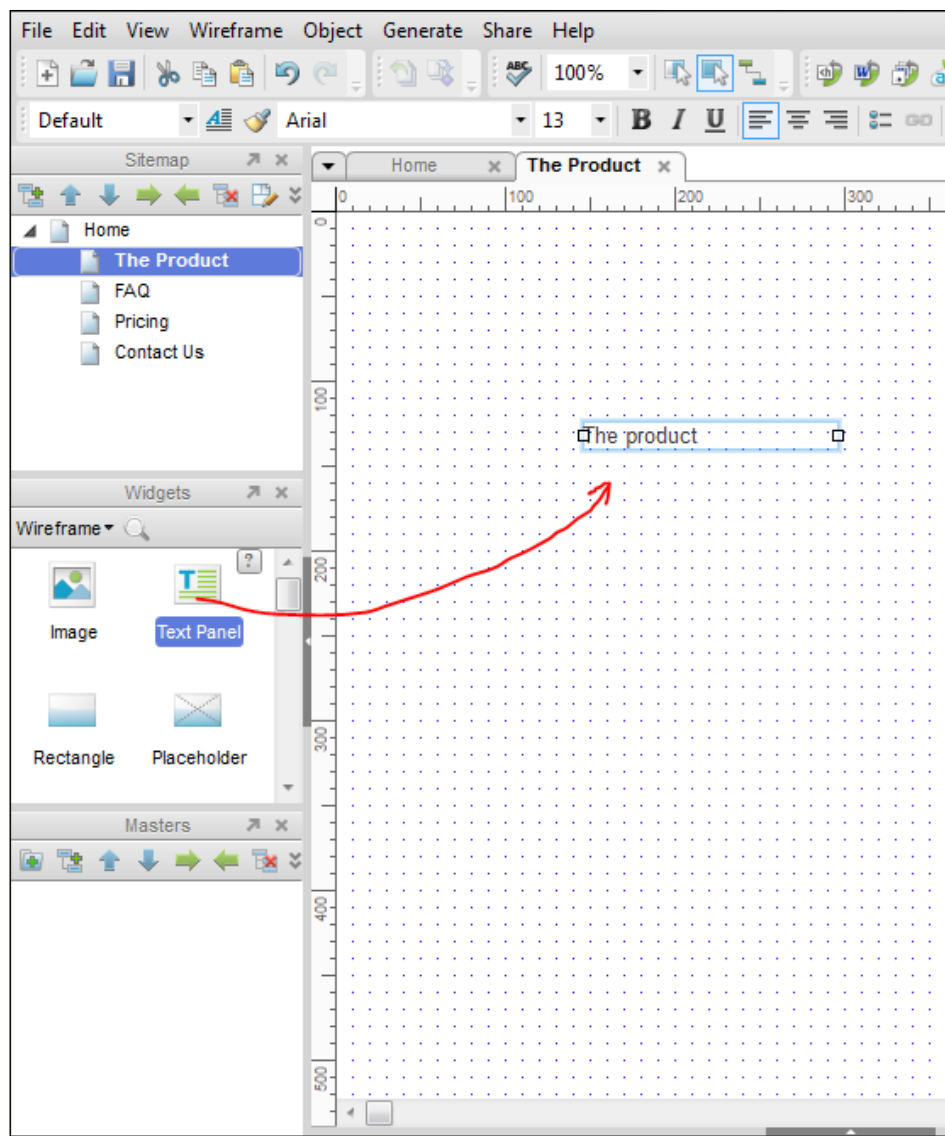
The top page (**Home**) is where the prototype flow begins. A typical site map in Axure has a **Home** page at the top and a set of child pages that the user can browse through after landing on the home page.



Things to do before proceeding to the next step

Three tasks to accomplish before the next step, where we will start playing with widgets and styles, are as follows:

1. Build the following site map elements:
 - A page named **Home** at the top
 - Four child pages under **Home**, named as follows:
 - ◆ The Product
 - ◆ FAQ
 - ◆ Pricing
 - ◆ Contact Us
2. Double-click on each page that you just created and drag a **Text Panel** widget onto it (drag it from the **Widgets** pane into the large **Wireframe** pane). This will help us, later on, to easily designate each page when we browse from one page to another.
 - Write the relevant page name (Home, The Product, and so on) in each text panel you just dragged. Notice that the editing of the page content is done by first double-clicking on the page name on the **Sitemap** pane. Refer to the screenshot that follows to see how your sitemap should look:

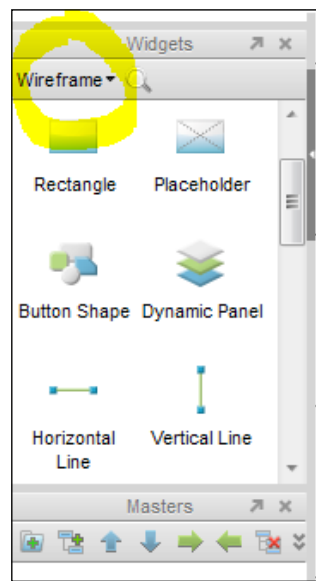


Besides the topmost page (where the prototype flow begins), the page order and indent levels have no functional implication. You can always change their order later. If you link to a page and later on change the page's name or location, the link will be automatically updated.

Step 2 – Using widgets and playing with styles

Widgets are common UI elements. While creating a prototype, you can locate an element that fits well into your design (such as an input field, checkbox, button, or simply a rectangle) and drag it from the **Widgets** pane onto your workspace.

Widgets are grouped in libraries. Click on the **Wireframe** drop-down list and explore the available widget libraries installed in your Axure environment. You will find the **Wireframe** library, which is a very basic set of UI elements. You will also find the **Flow** library, which is a widget set usually in use prior to prototyping, when creating flowcharts. You will also see the **Better Default** library—a must-have library that doesn't come with Axure. (Can't find it there? Go back to the *Installation and setup* section, and install it now. We will need it for the following steps.)

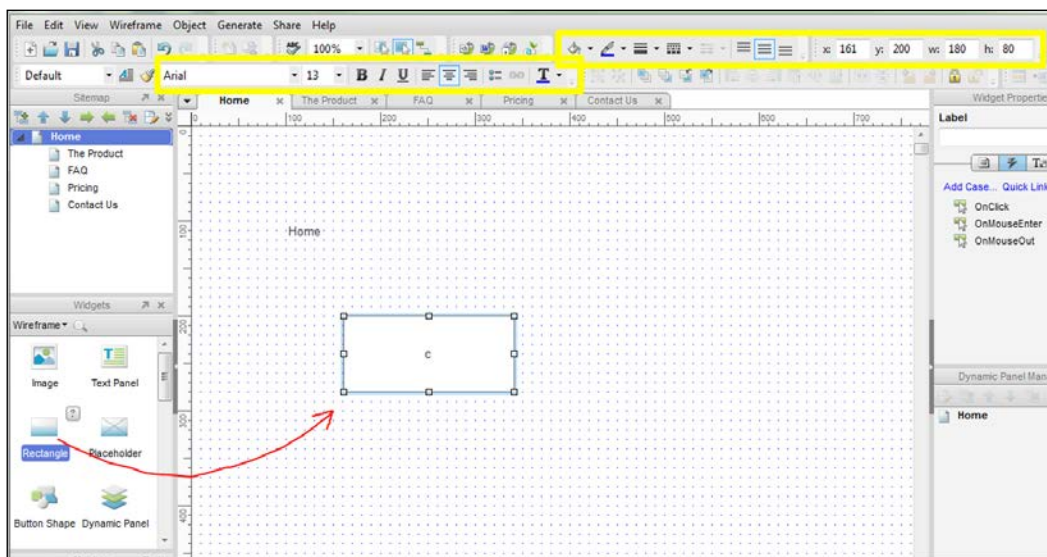


Understanding the widget basics

Go ahead and open the home page for editing (double-click on **Home** in the site map). Drag the **Rectangle** widget from the **Wireframe** library into the large workspace area. Take a look at the tab name on top of the pane to make sure you are indeed editing the home page.

Start playing with the set of style control buttons marked in yellow in the screenshot that follows. Change the rectangle font, color, line, pattern, and alignment. Also examine the widget position and size values. Make sure you recognize all the control buttons and the way they shape the widget.

The **Rectangle** widget is a widely used element, so whatever you practice here applies in the same way on most widgets that you will come across later on during your design work.



Next, let's right-click on the widget and explore some advanced menu options. Some options may be trivial but others may need some explanations.

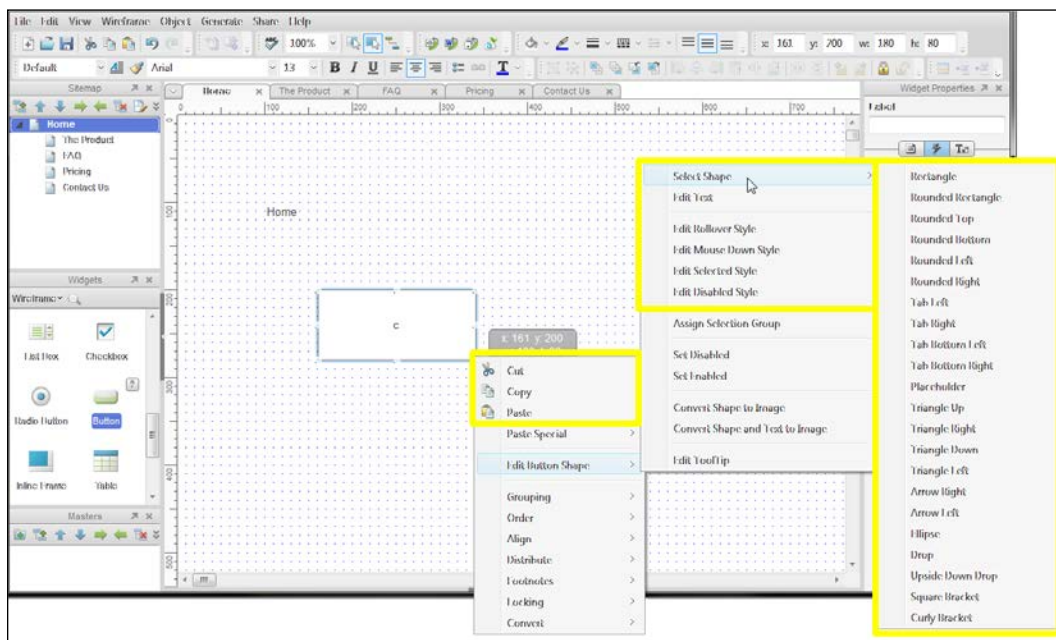
- ◆ **Rollover style:** If you want a widget style to change upon mouse rollover, that's where you configure it. (Select and see that you are familiar with all the style controls there.)
- ◆ **Mouse Down style:** This is the style that the widget gets when you click and hold the mouse button on it.
- ◆ **Selected style:** It is very common to assign a "selected" and "unselected" styles to widgets. Think about the navigation tabs of a standard website, at any given time there is only one "selected" tab with a prominent color while all others have a different, "unselected" color style.

- ◆ **Disabled style:** This is the style that a widget gets when it is in the "disabled" mode. This style set is quite standard (grey button, grey input field, and so on). In the Better Defaults library this style is already predefined for all widgets so you will hardly ever need to modify it.

Things to do before proceeding to the next step

The two tasks to accomplish before the next step, where we will generate our first live prototype, are as follows:

- ◆ Make sure you feel comfortable with the widget style controls and with all the right-click menu options marked in yellow (refer to the screenshot that follows).
- ◆ Save your work environment and identify the Axure source file created (it has a *.rp extension). We will continue working on this file for the next few steps.



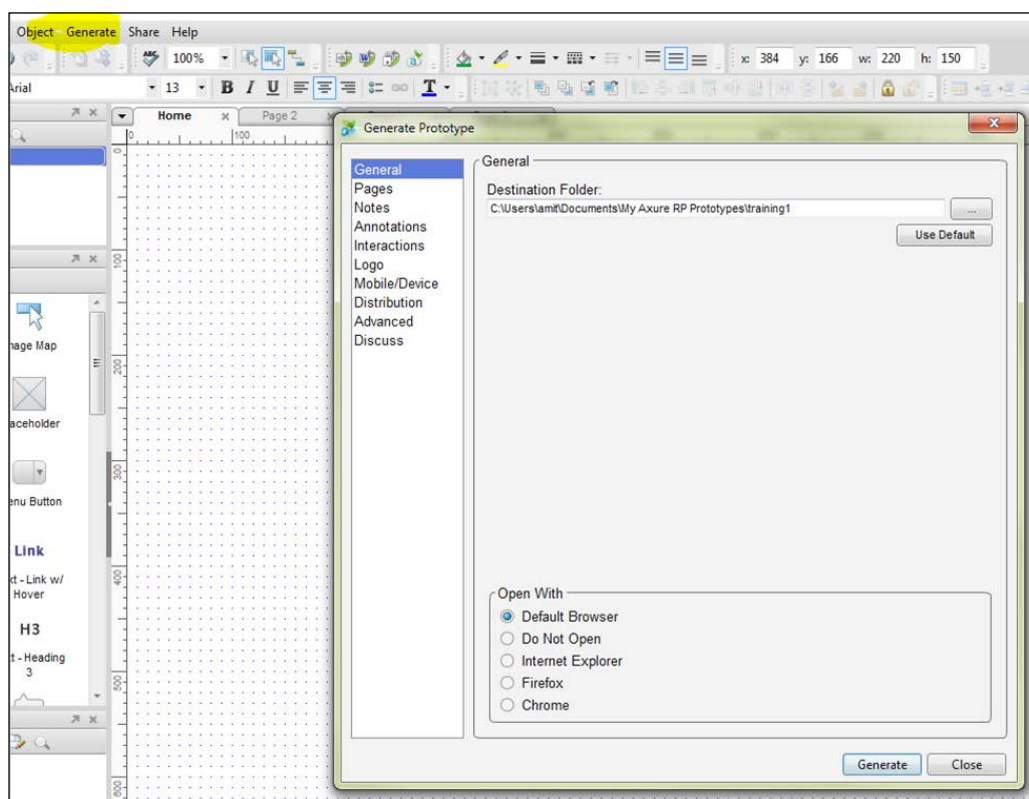
From now on, we will no longer use the default **Wireframe** library, which is a bit too limited. All widgets in the following steps are taken from the Better Defaults library. Make sure you already have this library installed (don't see it? Go back to the *Installation and setup* section, and follow the download process).

