Table of Contents

Table of Contents ........................................................................................................... - 2 -
Introduction ......................................................................................................................... - 5 -
Starting Work .......................................................................................................................... - 6 -
   Defining the Contents of a Screen .................................................................................. - 8 -
       Working with the screens ......................................................................................... - 9 -
   Importing external content ............................................................................................ - 10 -
   Interactive screen components ....................................................................................... - 11 -
   Links ................................................................................................................................. - 11 -
Browsing through your prototype ....................................................................................... - 12 -
   Interactive Mockups ....................................................................................................... - 12 -
   Sharing your prototype ................................................................................................. - 14 -
Dynamic Content .................................................................................................................. - 15 -
   Tabs ................................................................................................................................. - 15 -
   Dynamic Forms ............................................................................................................. - 16 -
   Conditions ...................................................................................................................... - 19 -
   Variables ....................................................................................................................... - 20 -
   Templates ....................................................................................................................... - 22 -
   Masters ........................................................................................................................... - 23 -
   Templates or Masters? ................................................................................................. - 24 -
   Widgets Libraries .......................................................................................................... - 24 -
Simulate Drag and Drop ..................................................................................................... - 24 -
   Simple Drag and Drop ................................................................................................. - 24 -
   Drag Events ................................................................................................................... - 25 -
   Actions ........................................................................................................................... - 25 -
Effects and Animations ................................................................. - 26 -
Fade .................................................................................................. - 26 -
Slide .................................................................................................. - 26 -
Swipe Gesture ................................................................................... - 26 -
Design Patterns ................................................................................ - 27 -
Changing the value of a text dynamically ..................................... - 27 -
Tooltip .............................................................................................. - 28 -
Videos from YouTube and Google Maps ...................................... - 28 -
Scroll Bars ....................................................................................... - 29 -
Conditional Drop-Downs ............................................................... - 29 -
Jump between input fields using the Tab key ............................... - 30 -
Simulating Data ................................................................................ - 32 -
Read Data ....................................................................................... - 33 -
Write Data ....................................................................................... - 33 -
Dynamic Data Lists ......................................................................... - 34 -
Create a Data Master ...................................................................... - 34 -
Simulate Searches .......................................................................... - 35 -
Paginating Data Grids ..................................................................... - 35 -
Forms ............................................................................................... - 37 -
Create .............................................................................................. - 37 -
Edit Form ......................................................................................... - 40 -
Delete Form ...................................................................................... - 43 -
Completing a form with the data from a line on a list .................. - 43 -
Shopping Cart .................................................................................. - 46 -
Scrolling and Push Content ............................................................ - 49 -
Scroll Bars ....................................................................................... - 49 -
Introduction

**Justinmind Prototyper** is a quick prototyping tool that allows you to create interactive and accurate simulations of the applications you need. With **Justinmind Prototyper**, you can easily incorporate any corporate image to your prototypes, export them in HTML format to demonstrate them on-line or automatically generate all of the documentation in an Open Office or MS Word file. And without requiring programming knowledge or a single line of code.

- **No Programming.** Justinmind Prototyper is an intuitive tool. All that’s required is to drag the components or interactions you need from the palettes to the work area.

- **Instantaneous Simulations.** With one click on the Simulate button, you will see the prototype of your application in action. No waiting.

- **Documentation automatically generated.** If you wish, your prototype comes with all the necessary documentation; Justinmind Prototyper generates this documentation for you; you only have to decide which format you want.

- **Exportable in HTML format.** Without any effort, so that your customers or users can see how it works on-line and give their opinion. Use it right from the beginning of your projects for total assurance, saving you rework costs and any inconvenience to your customers, and allowing you to create applications with easy usability.

Moreover, if you publish your prototypes in **Justinmind Usernote**, your customers can share their remarks on-line in a collaborative effort. Follow their comments as if they were conversation threads in a forum, and take advantage of their feedback; if you do it on a prototype, you won’t have to do it later on the application. Remember that **Justinmind Usernote** integrates the leading online testing tools on the market.
Starting Work

We’ve designed the interface of Justinmind Prototyper in such a way that you can create interactive prototypes as quickly and skillfully as possible. To simplify it further, we have incorporated drawing tools common to other applications.