Step 3 – Generating a prototype

In this step we will generate a prototype with the click of a button. But before that, let's understand what is happening behind the scenes while making a prototype.

The Axure tool takes your design, comprised of all these widgets, site map pages, and interactions, and generates hundreds (and sometimes thousands) of HTML pages. Each time you make a change in the design, you need to regenerate all or a part of the code so that the prototype will be able to reflect the design changes.

Select the **Generate** menu option and click on **Prototype**. A dialog box will open. The only thing we need to pay attention to at this stage is the **Destination Folder** path (that's where the HTML pages will be created). Right now we do not care about any of the other options in this dialog box. Click on the **Generate** button.





If your default browser is Chrome, you will probably notice that a Chrome plugin is required in order to view the prototype. There is no way to work around this. Simply follow the instructions and install the plugin. It's a one time bother.

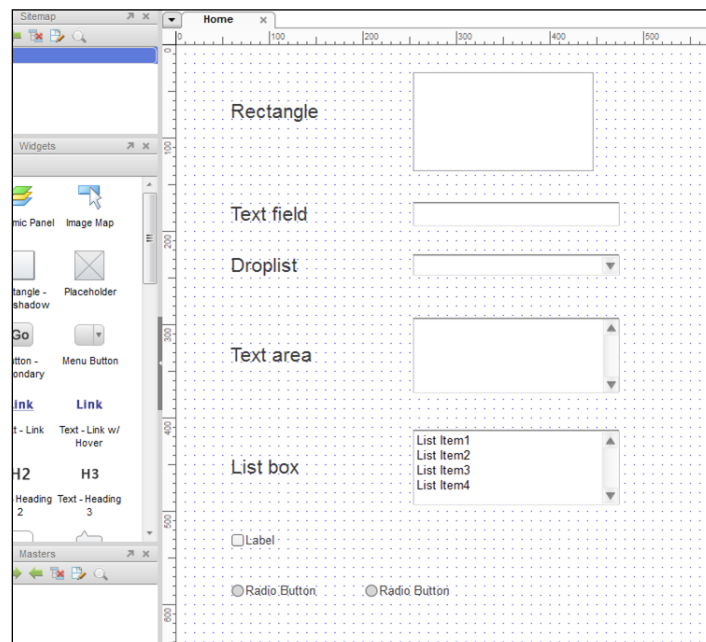
Right after generating the HTML code, Axure opens the default browser and you can see the very first page in your Axure design (the topmost in your **Sitemap**).

It is supposed to display the rectangle widget (with whatever style you gave it). Notice the control bar at the left side of the prototype. Click on the site map pages and see how the text you placed in the Text Panel widget appears on the page you have just selected.

Things to do before proceeding to the next step

The three tasks to accomplish before the next step, where we will learn how to reuse design elements, are as follows:

- ◆ Drag seven more basic widgets to the home page (see the specific widgets in the screenshot that follows).
- ◆ Some of these widgets may have a unique right-click option. Explore them.
- ◆ Keep on experimenting with the widget styles and right-click options, and constantly regenerate the prototype to examine the changes made.





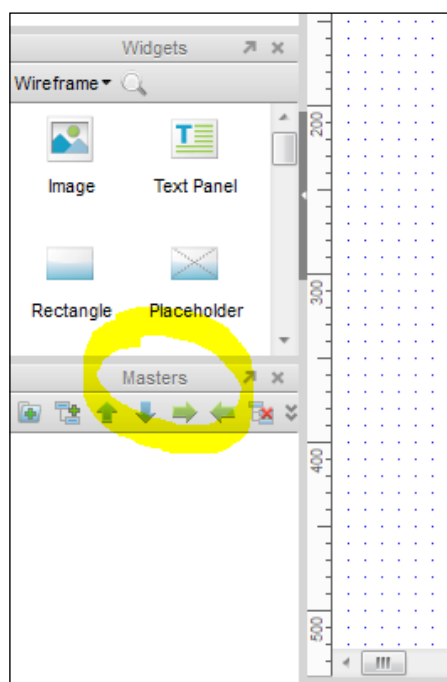
After examining the two **Radio Button** widgets, you probably noticed that something is wrong. Selecting one radio button does not unselect the other. To fix that, select them both and right-click on them. You will find the **Assign radio group** option on the right-click menu. Try it!

Step 4 – Learning how to use master pages

On most websites, you will find a certain element that is duplicated on all pages. The most common example is the page footer area. A page footer has the same design and same functionality, no matter which page of the website you are on.

The idea of a master page is to create a specific design only once (a "master") and then add that design to many web pages. From there on, every change you make in the master page will automatically be reflected in all the pages having this master.

Let's see how this can be done in Axure. Take a look at the **Masters** pane:



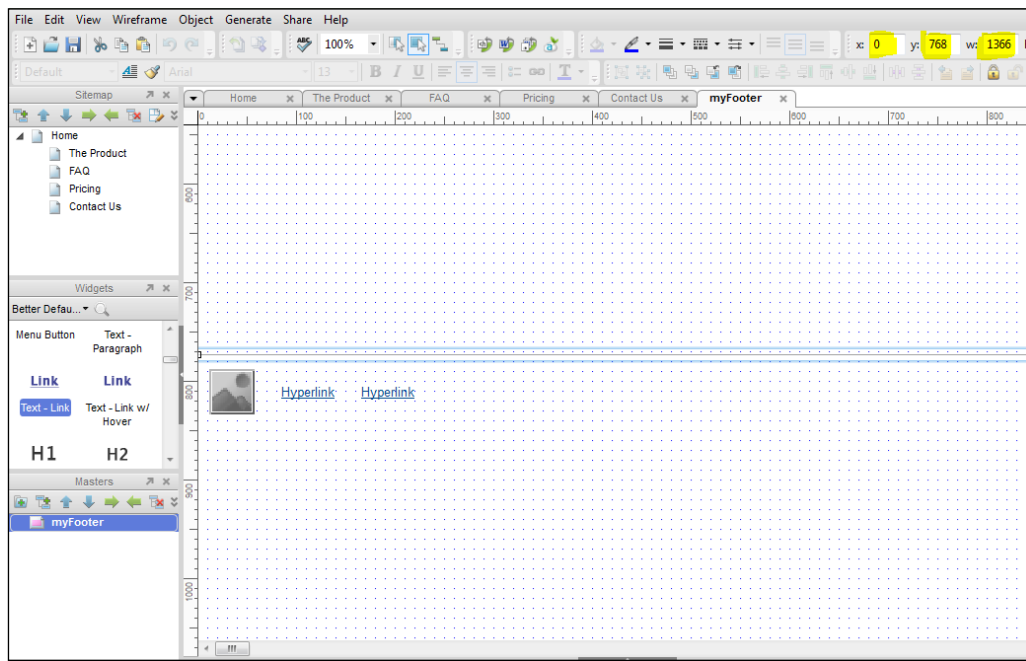
Now let's create a footer for our website. We want it to be a master page and so we need to create it in the **Masters** pane and not in the **Sitemap** pane.

Go ahead and click on the "Add master" icon to create a new master page. Rename it to `myFooter`.

You might accidentally click on the leftmost "Add folder" icon instead of the "Add master" icon. UI is a bit confusing here but now you will surely remember the right button. Delete the folder and let's move on.

Double-click on your new footer master page to edit it, and do the following:

1. Drag a horizontal line widget (from the **Better Defaults** library). You can position it at the (x,y) coordinates of (0,768) and give it a width of 1366 (remember the location and size parameters on the toolbar?). By the way, 1366 x 768 is the most common screen resolution used today.
2. Drag an **Image** widget and two **Text - Link** widgets and place them under the horizontal line.
3. Now let's use **myFooter** in all the site map pages by right-clicking on it and selecting **Add to pages....** Check all the pages in the site map.
4. Generate the prototype and make sure you see the same footer in all site map pages. If you make a change in **myFooter** it will be reflected in all pages. Try it!



The master page feature can be even more powerful after learning one more thing. Let's assume that you created a master page, but in a specific site map page you would like to have it in a different position! For instance, consider a (very common) case where your master footer is

located at 768 pixels while one specific page content ends at 900 pixels. You do not want the master footer at 768 to overlap with your long page content.

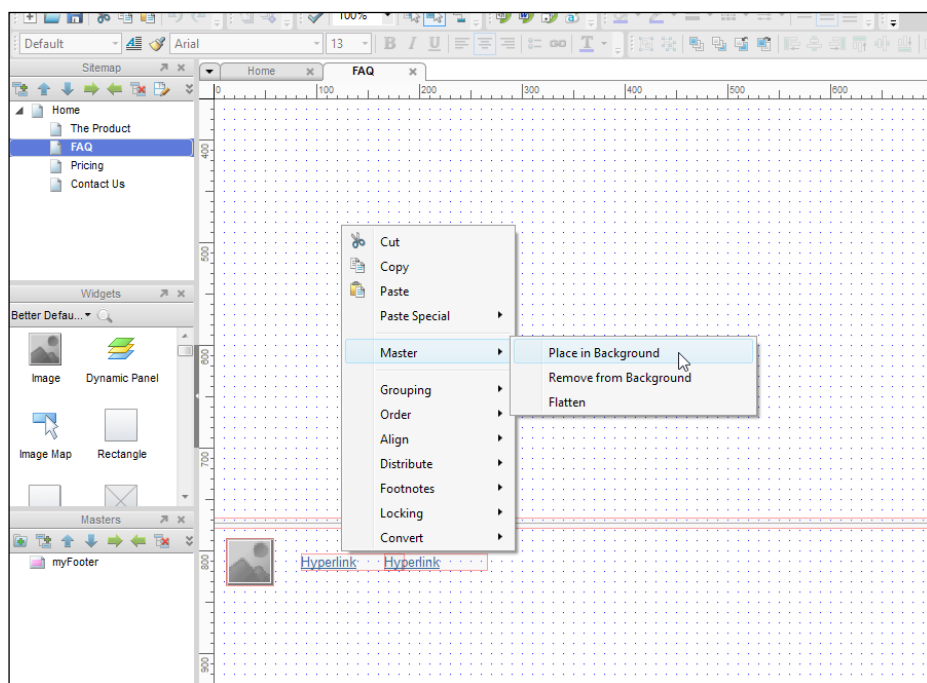
We need a mechanism to relocate the entire master page design only on a specific page.

Let's do exactly that on our **FAQ** page. Double-click on the site map's **FAQ** page, scroll down, and find the footer design (didn't find it? Make sure you checked the **FAQ** page on the **Add to pages** menu option of **myFooter**).

On right-clicking the pink footer design, you will discover these three important actions under the **Master** option:

- ◆ **Place in Background:** This is the default mode. Objects will be at the exact same position as designed in **myFooter**.
- ◆ **Remove from Background:** This will disconnect the association with the **myFooter** master position. You will now be able to relocate the footer anywhere on the site map page. Note that the design changes in **myFooter** will still be reflected in the relocated footer.
- ◆ **Flatten:** This will totally disconnect the association with the **myFooter** master. Changes in the **myFooter** master will not be reflected in the flattened footer.

Perform a **Remove from Background** action on the FAQ footer. Drag it down the page so it will be located lower, at $y=900$.





Try to focus only on the features specifically described in these steps. It is indeed tempting to try the nearby features and shortcuts ahead of time but usually it only leads to confusion.

Things to do before proceeding to the next step

These are two tasks to accomplish before proceeding to the next step (where we will add a navigation bar to our website):

- ◆ Flatten the footer master in the **Pricing** page and modify it a little bit.
- ◆ Make a small change in the **myFooter** master page (add text or something). Examine the site map pages and verify that the change is reflected in the **FAQ** page but it is not reflected in the **Pricing** page.

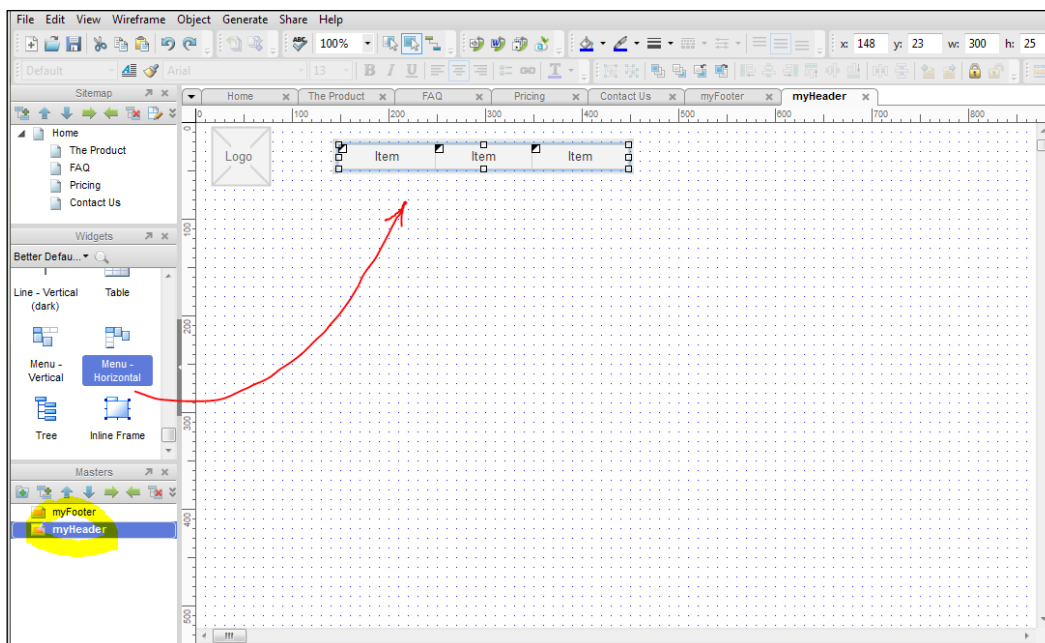
Step 5 – Configuring the menu navigation widget

In this step, we will add a second master page, which will include our website's header components – logo image and a horizontal navigation bar.

First, let's add a new master page and name it **myHeader**, then drag the widget **Menu - horizontal** into it (this widget can be found in the **Better Defaults** library).

Drag also the **Placeholder** widget and put it on the left side of the navigation bar to designate the site's logo location.

Right-click on the **myHeader** master page and add it to all pages in the site map.



The menu navigation bar widget has some added complexity worth checking.

Select one of the three items (tabs) in the navigation bar and examine its right-click menu options. You can see these actions:

- ◆ **Style changes actions:** Rollover style, selected style, and padding
- ◆ **Structure change actions:** Add and delete menu items and a drop-down menu option

The navigation bar widget comes with predefined styles for selected state and rollover state. You will find purple fill color for selected state and light blue for rollover state (check this).

Let's configure the navigation bar widget to fit into our website structure as follows:

1. Since we have five pages in the site map (including the home page), we need to add two more items to the navigation widget. Add them.
2. Name each of the five items according to pages names (Home, The Product, FAQ, and so on).
3. Generate the prototype and examine the navigation bar.

Things to do before proceeding to the next step

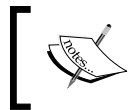
Before we make our first interactions, let's make sure we are most comfortable with widgets, masters and site map pages:

- ◆ Change the rollover style's fill color of the navigation bar items to use a color other than the default light blue one. (Have you noticed the **preview** checkbox? Try it.)
- ◆ Erase all the widgets you played with during step 3 from the home page. We no longer need them.
- ◆ Click on **Generate**. Check the rollover style new color.

Step 6 – Adding some interactions

Interactions are what make Axure exceptional by transforming a static design into a live prototype.

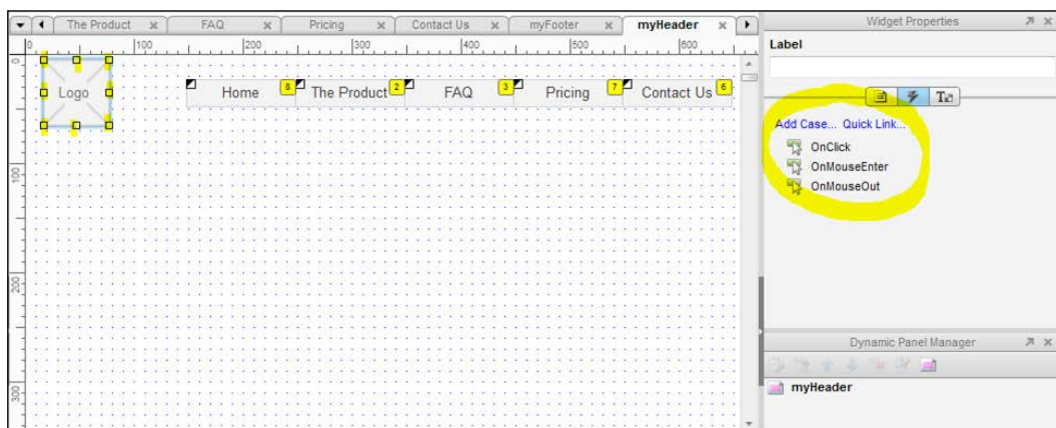
So far we used Axure in its basic, static form—we sketched, selected widgets, and created a site map architecture. We didn't go much further from designing on a piece of paper, MS PowerPoint, or Visio. It's now time to make a leap forward!



This step is packed with a lot of instructions and details but it deals with the core of Axure. So take it easy, and do not rush to complete it. The later steps will be much lighter.

The interactions pane is where all the fun begins. For every widget selected, you will find that there is a list of conditions. All you need to do is assign a relevant consequent action to a condition.

For example, a mouse click is an event and the opening of a page may be a consequent action for that click event.



The most commonly used events are:

- ◆ **onClick**: An event when a mouse click is performed on the widget.
- ◆ **OnMouseEnter**: An event when the mouse cursor enters the widget boundaries (think of a hover state of a button).
- ◆ **OnMouseOut**: An event when the mouse cursor exits the widget boundaries (think of a tooltip box disappearing when you exit a question mark icon boundary).
- ◆ **OnChange**: This one is relevant only for widgets with multiple selections, such as drop-down lists. Each time a user is changing the widget's state, this event is triggered.

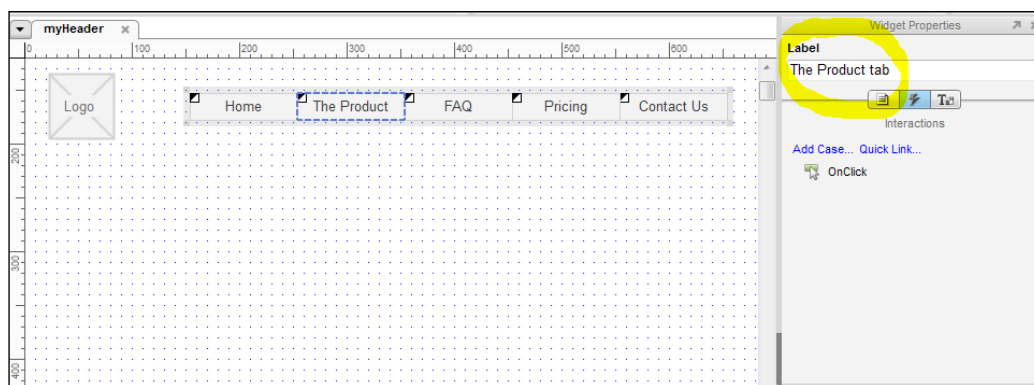
Open the **myHeader** master page for editing and let's start exploring.

Labeling widgets

Before we assign events and actions to widgets, remember that there is an important step to be followed.

Notice the input field called **Label** in the **Widgets Properties** pane. Get used to putting a descriptive name for each widget you create here. In the screenshot that follows, I used **The Product tab** to represent the product item widget in the navigation bar. Later on, you will see how essential this label becomes.

Now, label all the other navigation items. Always use unique identifiers, and don't forget to label the logo placeholder as well (it is also a widget).

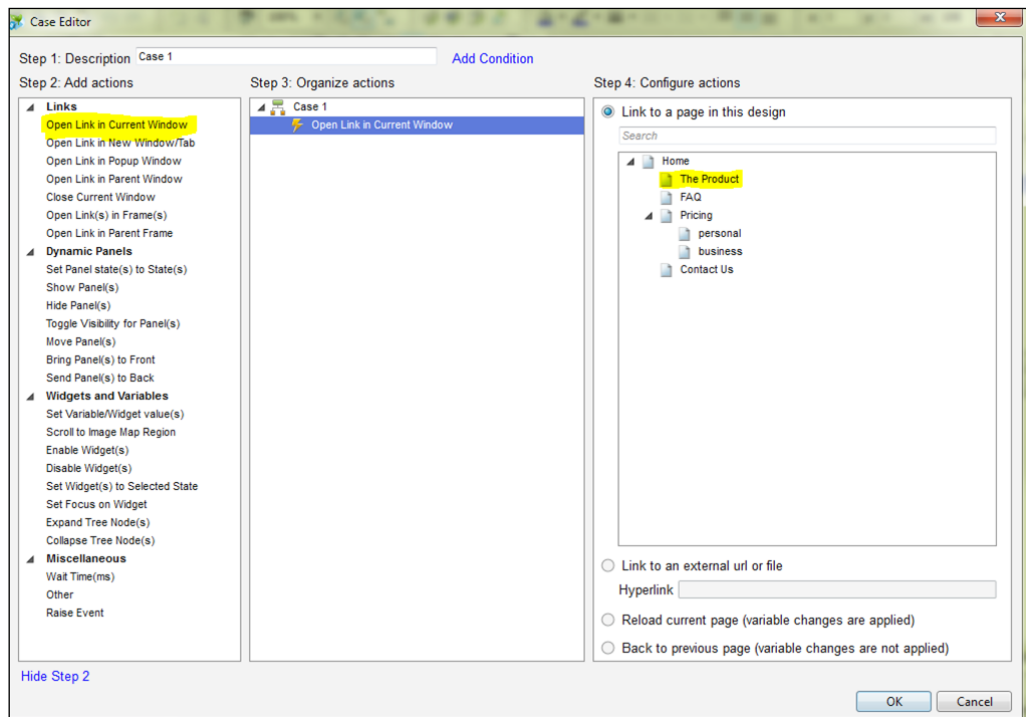


Naming conventions are beyond the scope of this book. Just keep in mind that they become crucial when dealing with large, shared projects where several people are involved, and potentially, documentations may be required. Read this article to get the idea: <http://onartandtech.blogspot.co.il/2010/11/domain-agnostic-naming-convention-for.html>.

Opening a window with a mouse click

This is probably the most common type of interaction. Opening a specific page following a mouse click event:

1. While editing the **myHeader** master page, select the **The Product** tab item in the navigation bar widget and double-click on the **OnClick** event in the widget's interactions pane.
2. The **Case Editor** screen appears (see the screenshot that follows). Select the **Open Link in Current Window** action and configure the page you want to open following a mouse click (in this case it is obviously the **The Product** page in the site map). Notice that there is also an option to open the page in a new browser tab, in a pop-up window, or even to open an external URL.



3. Configure the **OnClick** events actions in the same way, in all other navigation bar tabs, and generate the prototype. Verify that clicking on a navigation tab opens the correct site map page.

Setting widgets to the selected state

Earlier in this section, we discussed the selected style of a widget. It is a specific style that becomes valid only when you set the widget to a state called **selected**.

In our website prototype, we will set a navigation tab item into the selected state once the visitor opens its related page. For example, we would like to set the **FAQ** tab to the selected style only when the **FAQ** page opens (and keep it unselected if any other page is open).

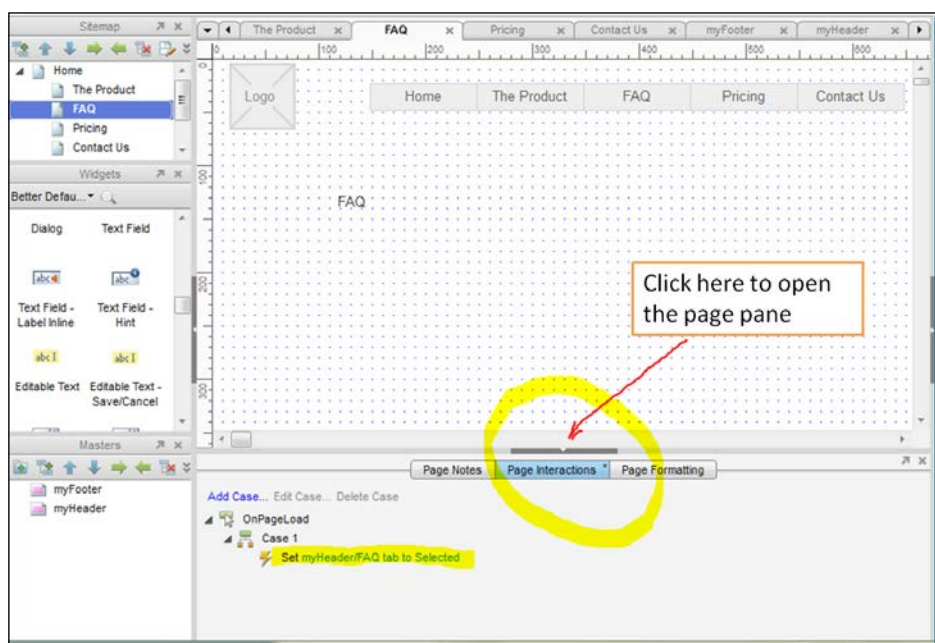
To do that we need to get familiar with yet another control pane: the **Page Interactions** pane. This one is located under the main wireframe. It may be a bit tricky to open it, see the screenshot that follows.

This is how a widget is set into the selected state on page load:

1. Let's start with the home page. Double-click on **Home** in the site map for editing.
2. Open the **Page Interactions** pane and double-click on the **onPageLoad** event that appears there. This is where you configure the actions to be performed each time this specific page is loaded. By the way, make sure the site map page is highlighted and not the **myHeader** master page (this is a very common mistake).
3. In the **Case Editor** window that opens, click on **Set Widget(s) to Selected State** and find the **Home** tab label (did you remember to label this tab item earlier?).
4. Repeat all the preceding steps for every page in the site map. This will configure the prototype to activate the selected style upon each page load. Generate and test it.