1. **The Menu Bar.** The menus for the different tasks are organized according to topic. For example, in the “Generate” menu you will find the commands to generate HTML, documents or images.

2. **The Tool Bar.** Contains quick editing options for the component you have selected. It also has access to the more common options and a selection of the most commonly used editing tools.

3. **User Interface Tab.** This covers everything relating to screen editing and interaction in the prototype.

4. **Component Palette.** This feature gives you all the elements that you can add to a screen. To do this, simply drag them to the canvas or click on the item and then click again on the canvas position desired.
5. **Canvas or Work Area.** Displays the content of the active screen and allows you to add or change its components.

6. **Screen Tree.** Lists the screens that your prototype contains. Double-clicking on any part of the tree gives you access to its contents.

7. **Properties of the selected item.** Here you can edit the properties of the component’s graphic style that you selected (color border, text, font). It also includes a section where you can add comments, interactions and requirements relating to the item.

8. **Content of current screen.** In the form of a list, you will see the components included in the screen that is active on the canvas. From here you can also select them individually or change them.

9. **Progress Tabs.** They are extremely useful for jumping quickly between the most recently-opened screens. The selected tab indicates the active screen on the canvas.

10. **Simulation Button.** When you click on it, you can test the navigation of the prototype as though you were the end user.
Defining the Contents of a Screen

From the component palette, drag and drop the item you want to the canvas. Once entered in the canvas, you can re-size and move it to fit into your design. To move it, select it and drag it to the destination you want; or you can also push it with the arrow keys (holding down the shift key speeds it up). To help you debug your design, you also have the following instruments:

- The automatic alignment system (blue guides).
- The grid (see Mesh).
- The Rules.
- The Guides. To position them, click on the rules and drag them to where appropriate. To delete them, simply select them and press the “delete” key.

Between the lower part of the canvas (5) and the Properties view, (7) you will find a toolbar related to the work area. The drop-down indicates a reference resolution: you can modify it by directly editing these values. On the right you will find the zoom indicator: you can enlarge or reduce the size using this drop-down menu.

The elements on the screen keep a certain order of depth between them. You can see in which order they are arranged in Outline of the actual screen (8). Thus those that are highest block out those below them. To change this order, click on the right mouse button over the item (or select it in the content view) and select the Order option of your choice.

The Image component allows you to visualize which parts of the screen will be occupied by images. Or, if you prefer, replace these representations with real images by selecting a file from the Properties view (7).
Working with the screens

- To create more screens in your prototype: Click on the “+” button located above the list of screens.

- Rename it by clicking on it with the right button of the mouse. Select the Edit option from the context menu.

- If you want to delete one, click on it with the right button of the mouse and click on the Delete option.

- And to edit the content of a screen that is not active, double-click on it from the list of screens. Its content will be loaded onto the canvas where you can change it easily.

- Justinmind Prototyper allows you to copy and paste screen components within the same screen or onto other screens. You can use the options in the Edit menu or the usual keyboard shortcuts of “Ctrl+C” (copy) or “Ctrl+V” (paste). To duplicate items, drag them to the desired position by holding down the “Ctrl” key or “CTRL+D” (Mac keyboard shortcuts are different).

**Remember:** Whatever action you do, you can undo it or redo it from the Edit menu.
Importing external content

**Justinmind Prototyper** includes various utilities so you can enhance your prototypes with content generated in other applications. Add texts, images and web content like videos, Flash or Google maps. There are various ways of doing this:

- Copy text from the source application and paste it onto the canvas. Or drag it from the application where it is held. It is not always possible to keep the styles (bold, italic, etc.); this depends on the original program they were developed in.

- To insert an image, drag it from the file system to the canvas.

- Or import it from the Properties view of a Picture item.

- You can also drag them directly onto the canvas from another application, including from any Internet browser. Copy an image made in Photoshop and paste it as an image in the prototype.