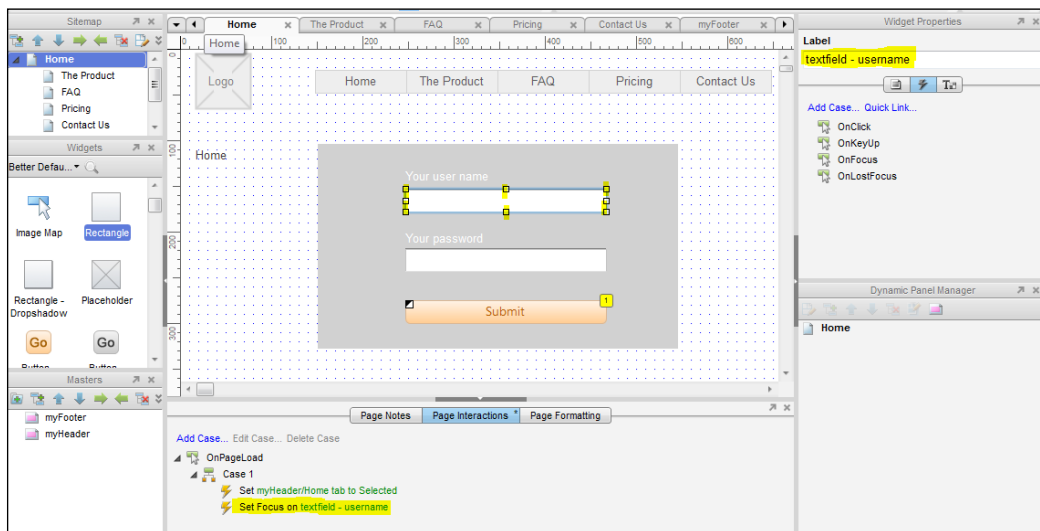
The **Case Editor** window has a filter box helping you find widgets faster. Make sure you follow some sort of naming convention, and you will realize that this feature is a great time saver.



Prototyping a login box

A login box is something you will most likely design frequently. Prototyping such an element will help us explore some new actions. Follow these steps to create one:

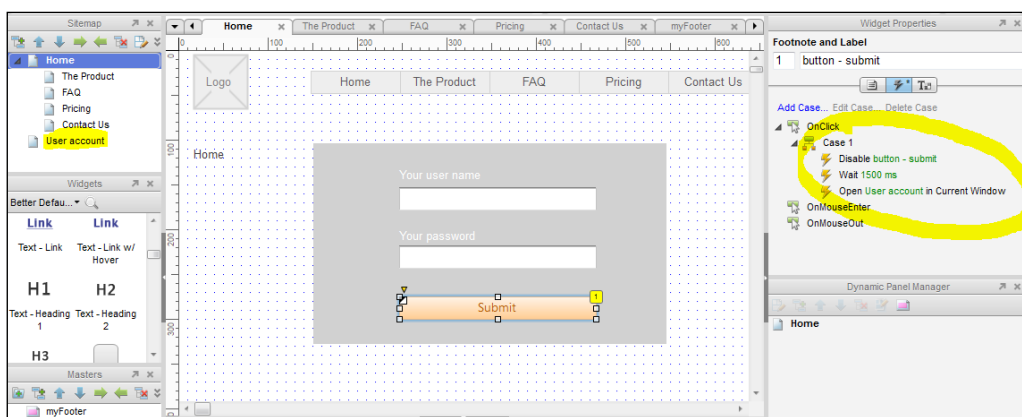
1. Double-click on the **Home** page on the site map for editing, and drag into it the following:
 - One **Rectangle** widget—to be used as the login box background
 - Two **Text Field** widgets—to be used as the username input and the password input fields
 - One **Button** widget (any button style will do)
 - Two **Text - Paragraph** widgets
2. Create a username field, a password field, and a submit button. Put the Rectangle widget in the background (find out how).
3. Label all the widgets.
4. On the **Home** page's **Interactions** pane (yes, the one below the wireframe), find the **onPageLoad** event, double-click on **Case 1**, and add a second action under the same case (the first action was the selected state action). The new action is **Set Focus on Widget**. Assign this action to the username text field widget. This will put the cursor inside the username input field upon page load for improved usability. Generate and test it.



5. Add a new page to the site map and name it `user account`. That's where we will take the visitor from the home page after a successful login. Don't forget to add **myHeader** and **myFooter** to it.

Now we get back to editing the home page. Select the **Submit** button and assign a few actions to its `onClick` event in the following order:

1. In **Case Editor**, find the **Disable Widget(s)** action. This will make the button's color grey after clicking on it, to achieve an immediate visual action feedback.
2. Find and add (in the same window) the **Wait Time(ms)** action. Set it to 1500. This will simulate a delay.
3. Find and add the **Open Link In Current Window** action. Assign it to the **User Account** page. This will take the visitor to his personal account (simulating a successful login).



A single event may have multiple cases (named Case 1, Case 2, and so on). If you have such a configuration in your prototype, then you probably made a mistake. We are not yet dealing with multiple-case logic. Simply delete the extra cases and make sure that all your interactions are created under Case 1.

Things to do before proceeding to the next step

Only two quick tasks and we are done with this heavy weight step:

1. Change the order of the button's three actions under Case 1. Generate and see how it reacts.
2. Find a way to mask the input text of the password field while typing. It should show only dots while typing the password (hint: the right-click menu).

Step 7 – Adding some logic to the interactions

So far, clicking on the **Submit** button always took us to the **User Account** page. This is nice but not detailed enough for a high fidelity prototype.

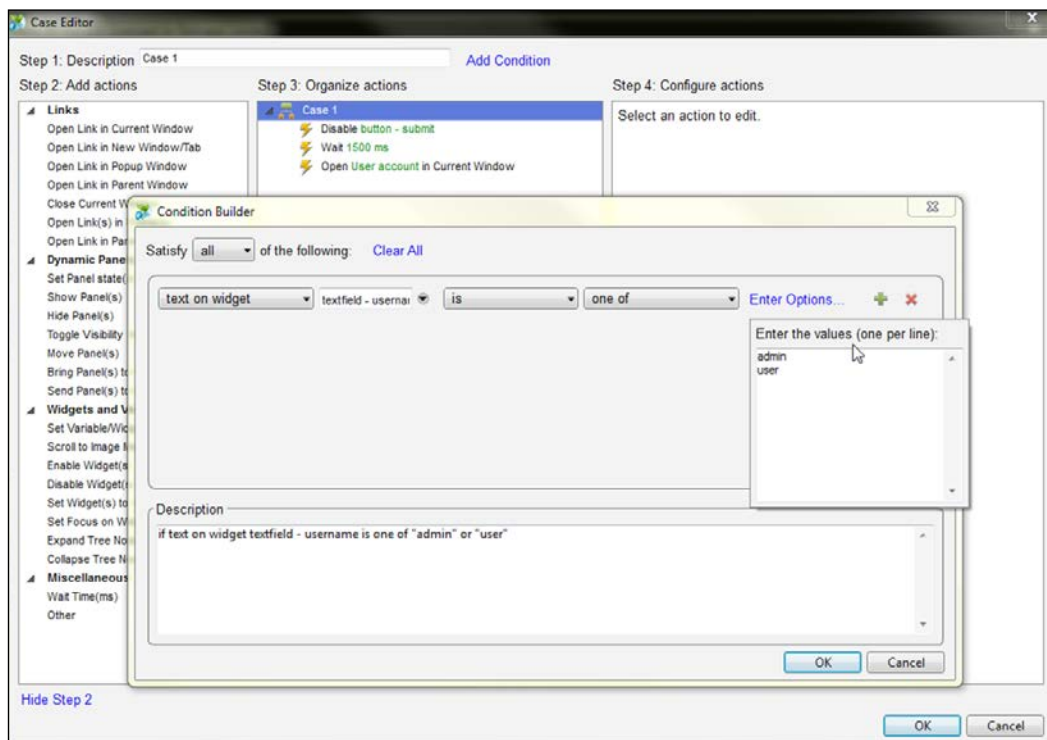
Our goal is to simulate success flow, failure flow, and sometimes even edge case flows. Adding conditional statements to interactions helps us achieve that higher goal.

Let's see how we can apply logic to the **Submit** button where successful logins will be granted only to specific usernames.

Select the **Submit** button on the home page. Double-click on **Case 1** to open **Case Editor** (this is the same case we created previously with the three actions).

At the top of the **Case Editor** window, find the link named **Add Condition**. This will add a logical condition to the three actions we defined earlier (a successful login scenario).

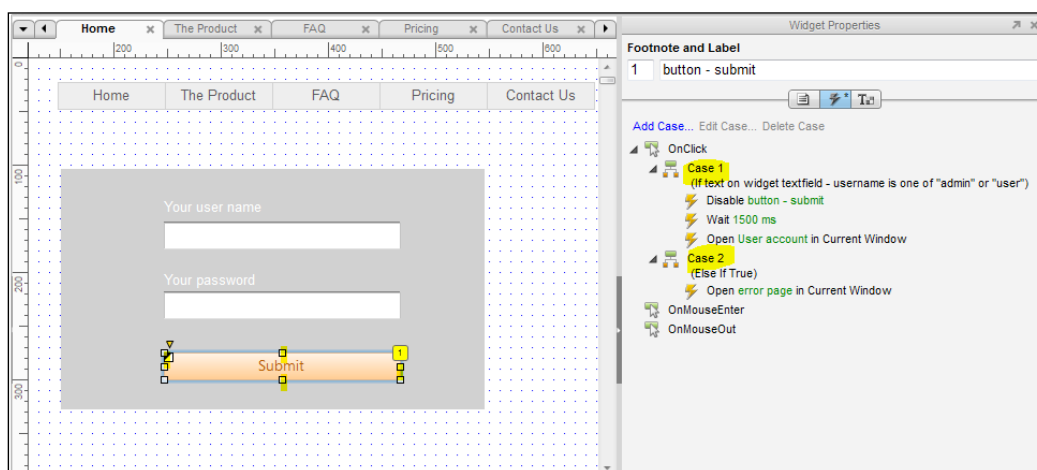
The **Condition Builder** window opens. There are numerous options there so don't get confused, just understand the idea. We will configure a logic saying that a successful login will happen only if the text in the username field widget is one of either `admin` or `user`. See the screenshot that follows and configure the same. Generate and test it.



Things to do before proceeding to the next step

A conditional statement is not complete without taking care of the "else" condition. What is to be done if the condition (in our scenario, if the username is neither admin nor user) is not fulfilled? Let's fix that.

1. Create a new site map page – a page that a visitor will be redirected to upon login failure. Name it `error page` and add **myHeader** and **myFooter** to it. Put some relevant error message text there.
2. Select the **Submit** button on the home page, highlight its **OnClick** event, and click on **Add case**. A new **Case Editor** window will open (**Case 2**) for the same OnClick event; notice that it automatically considered **Else (if true)**.
3. The interaction(s) you are about to add to this case (**Case 2**) will be activated only if the preceding logic condition (at **Case 1**) is not fulfilled (or in other words, if the visitor did not enter `admin` or `user` in the username field). In this case, we would like to send the visitor to the new error page. Configure it to open that page, generate, and test it.



Right-click on **Case 2** (under the button's **OnClick** event) and examine the **Toggle IF/ELSE IF** selection. Have you understood this? Run some tests to make sure you do.

Step 8 – Understanding dynamic panels

Let's face it, our login box interactions are so gos. A login failure message opens on a new page, instead of using in-page messages and tip boxes.

Dynamic panels are exactly the thing that leverage the prototype with elaborated client-side capabilities. You will find that dynamic panels are among the most used tools in your prototyping toolbox.

Imagine that you have a special area, floating on top of your wireframe design. An area comprised of a set of widgets, conditions, and actions. You can easily hide that area, show it, drag it, and swipe it. That area is a dynamic panel.

We will add two capabilities to our prototype, which reflect the most common usage of dynamic panels.

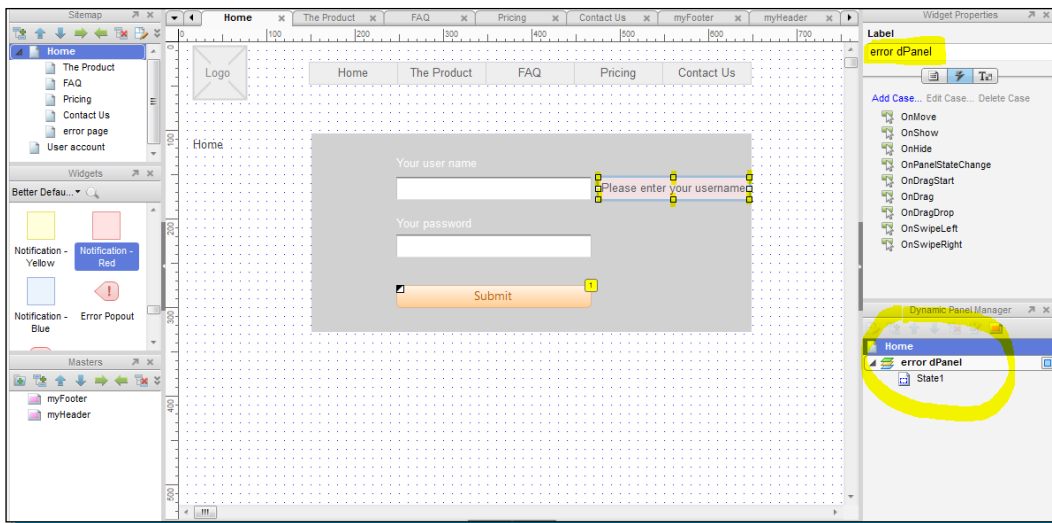
Adding a form field error message

Contextual messages usually appear without a click or a rollover and are triggered when an action is performed by the user. We will now create a message that appears in case the username field was mistakenly left blank.

Find and drag a widget called **Notification-Red** (which is actually a red rectangle but it is very useful) and place it close to the username input text field widget. Type `Please enter your password` on the red widget.

Right-click on the red widget and select from the menu **Convert To Dynamic Panel**. This will do the trick. The error message widget is now a dynamic panel.

Label the dynamic panel with a descriptive name and examine all its available events. In this example we will only utilize the `OnShow` and `OnHide` events giving us the ability to control when the message will appear and when it will be hidden. See the error message created in the screenshot that follows:

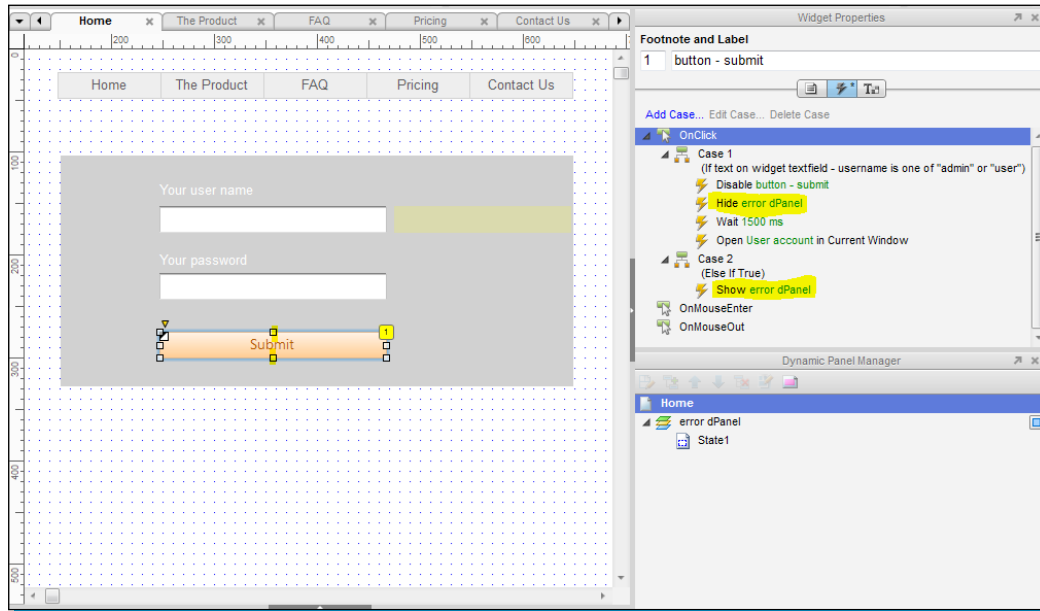


Now let's go back to the **Submit** button. Click on the **Button** widget and examine its interaction cases. Recall that Case 1 handles a successful login and Case 2 takes the user to the login failure page.

Delete the open page action and use a show panel action instead. To complete the logic, add a hide panel action in Case 1 (so a successful login will also hide the error message).

Set our dynamic panel to be hidden by default (we want to show it only after an erroneous login). To do that, right-click on the dynamic panel and select **Edit Dynamic panel – Set Hidden**. A yellow mark now designates the dynamic panel's location and we no longer see its content.

Use the following screenshot as a reference to generate and test the prototype:



Creating a tooltip box

Another cool capability of dynamic panels is that it allows us to create a tooltip box:

1. Drag a **Notification - Green** widget into the wireframe workspace, and type I am here to help! inside it.
2. Convert it into a dynamic panel and label it with a descriptive name. Hide this dynamic panel by default.
3. Search the Web, find a nice question mark icon, and paste it into the wireframe workspace. Once you paste it in the workspace, it becomes a widget, and you can assign interactions to it.
4. Configure the OnMouseEnter event of the question mark widget to show the dynamic panel. Configure the OnMouseOut event as well, to hide the dynamic panel.

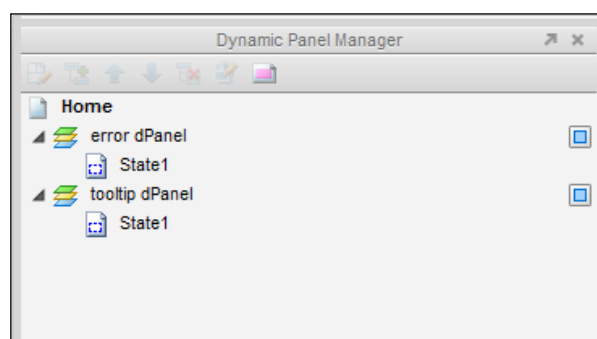


Importing an image into your Axure wireframe workspace is a very common practice. My favorite source for icons and symbols is www.iconfinder.com.

Understanding the Dynamic Panel Manager pane

This pane can be a bit confusing when taking your first steps in Axure, so let's understand the basics. It provides us with an overview of all the dynamic panels added (notice the two panels that you have already created). It is particularly helpful when you're using many dynamic panels on a page and want to navigate to them quickly. It also provides the ability to hide dynamic panels from the Wireframe workspace view, making it easier to select widgets that may be buried under them.

Each dynamic panel has at least one state. At any given time, there is only one active state per panel. If the dynamic panel is visible, what you see is the currently active state of that dynamic panel. In the next step, we will understand why these states are so powerful.



Things to do before proceeding to the next step

We created two dynamic panels and implemented a show and a hide interaction. Let's explore these panels more in detail.

- ◆ Refer to the **Dynamic Panel Manager** pane and double-click on **State 1** (the default state) of each panel. Edit the state's content and generate the prototype. Keep on doing this until you feel comfortable enough with the panel and state concepts.
- ◆ While editing a panel's state you'll notice a blue dashed outline. This shows the size of the dynamic panel and also acts as the boundary for the state. You can change the boundary's size by going back to the panel at the wireframe pane (it appears now as a yellow rectangle) and resize it.

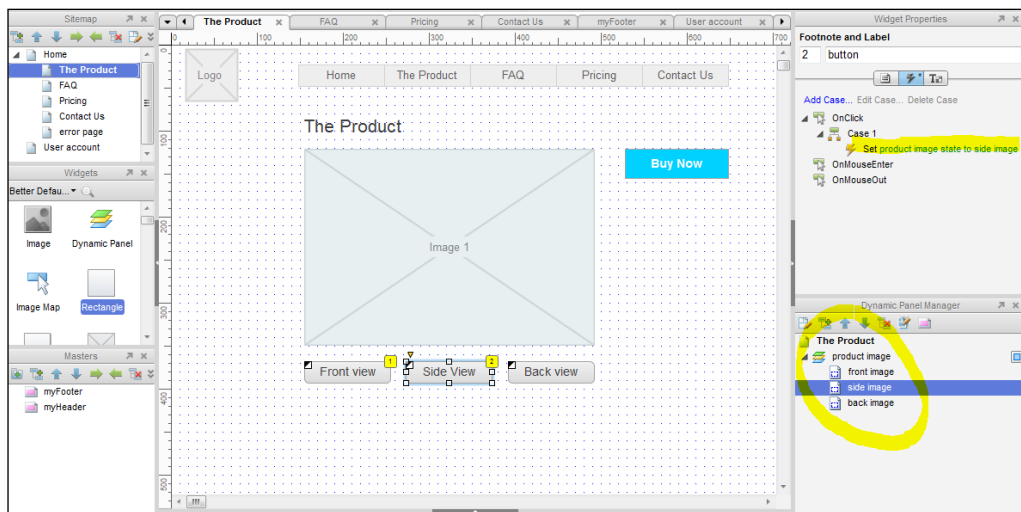
Step 9 – Adding states to dynamic panels

So far we had a single, default state per dynamic panel. In the following example, we will create a multi-state panel and demonstrate the high potential of such a capability.

Take a look at the screenshot that follows. Try to create a similar design in your prototype.

The following should be your task guidelines:

- ✦ Clicking on any of the three buttons changes the main image above them (there are three different images)
- ✦ The product image is a single dynamic panel with three states (each state includes a different image)
- ✦ In the **Case Editor** window, you will need to use the action **Set Panel state(s) to state(s)** in order to switch between the dynamic panel states



Things to do before proceeding to the next step

We created a dynamic panel with multiple states. Let's make sure we know how to get the best out of them.

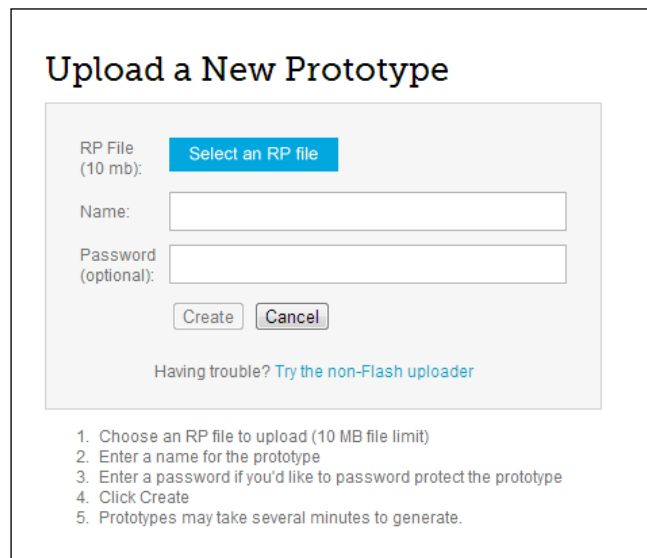
- ✦ Notice that while configuring the action **Set Panel state(s) to state(s)** in **Case Editor**, there were a few animation options (animate in and out). Explore these options.
- ✦ The dynamic panel's state can include many widgets. You can even add widget conditions inside specific states. Try this by adding a hyperlink (**Text - Link** widget) into one of the states (simply put it on top of the image already there), and make it a hyperlink that offers access to the pricing page.

Step 10 – Publishing the prototype

The Axure tool automatically generates HTML files based on your prototype design. A typical web design can easily involve a few thousand HTML and JavaScript files in multiple folders and around 20-30 MB of disk space. These files need to somehow be shared with your client, colleagues, graphic designers, web developers, or simply be presented in your web portfolio.

The most effective way to share prototypes without dealing with endless files and uploads is by using the AxShare cloud-based service. (<https://share.axure.com/>)

AxShare is a free hosting service for Axure prototypes. Simply open an account and upload your latest version of the *.rp file (only the source file is needed). The AxShare service automatically generates the HTML files, stores them, and then creates a unique URL to the prototype for you to share with your clients and colleagues.



The screenshot shows a web form titled "Upload a New Prototype". It contains a file selection area labeled "RP File (10 mb):" with a blue button "Select an RP file". Below this are input fields for "Name:" and "Password (optional):". At the bottom of the form are "Create" and "Cancel" buttons. A link "Having trouble? Try the non-Flash uploader" is located below the form. A numbered list of instructions is at the bottom of the page:

1. Choose an RP file to upload (10 MB file limit)
2. Enter a name for the prototype
3. Enter a password if you'd like to password protect the prototype
4. Click Create
5. Prototypes may take several minutes to generate.

An alternative way to share the prototype is to upload it directly into AxShare from the tool itself. (The feature is located under the **Generate** menu)

You have a few options to share a prototype:

- ◆ If you have never shared the prototype before, select **Create a new prototype** and Axure will upload, generate, and create a unique URL for you to use.

- ◆ If an older version of the prototype is already shared on AxShare, select **Replace an existing prototype** and enter a prototype ID, which is the six character long code that appears in your already shared URL. A new URL will not be created.

Publish to AxShare

Create an account

Enter your AxShare email and password. If you do not have an account, create one free at share.axure.com.

Email: amit.dalot@gmail.com

Password:

☒ Save password

Prototype: Html Prototype 1 (default)

Choose to create a new prototype or replace an existing one.

☒ Create a new prototype

Name: starter

Password: (optional)

☐ Replace an existing prototype

Prototype ID:

Publish Cancel

We are done!!

By now, you should have a live (and maybe even shared) prototype with a few pages, some widgets, some dynamic panels, and some logic. These are the building blocks for our next section, where we will sharpen our skills and make some common user interface design patterns.

Three common design patterns and how to prototype them

When you tackle a design challenge and decide to re-use a solution that was used in the past, you probably have a design pattern.

Design patterns are commonly used concepts that plead to be re-used. They save time, they are well recognized by a majority of users, they provide their intended functionality, and they always work (well, assuming you selected the right design pattern and used it in the right place).

In this section, we will prototype three commonly used design patterns:

- ◆ Module tabs pattern
- ◆ Carousel pattern
- ◆ Inline field adder with a spotlight effect pattern

Once you created a design pattern prototype, you can copy and paste it into an Axure project. A better alternative is to create a custom widgets library and load it into your Axure environment, making it handy for every new prototype you make.

Module tabs

Module tabs pattern allows the user to browse through a series of tabs without refreshing the page.

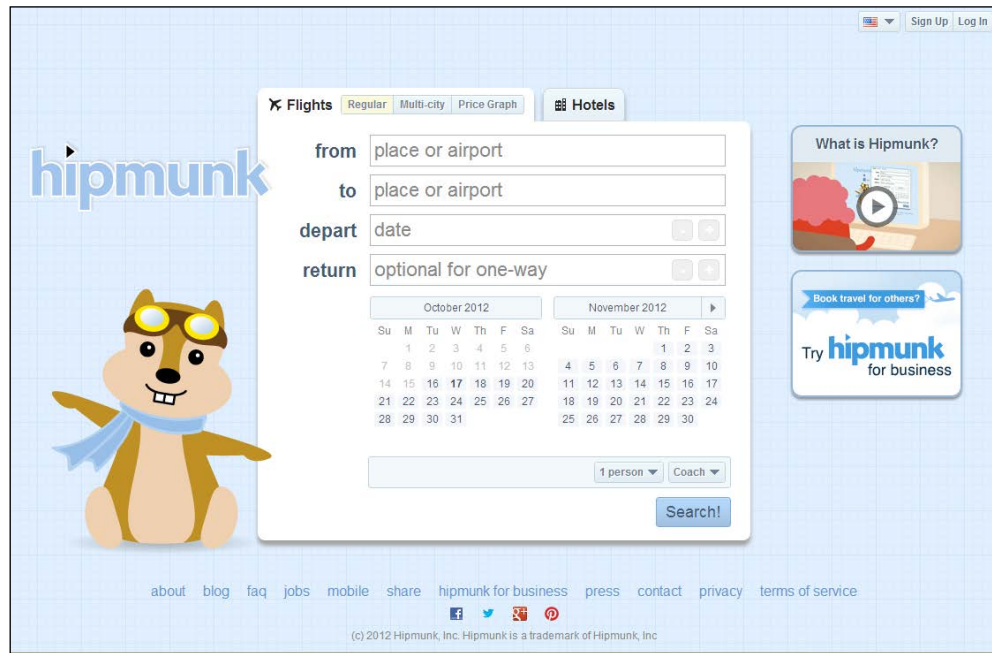
Why use it?