It also imports content from the Internet using Web Widgets: you will find them in the last group of widgets in the component palette (4). Reserve a space on the active screen so you can display an Internet address; to do this, drag the URL component to the canvas and indicate the web address that you want in the Properties view. Include videos from YouTube or Google Maps by copying the HTML of the sites you’re interested in and pasting them into the properties of the HTML component.
Interactive screen components

Most of the components in our palette are interactive. To do this, drop them in the canvas and click on the Simulate (10) button to activate them. For example, if you move the Form Text component into the canvas and activate the simulation mode, you can enter text in the fields which you have defined as if it were the final application. The drop-down menus are also interactive components: Indicate the values that the component must have from the Properties view (7). The Menu and Navigation Tree components are special cases.

- **Menu.** Drag it to the canvas and indicate the size that the bar will occupy. Move the elements to be contained inside it and put them in the desired position.

- **Navigation Tree.** Works similarly to the Menu: pre-select the required size and move the nodes you need inside it. In both cases you can customize the different parts from the Properties view.

Links

To convert text from your prototype into a link, paint it with a particular style (for example, the classic blue underlined highlight) and drag it to the screen you want to link to in the list of screens. The link will be set automatically. Therefore, in activating the simulation mode, you can interact with the components you have defined and navigate through different screens.

If you want to click only on a part of the text, place an Image Map over it and then drag it to the destination screen to generate the link. The Image Map is not displayed if you click on the Simulate button but the area it occupies will continue to be interactive.

The menu items and the nodes of the tree can also be moved to other screens. The only thing you need to do is drag them onto the screen you want (if you do it in reverse, the effect is the same).

**Remember:** Any element that is dragged to a screen will link the active screen to the destination one.
Browsing through your prototype

As you enhance your prototype with new screens, you will need to navigate between them. To do this, you have various options.

- **Browsing Utilities.** These are located to the left of the Simulate button, including:
  - View Home Page. This is the threshold of your prototype; through it, users will be able to enter other screens, and it is marked with an arrow. If you want to replace it with another screen, change it by clicking the right button of the mouse on the screen list and select the Home Page option.
  - Previous
  - Next

- **Navigation Mode.** This is another way of looking at your prototype structure. To do this, click on the Design drop-down and select Navigation: you will find it between the lower part of the canvas and the Properties view, on the left-hand side of the monitor. From here you will see what screens you can navigate to from the active screen. Each time you go from one screen to another, the application shows you all the available routes. This map is updated each time a navigation is added in the Design view.

Interactive Mockups

When defining an application, the graphic design phase is always included. Typically, the designer presents his proposals in the form of static images, for example, created with Photoshop, to represent a few screens. While the presentation and appearance of these images can be excellent, the inconvenience is that they are static images and don’t allow any type of interaction. It is therefore relatively easy to lose sight of the application’s usability and the navigation between screens. These graphic proposals are known as mockups.

**Justinmind Prototyper** now allows you to add interactivity to them with minimal effort.

1. Set up each of the mockups that you have on a different screen from your prototype (remember that you can drag files or images directly or copy and paste them).
2. Move the Image Map component over the mockup areas representing buttons or any other element that causes a screen change when clicking on it.
3. Drag the Image Maps over the screens that you want them to link to.

**Tip:** Even forms can be brought to life: place form fields over their graphic representation, remove borders, and adjust size and font. Now click on the Simulate button.
Sharing your prototype

At Justinmind, we think that a prototype is the best communications tool between customers/users and the software development team. If you have one at the start of the project and it is as close as possible to the final application, the user can be involved from the beginning of the project and give his opinion and validate it. The more key people get involved, the more successful the project will be as a whole, reducing the number of final adjustments to be made to the application and not on the prototype. Justinmind Prototyper has various ways to display or distribute the prototype among your target users.

- **By clicking on the Simulate button** and clicking on F11 (full screen mode) you can review the simulation together with the users. The main advantage is that you can introduce changes as and when they are requested, validating them immediately. The disadvantage is that this is done in situ.