Module tabs design pattern is commonly used in the following situations:

- ◆ When we have a group of closely related content, and we do not want to expose it to the user all at once. (In order to decrease cognitive load, because we have limited space, or maybe in order to avoid a second navigation bar.)
- ◆ When we do not care for (or maybe even prefer) a use case where a visitor interacts only with one pane at a time, thereby limiting his capability to easily compare information or process data simultaneously.

Hipmunk.com is using it

A module tabs design is used in www.hipmunk.com for a quick toggle between **Flights** and **Hotels**. By default, the user is exposed to Hipmunk's prime use case, which is the flight search.



Prototyping it

This is a classic use of the Axure dynamic panels. We will use a mixture of button-shaped widgets along with a dynamic panel.

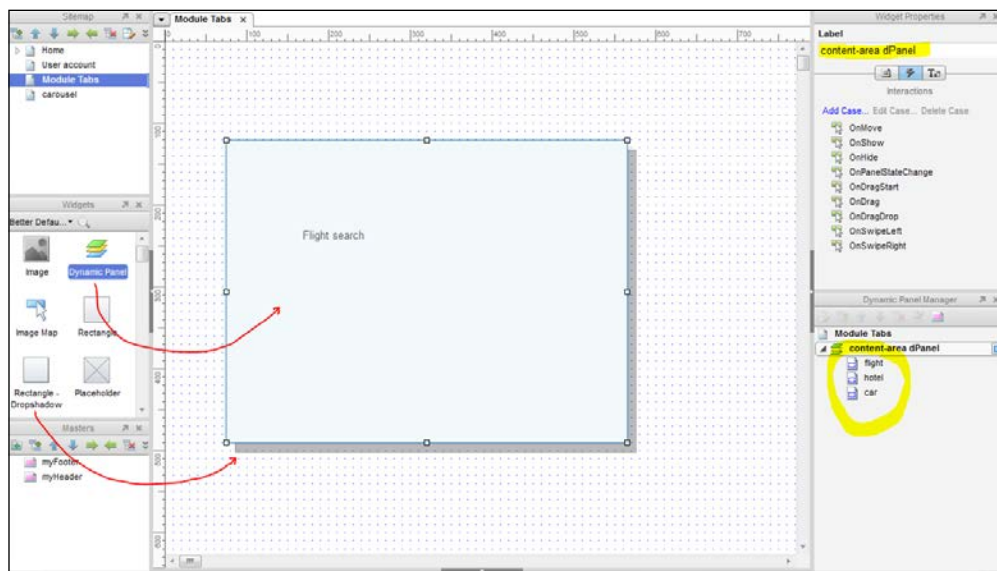
A module tabs design is comprised of a set of clickable tabs, located horizontally on top of a dynamic content area. Tabs can also be located vertically on the left side of the content area.

We will start with creating a content area having three states:

1. Drag a **Rectangle-Dropshadow** widget from the **Better Defaults** library into the wireframe and resize it to fit a large enough content area. This will be our background image for all content states. Now drag a **Dynamic Panel** widget, locate it on top of the rectangle, and resize it to fit the blank area. Label the dynamic panel with a descriptive name (the **Label** field is in the **Widget Properties** pane). In this example we will use the label: **content-area dPanel**.
2. Examine your dynamic panel on the **Dynamic Panel Manager** pane, and add two more states under it. Since we are in the travel business, you can name them as **flight**, **hotel**, and **car** (these states are going to include each tab's associated content).

- Put some text in each of the three dynamic panel states to easily designate them upon state change (later on you can replace this text with some real content).

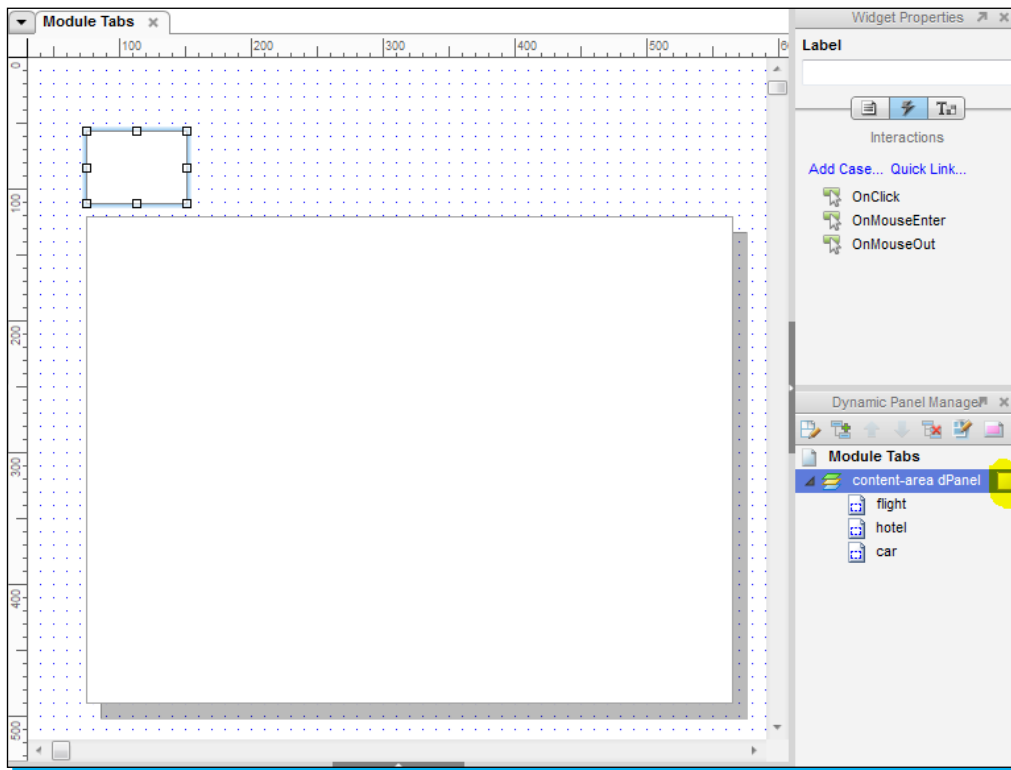
In the screenshot that follows, you can see the **content-area dPanel** with its three states.



Next, we will add three tab buttons to control these three states:

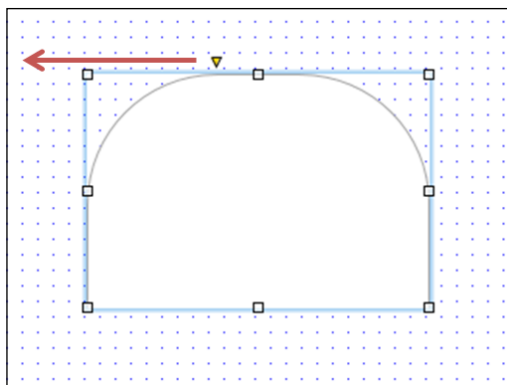
- We are going to experiment a little bit with styles and aesthetics, so let's hide the dynamic panel object from view by clicking on the small, blue box next to its name on the **Dynamic Panel Manager** pane. Hiding a dynamic panel has no functional implication; it will still be generated in the prototype.
- Drag one basic **Rectangle** widget, and place it above our large dropshadow rectangle. Resize the widget so that it will have the right proportions.

In the following screenshot, you can see the new rectangle widget location. Also you can see what happens when we hide the dynamic panel from view:



When a tab is selected (becomes active), it is important to make it look like it is part of the active area. We can achieve this by giving it the same color of the active state background as well as eliminating any border line separating them. But let's start with reshaping the rectangle widget, making it look like a tab:

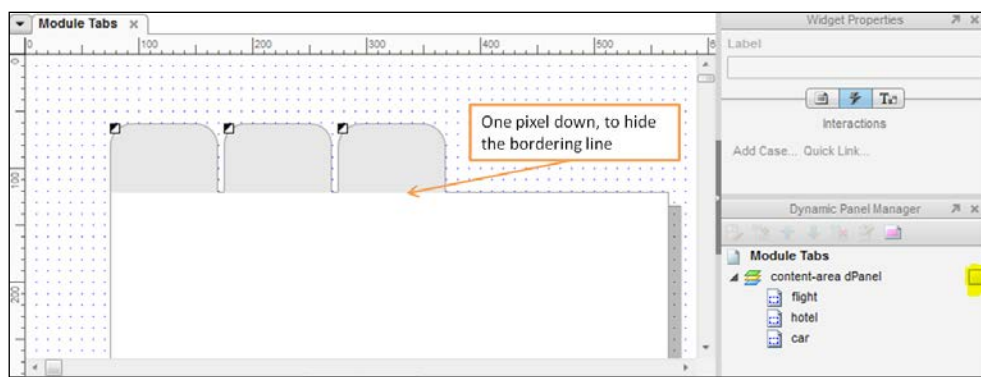
1. Right-click on the **Rectangle** widget and edit its shape (click on **Edit button shape** and select a shape). Choose the rounded top shape. This shape has no bottom border line and that's why we like it.
2. Notice the small yellow triangle on top of the rounded top. Try to drag it to the left. You will see that we can easily control its rounded corners' angle, making it look much more like a tab button (See the screenshot that follows).



Now that we have a nice looking tab button we can move forward.

1. By default, the tab should have a gray color, which designates an unselected state. So, select the widget and click on **Fill color button** on the Axure toolbar to set its default color to gray.
2. Once the tab is selected, it needs to be painted white, so that it easily blends with the content area background color under it. To do that, right-click on the tab and edit its selected style (**Edit button shape | Edit selected style**); the fill color should be set to white (select the **preview** checkbox to verify that).
3. We are almost done. Duplicate the tab and place three identical widgets above the content area. Move them one pixel down so that they overlap and hide the rectangle border line underneath them.

In the screenshot that follows, you can see how the three tabs are placed on top of the white rectangle in such a way that they hide the rectangle's border line.



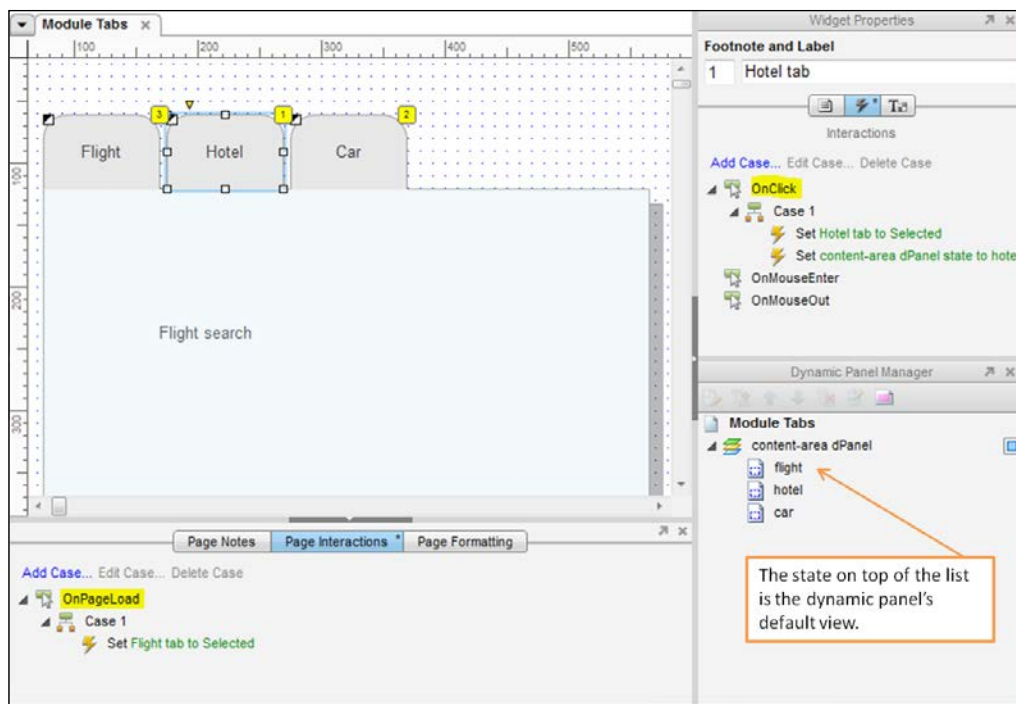



Examine the split square in the upper-left corner of each tab widget you created. You can preview the rollover style and selected state style of the widget by hovering over and clicking on it. Try it!

The last thing to do is to add interactions. We would like to set a tab to the selected state upon mouse click, and to change the state of **content-area dPanel** accordingly.

1. Put a name on each tab (**Flight**, **Hotel**, and **Car**), and give each tab a descriptive label. Set the **OnClick** event for each tab to perform two actions:
 - **Set Widget(s) to selected state**: Setting the widget after it was clicked to be in a selected style.
 - **Set Panel state(s) to state(s)**: Setting the dynamic panel state to the relevant content (**Flight**, **Hotel**, or **Car**).
2. Since we would like to have only one selected widget at a time, we should put the three of them in a selection group. Do that by selecting all the three widgets, right-clicking, and choosing the menu option **Assign Selection Group**.
3. Each time this page is loaded, the **Flight** tab needs to be selected. To do that, we will configure the **OnPageLoad** event to set the **Flight** tab to be selected (find it at the pane located under the wireframe). By the way, there is no need to set the dynamic panel state to **flight** as well since it is already located on top of the three states in the **Dynamic Panel Manager** pane, making it the default state.

In the screenshot that follows, you can see the **Page Interactions** pane located under the wireframe area. You can also see that the **flight** state is the one on top of the list, which makes it the dynamic panel's default view.




 The menu navigation widget described in the *Creating your first prototype* section, cannot be used as an alternative for the tab buttons design explained here. The reason is that the navigation menu does not support the Selection Group option.

Carousel

Carousel is a set of items that the user can browse through (using buttons, arrows, numbers, or any other visual clue).

Why use it?

A carousel design pattern is commonly used when you want to showcase a set of "featured" visuals, which have equal importance to your user but you do not have enough space to show all items at once.

Apple.com is using it

A carousel pattern is commonly used on the Apple website that showcases a featured product with a large, prominent image, enabling visitors to browse and see some more visuals.



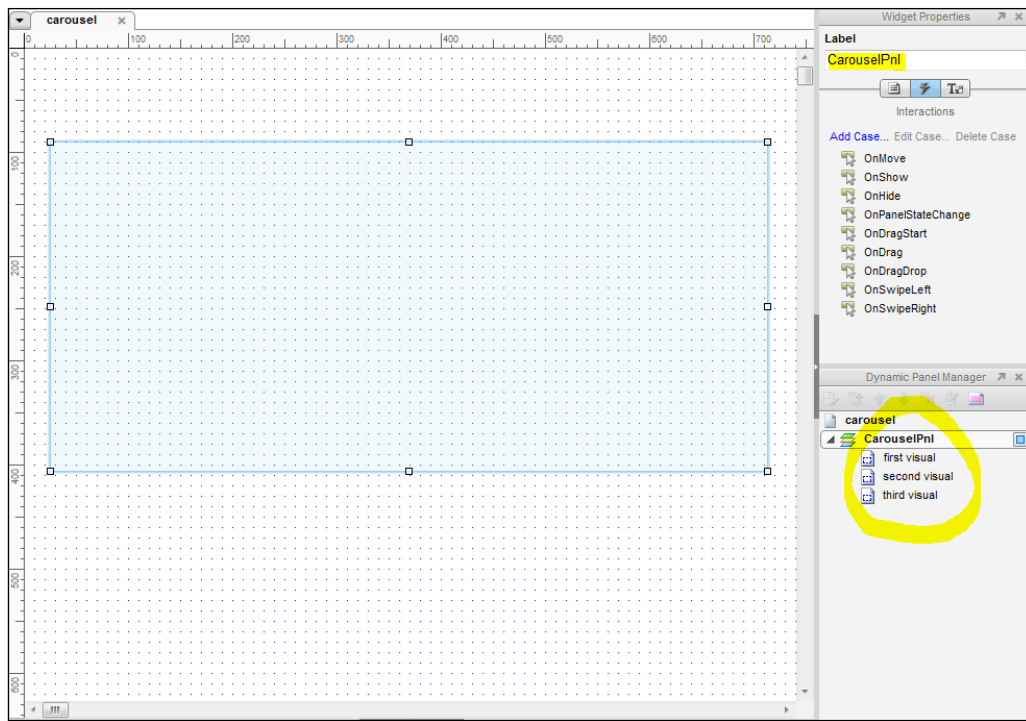
Prototyping it

The carousel pattern prototyping technique has some similarity to prototyping the module tabs pattern.

We will use control buttons that change the states of a dynamic panel:

1. Drag a **Dynamic Panel** widget onto the wireframe pane. Resize it to make it large enough to include a prominent image, some text, and a call for action button.
2. Label the dynamic panel as **CarouselPnl**.
3. Add two additional states to **CarouselPnl** (at the **Dynamic Panel Manager** pane).

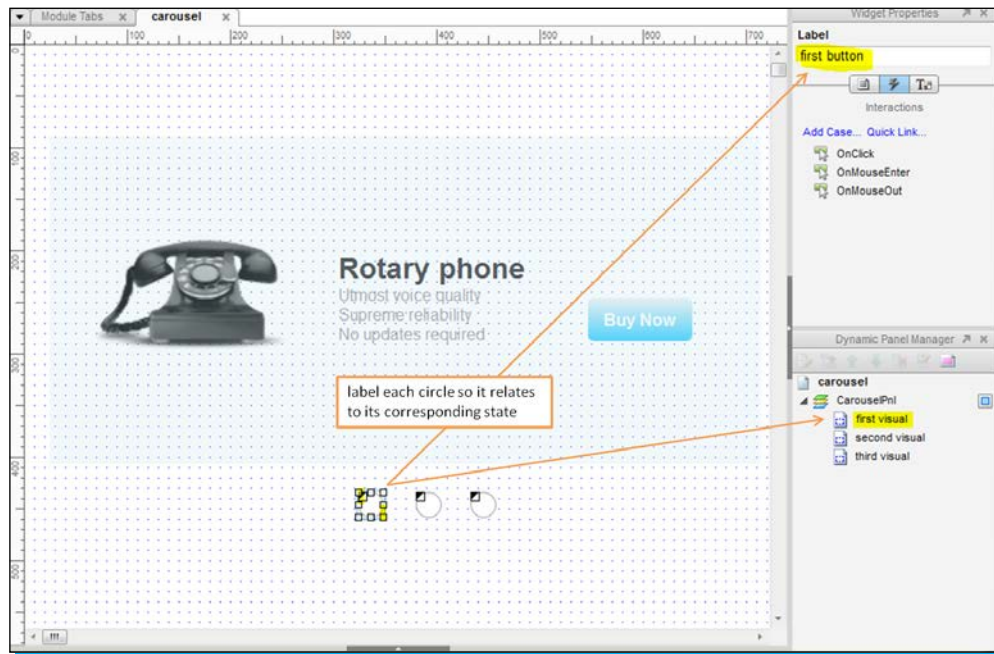
In the screenshot that follows, we added three states under **CarouselPnl** named: **first visual**, **second visual**, and **third visual**:



Next, we will put some content in each one of the carousel states, and allocate three control buttons:

1. Do the following for each one of the three dynamic panel states:
 - Add some content into the state by dragging some widgets (text widgets, images, buttons, and so on)
 - Make sure that the content resides within the light blue lines (the dynamic panel size limits)
2. Go back to the main page (double-click the site map page) and drag a **Rectangle** widget under the carousel panel. We are going to transform it into a circle. Convert the rectangle widget shape into an ellipse (use the right-click menu, and select **Edit button shape | Select shape**). Now experiment with its dimensions so that it will have the shape of a small circle.

3. Make three identical circles, and place each one next to the other. Give each one a descriptive label (remember that clicking a button triggers a change in the dynamic panel's states, so label each circle with a name corresponding to its relevant state).



At this stage we have three carousel visuals and three control buttons. All we need now is to assign some interactions, and we are done.

1. Select the three control buttons together and right-click on them. Under the **Edit Button Shape** option you will find the **Assign Selection Group** feature. This will assure you that only one button is selected at a time (recall that we have done this before with the module tabs pattern as well as with the radio button widgets).
2. Edit a selected style for the three buttons. Choose a bold fill color to designate a selected style state. (In the "iPhone" screenshot they used a light blue selected style.)
3. Set two OnClick event actions for the three buttons as follows:
 - Upon click, the button style should be changed to selected style
 - Upon click, the relevant dynamic panel state should be changed consequently to show the required visual

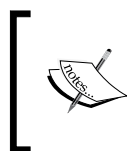
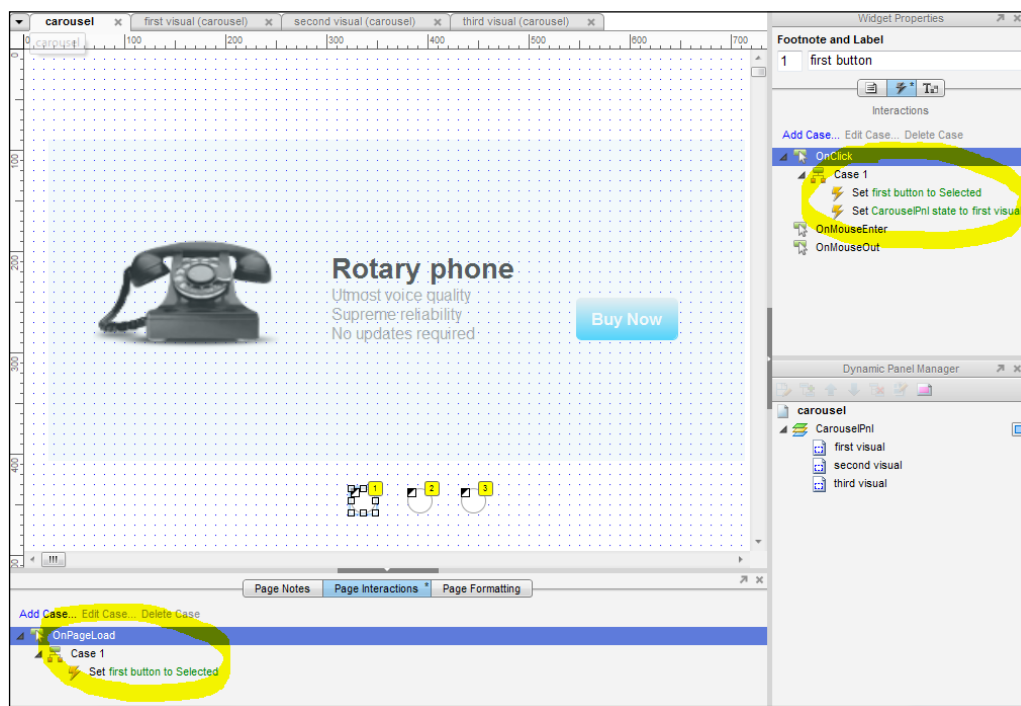


Try to copy and paste the events and actions. It works and it is very handy. Copy the entire event from one circle and paste it into another. Now you only need to modify the action's details.

And last, let's not forget to configure the page default mode.

On page load, we would like to highlight the default, leftmost button making it obvious to the visitor that he sees only one visual out of a set of items. To do that, open the **Page Interactions** pane (the pane under the wireframe area), and configure its **OnPageLoad** event to set the leftmost button to selected state.

We are done!



The panel's state change action has some useful animation options. Try changing one state to another while animating a slide effect. The user may actually expect to see that kind of interaction in a carousel. You will need to add some simple logic conditions to control the sliding direction.

Inline field adder and a spotlight effect

An inline field adder allows the user to add more fields to the page only when they are needed.

A spotlight effect will make this prototype shine! We will come to this later.

Why use it?

Inline adders are commonly used when:

- ◆ When you want to show minimum number of fields by default (because you want to save space, to avoid congestion, to improve conversion rate, and such)
- ◆ When the number of potential fields to be added differ from one user to another

Kayak.com is using it

Inline adder pattern is used on www.kayak.com as well as on many other websites dealing with flight itinerary forms. The number of flights the visitor may need cannot be predicted, but a reasonable maximum number of fields can be defined (in this case the number of fields is limited to 6), making it a perfect match for an inline adder pattern.

The screenshot displays the KAYAK website's flight search interface. At the top, the KAYAK logo is on the left, and navigation links for Flights, Hotels, Cars, Deals, and More are in the center. On the right, there are links for Login, My Trips, Free App, and a user profile icon. Below the navigation, there are three tabs: Round-trip, One-way, and Multi-city, with a 'clear all search criteria' link. The main search area contains three identical flight entry forms stacked vertically. Each form has 'From', 'To', and 'Depart' fields. The 'From' field is highlighted with an orange border. Below each 'From' field is an 'Include nearby' checkbox. The 'Depart' field has a date picker icon and a dropdown menu set to 'Anytime'. To the right of the third form is a close button (X). Below the forms is a link 'Add another flight (up to 6)'. At the bottom of the search area, there are dropdowns for '1 adult' and 'Children or Seniors', a dropdown for 'Economy', and a checkbox for 'nonstops only'. A large orange 'Find Flights' button is at the bottom. On the right side of the page, there is a section titled 'SEARCH ONE AND DONE.' with a subtext 'Compare hundreds of travel sites at once. Choose where to book.' Below this is a section for 'Free KAYAK Mobile Apps' featuring an app icon and text: 'The #1 Travel App 19,000,000+ downloads Try KAYAK Mobile.' At the very bottom, there is a footer with links for About, Privacy, Destinations, Feedback, and social media icons, followed by a small disclaimer: 'Cheap flights, hotels and travel deals. Compare hundreds of travel sites at once to find cheap flights, the best flight deals and cheap airfare. Use KAYAK to book the cheapest.'

Prototyping it

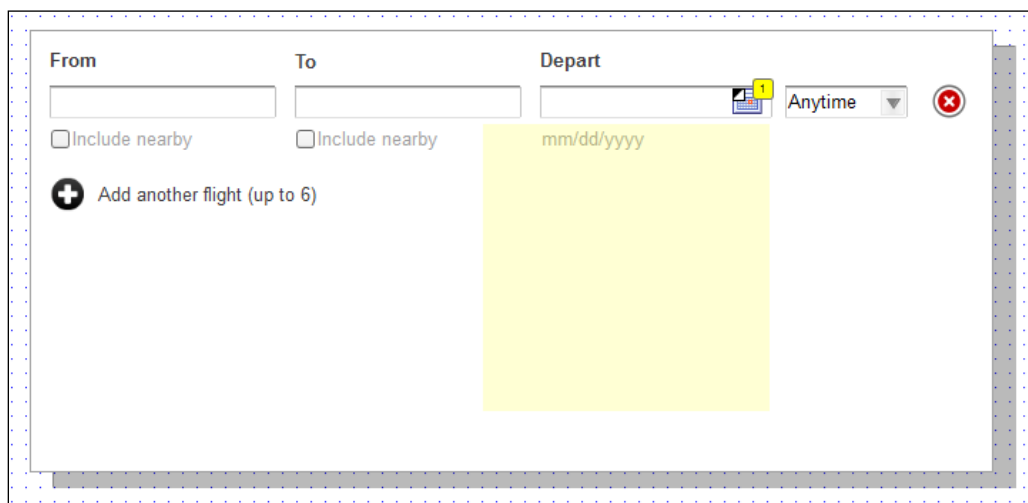
The inline adder mechanism has two core elements: the field addition mechanism and the field deletion mechanism. When prototyping such a pattern, it is enough to simulate only these two scenarios accompanied with some kind of annotation, describing the maximum additions allowed.