- **Exporting it in HTML format.** Our prototype opens in any Internet browser and can be distributed to all the key users you want: they can run it and test it. The advantage is the system’s flexibility. The disadvantage is that many of them have anti-spam filters installed that block emails with attached HTML code.

- **With Justinmind Usernote.** Your prototypes will be accessible on the Internet for all authorized key users. They can test and comment on them in a collaborative way from their browsers and all of their notes are registered centrally so that details are not lost. Through this service you can also integrate your prototype on-line with test tools and remote users.

**Remember:** you can export all of the prototype information in one specification document to integrate it into the different documentation processes of an organization, or simply to have a document that can be signed.
Dynamic Content

Unlike static content, you can change dynamic content without having to switch screens; you will find a few examples here that will allow you to go deeper into the capabilities of Justinmind Prototyper to do a simulation. Mainly we will use the Dynamic Panels component, but the methodology can easily be extrapolated to other cases.

Tabs

These are one of the most common ways to organize information on a single screen. Each tab displays different content that is activated when a user clicks on it. Use the Dynamic Panels component to group content in different panels or layers and decide which one of them must be viewed.

- To create tabs, drag the Dynamic Panels component to the canvas.
- Once positioned, it shows us the area reserved for the active panel, its name and the “+” icon with which we can create as many layers as we need (in this example, three).
- Assign content to them by first selecting the Dynamic Panels component and then the grey label that corresponds to the panel you wish to work on.
- Remember that to add screen components, you have to drag them from the palette and drop them on top.

To give a tab form to text, align it at the centre vertically and horizontally. Then in Properties, indicate which border you want; you can customize it by indicating that the lower part should not be visible, or rounding off the corners to enhance the graphics. Once you have drawn the first tab, copy and paste it twice.

Once the graphic part is completed, add interactivity so that the content is displayed when the user clicks on the buttons you have simulated. You will find all the interactions that can be defined between the user and the selected item in the Events view next to the Properties view. Clicking on the Add Interaction button opens a dialogue box in which you can specify what actions must be taken after a user clicks on your tab (for example, display content). To do this, choose Show/Hide, select the panel you want to display, and indicate that it must be displayed: it is not necessary to hide the others because only one can be active. Then repeat the process with the other tabs. Click on the Simulate button to see how it displays different content based on the tab selected.

Tip: If, as well as changing the panel, you also want to highlight which tab has been selected, move the tabs into the panels and change the style of each panel.
Dynamic Forms

In some applications, forms for entering data can contain many fields. There are various ways to organize them so as not to inundate the user with information. One of the ways is to display only the essentials while still allowing the user to view the options.

- You can simulate them with Justinmind Prototyper in a similar way to the tabs:
- Draw the fields that you consider essential.
- Drag the Dynamic Panels component to the canvas and create two panels or layers.
- In the first one, add a label entitled “Display optional fields” and in the second add a label with the words “Hide optional fields.” Add a few fields below this one.

Define an event in the “Display optional fields” label so that when a user clicks on it, the corresponding panel is displayed. And define another event in the “Hide optional fields” label so that when a user clicks on it, the other panel is activated. In the intermediate level tutorial, you will learn how to stop the dynamic content from overlaying the rest of the content by using the Layout property of the Dynamic Panels.

Rich Interactions

In Justinmind Prototyper, the definition of events is done in the Events tab located next to Properties. To create one, first establish what action the user should trigger and second, what it will be.

Choose the user’s action from the options offered in the menu on the left-hand side. Remember that you can assign more than one action to a single component. For example, you can make a button have one reaction when clicked on and another when the cursor goes over it. In the actions menu, you will find which and how many have been identified for each trigger.

After choosing what the user needs to do, select which response the application should give. To do this, click on Add Interaction and configure it in the dialogue box that appears. On the left-hand side of this dialogue box, there is a list of options you can simulate.
- Link to: allows you to change the current screen for the selected screen or an Internet address (URL). If you indicate it, the new screen will be displayed as a pop-up.

- Data Master Action: are related to the simulation of dynamic data lists and functional forms. You will find a more detailed explanation in the last chapters of this document.