In other words, if your project allows the user to create up to seven field additions, it is your responsibility as a UI designer to make sure seven fields fit the page, but it is more than enough to prototype only a single interaction.

The following steps describe the process for prototyping the main interactions of an inline field adder:

1. Drag a **Rectangle - Dropshadow** widget into the wireframe area. This will act as a background image.
2. Let's make the same field used in the "Kayak.com" screenshot. Place three **Input Field** widgets on top of the rectangle. Drag the **Date Picker** widget (it is a nice opportunity to get familiar with it), a droplist, and two checkboxes.
3. Search the web for two nice looking images: the plus symbol and the delete symbol. These will be our control buttons.

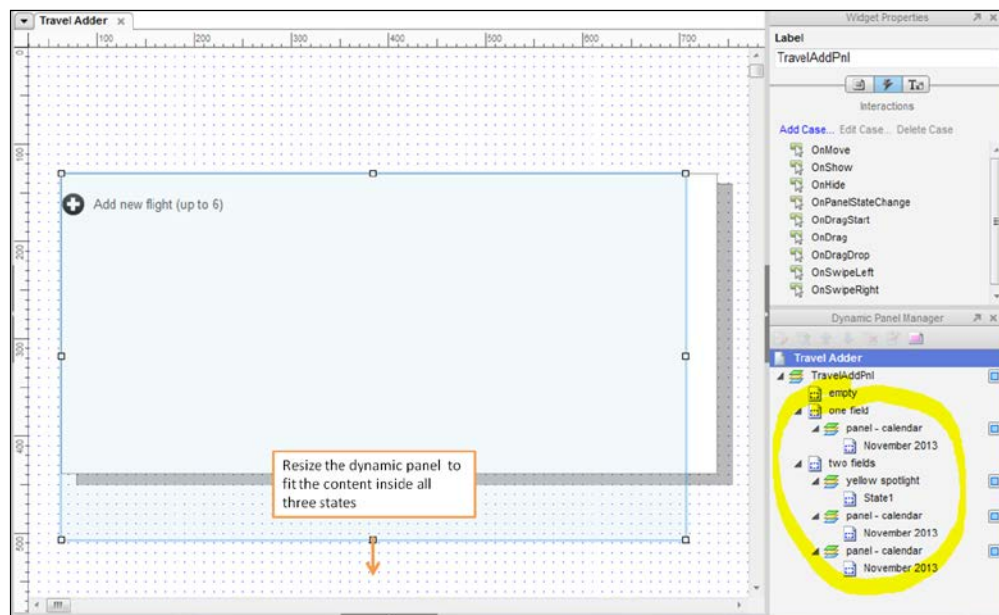
In the screenshot that follows, you can see a design close enough to the input field structure on Kayak.com:



By now we have a single template of an input field. Clicking on the plus symbol should add another similar field, and clicking on the delete symbol should delete an existing field. We will simulate this by setting up three dynamic panel states that will demonstrate each scenario:

1. Select the entire input field (don't forget to include the symbols and the date picker dynamic panel, but do not include the rectangle background), and convert this to a dynamic panel. Label the panel as `TravelAddPnl`.
2. Create two more states for **TravelAddPnl** and edit their content as follows:
 - A state named `empty`, which includes only the plus symbol with the text `add new flight (up to 6)`.
 - A state named `one field`, which is actually the state you already created—the one with the single field.
 - A state named `two fields`, which includes two input fields and the same control symbols (simulating what the user sees after adding a second field) . You may need to resize **TravelAddPnl** to fit all the components.
3. The first time, it may take you a while to duplicate the fields, copy, paste, align them, and resize the dynamic panel to fit all the components. But this practice is crucial. Shortly, you will make these actions fast and effectively.

The screenshot that follows shows the **empty** state, but it was taken right after creating the 3rd state (named: **two fields**). We had to go back and resize the dynamic panel to fit all the widgets required there.

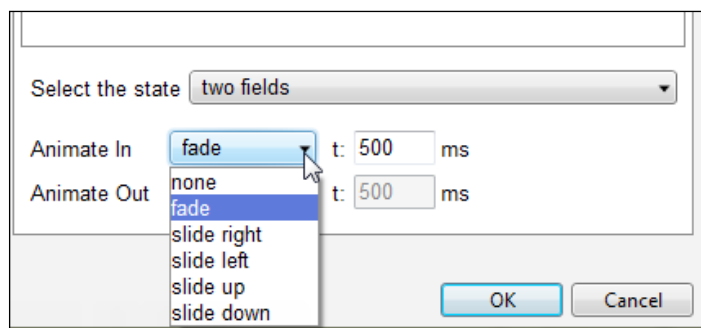


Finally, it is time for interactions!

Clicking the control symbols should set the dynamic panel to the relevant state:

1. Edit each state separately and assign an **OnClick** event to the plus symbol as well as to the delete symbol. You will only need to use the **Set Panel State(s) to State(s)** action.
2. Try the animation effects. Get used to the sliding animation as well as the fade animation.
3. Once you are done practicing animations, set them all back to **none**. The spotlight effect, which is coming soon, won't work with animations!

The screenshot that follows shows the various interaction animation options in the **Case Editor** window.



We already mentioned that prototyping all possible scenarios is a waste of time, but yet, it is most recommended to dedicate one additional state for the edge case. This will visualize what happens when the user reaches the maximum fields allowed.

Spotlight effect is a great way to focus the user's attention on an interface change just made. Our spotlight will use a yellow background color that appears behind the newly created field and fades out a second later.

1. Grab the **Notification - Yellow** widget (which is actually a yellow rectangle, and a very useful one) into the dynamic panel state named **two fields** (in this example we will implement a spotlight effect only there). Place it behind the second field, and resize it to fit the entire field background. (Explore the **Order** options in the right-click menu.)
2. Convert the yellow widget to a dynamic panel and label it as **YellowSpotPnl**. Set it to be hidden by default (**Edit Dynamic Panel | Set Hidden**).

3. Go to the **one field** state and add the following actions to the **OnClick** event of the plus symbol (right after the panel's state change action):
 - **ShowYellowSpotPnl**
 - **Wait 500 ms**
 - **HideYellowSpotPnl** (configure this action with a fade animation!)

The screenshot that follows shows what the visitor will see (for half a second), immediately after adding a new flight:

The screenshot shows a flight booking form with two rows of input fields. Each row has three columns: 'From', 'To', and 'Depart'. The 'From' and 'To' columns have text input fields. The 'Depart' column has a date input field with the value '11/14/2013' and a dropdown menu with the value 'Anytime'. Below each 'From' and 'To' input field is a checkbox labeled 'Include nearby'. To the right of each row is a red 'X' icon. At the bottom of the form is a plus icon and the text 'Add another flight (up to 6)'.

In this section, we practiced only three common design patterns: Module tabs, Carousel, and Inline field adder.

In the next section, you will find web references describing many others.

Bear in mind that some design patterns may look complicated at first sight, but with the skills you gained so far there is no longer a pattern that you cannot prototype. Keep on practicing, and you'll become an Axure professional much faster than you think!

Good luck!

People and places you should get to know

Every successful tool is centered around a community. This section provides you with some useful links to resources dealing with user interface prototyping and Axure.

Developing your Axure skills

The resources that follow are probably the best places to find answers to your questions, or to get tips on great new ways to use Axure:

- ◆ www.axure.com/training: The official website of Axure has plenty of training materials, tutorials, and videos. They are all organized by skill level.
- ◆ www.axure.com/download-widget-libraries: It is the largest reference for Axure widget libraries. Download and install them into your Axure workspace.
- ◆ www.axureland.com: It is the place to show off your design work for free, get some inspiration from others in the community, download the latest libraries, and explore some new Axure techniques.
- ◆ www.axureworld.org: It provides Axure practitioners an opportunity to share their expertise. The videos library has some most interesting sessions.

Finding inspiration

It is always nice to see how other designers successfully faced challenges, which may be similar to the ones in your project. The links that follow will help you with that:

- ◆ www.konigi.com: Refer to the *SHOWCASE* section, where interface design patterns are sorted in a tag cloud.
- ◆ www.pttrns.com: Browse through beautiful iPhone application designs.
- ◆ www.smileycat.com/design_elements: Browse among 37 categories
- ◆ www.mobile-patterns.com: It is a mobile design patterns gallery. It contains only a few categories, but is focused on the world's most successful applications.
- ◆ www.littlebigdetails.com: It showcases a single, fun, UX feature every day.

Participating in forums and subscribing to blogs

The following links will take you to places where lively and insightful discussions happen and to places where opinionated leaders express their thoughts:

- ◆ forum.axure.com/forum.php: With thousands of posts and a very active community, the Axure forum is the right place to sharpen your skills
- ◆ uxdesign.smashingmagazine.com: It contains practical articles and case studies written by professionals and experts in the field of user experience design.
- ◆ stackoverflow.com: It is probably the most popular, but yet professional place to find discussions about web usability topics, challenges, and solutions.
- ◆ www.lukew.com: Luke Wroblewski's blog includes resources for mobile and web product design and strategy, presentations, and articles.

Using some valuable services

Services that can help you be more efficient while prototyping:

- ◆ www.iconfinder.com: It is the ultimate collection of free web symbols and icons. Use these symbols to enrich your Axure prototype.
- ◆ www.axutopia.com: It is a huge collection of high quality, high fidelity widget libraries for Axure, categorized by platforms and devices. The libraries are not free, but nowhere else can you find widgets with such detail and accuracy.
- ◆ share.axure.com: AxShare, the free prototypes hosting service, as described earlier in the *Quick start – creating your first prototype* section.

Reading other Axure RP books

The following books provide complementary reading of other aspects in Axure prototyping:

- ◆ Ezra Schwartz's *Axure RP 6 Prototyping Essentials* (446 pages), provides a thorough approach to each of the topics described in this book (<http://amzn.to/Zf9okQ>).
- ◆ Lennart Hennig's *Axure for mobile*, focuses on Axure prototyping for mobile devices and tablets (<http://amzn.to/S6u64h>).



Thank you for buying
Instant Axure RP Starter

About Packt Publishing

Packt, pronounced 'packed', published its first book "*Mastering phpMyAdmin for Effective MySQL Management*" in April 2004 and subsequently continued to specialize in publishing highly focused books on specific technologies and solutions.

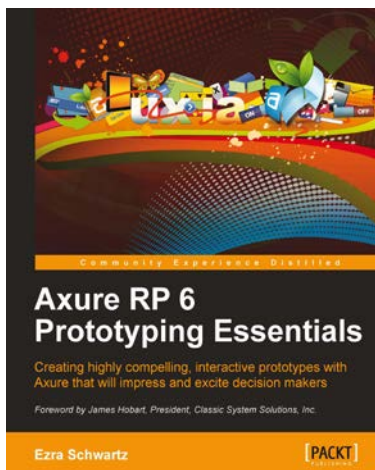
Our books and publications share the experiences of your fellow IT professionals in adapting and customizing today's systems, applications, and frameworks. Our solution based books give you the knowledge and power to customize the software and technologies you're using to get the job done. Packt books are more specific and less general than the IT books you have seen in the past. Our unique business model allows us to bring you more focused information, giving you more of what you need to know, and less of what you don't.

Packt is a modern, yet unique publishing company, which focuses on producing quality, cutting-edge books for communities of developers, administrators, and newbies alike. For more information, please visit our website: www.packtpub.com.

Writing for Packt

We welcome all inquiries from people who are interested in authoring. Book proposals should be sent to author@packtpub.com. If your book idea is still at an early stage and you would like to discuss it first before writing a formal book proposal, contact us; one of our commissioning editors will get in touch with you.

We're not just looking for published authors; if you have strong technical skills but no writing experience, our experienced editors can help you develop a writing career, or simply get some additional reward for your expertise.



Axure RP 6 Prototyping Essentials

ISBN: 978-1-84969-164-2 Paperback: 446 pages

Creating highly compelling, interactive prototypes with Axure that will impress and excite decision makers

1. Quickly simulate complex interactions for a wide range of applications without any programming knowledge
2. Acquire timesaving methods for constructing and annotating wireframes, interactive prototypes, and UX specifications
3. A hands-on guide that walks you through the iterative process of UX prototyping with Axure



Balsamiq Wireframes Quickstart Guide

ISBN: 978-1-84969-352-3 Paperback: 142 pages

Wireframe like a pro, the easy way

1. A simple yet professional approach to wireframing and prototyping using Balsamiq Mockups
2. Practice essential wireframing skills using real-world examples and challenging exercises
3. Build simple, interactive, clickable and effective prototypes with Balsamiq

Please check www.PacktPub.com for information on our titles



Responsive Web Design with HTML5 and CSS3

ISBN: 978-1-84969-318-9

Paperback: 324 pages

Learn responsive design using HTML5 and CSS3 to adapt websites to any browser or screen size

1. Everything needed to code websites in HTML5 and CSS3 that are responsive to every device or screen size
2. Learn the main new features of HTML5 and use CSS3's stunning new capabilities including animations, transitions and transformations
3. Real world examples show how to progressively enhance a responsive design while providing fall backs for older browsers



HTML5 Boilerplate Web Development

ISBN: 978-1-84951-850-5

Paperback: 174 pages

Master Web Development with a robust set of templates to get your web projects done quickly and effectively

1. Master HTML5 Boilerplate as starting templates for future projects
2. Learn how to optimize your workflow with HTML5 Boilerplate templates and set up servers optimized for performance
3. Learn to feature-detect and serve appropriate styles and scripts across browser types

Please check www.PacktPub.com for information on our titles