- Pagination: as previously stated, this is explained in the last chapters.

- Show/Hide: show or hide existing content in the current screen.

- Change Style: change colors, borders, background properties, etc. of a component from the current screen.

- Enable/Disable: makes an input form editable or non-editable.

- Set focus on: centers the screen on the item selected and activates it as an input form field.

- Set Value: modifies the value of a label, form field or variable as indicated by the user.

- Select Value: is only valid on the input form field for selection types (drop-down menus, lists, groups of checkboxes, etc.) and selects the values entered by the user.

- Pause: waits for the time indicated by the user before running any action. If combined with others, you can simulate animations; for example, a progress bar.
Once the response to the application is defined, click on OK: the action selected will appear in the Events tab. A single user action can trigger various actions in sequence; for example, to add another action that will run after the previous one, click on the arrow button next to the word Do. Or you can also add more cases by clicking on Add Interaction again. The cases are run from top to bottom and the actions from left to right.
Conditions

Justinmind Prototyper allows you to add conditions to events so they only run at certain times of the simulation. For example, simulate a “log in” form that allows access to the next screen if the correct username and password are entered; if they are not correct, an error message is displayed.

Draw it on a screen by dragging labels, form fields, and buttons to the canvas. Drag the Log in button to the welcome screen. In the Events tab, a navigation event is automatically created. On top of that navigation you will find the Add Condition option. Click on it to open the corresponding dialogue box:

In the Conditions Editor you will find a blank area where you can build them and a palette where you will find grouped items that can be used to define them. They are, in turn, divided into components and into functions. The components consist of form fields, text, variables, data masters or data master fields. You can select any of them and drop them into the gaps in the editing area. Meanwhile, the functions are the operations you can do to build on the previous condition.

Following the Log in example, we will build a condition to verify that the username field is equal to admin, and that the password field is equal to 1234.

1. Drag the “&” function into the editing area:

2. ____ & ____

3. In the first gap, release the “=” function and do the same for the other gap.
4. ____ = ____ & ____ = ____

5. Drag the username field from the screen to the first gap of the first “=” function and in the second, type admin:

6. user = admin & ____ = ____

7. Drag the password field from the screen to the first gap of the second “=” function and in the second gap enter 1234:

8. user = admin & contraseña = 1234

9. Click on OK to see the condition in the Events tab.

**Note:** It will be easier for you to select the items you want from the active screen if you use the zoom feature.

When the previous process is finalized, the “else” option will appear in the navigation event. Here we will establish that your application must be used if the condition is not met (in this example, send an error message).

- Drag the Label component to the screen and write the error message.
- Mark it as Hidden in the Properties panel.
- Select the Log in button and click on the “else” option in the Events tab.
- Check Show/Hide, select the error text and hit the Show option.
- OK and click on the Simulate button to verify that the simulation is correct.

Create very complex conditions as you need and link them to all cases of “if A does B” as appropriate. You can copy and paste them into other events and even into events of other components.

**Variables**

The variables system allows you to input and save data so that you can use it during the simulation of your prototype. You can thus ask your user to store information on one screen to use it on another.

For example, we will create a registration form that, once completed, will take us to the welcome screen where it will show us the name that the user entered in the “Customer” field.

- Draw the registration form using the component palette.
- In the Variables tab (located below the component palette), create one.
- Drag the “Customer” field from the registration form to the variable; whatever your user has written will be saved. You can also do it through an “on page unload” event.

- Open another screen and insert two text labels. In the first, type “Welcome”. Drag the variable above the second variable. This way you are indicating that when the screen is loaded, the text displayed will be that of the variable.

- Click on the Simulate button to verify how the text goes from one screen to another.

By dragging the “Customer” field and dropping it on top of the variable, Justinmind Prototyper created an event automatically. This event is of the “on page unload” type and replaces the value of the variable with the content of this field. But this is not the only way to initialize a variable: any “Set Value” action related to an event can do it. To do this, select a component and in the events dialogue, click on the “Variables” tab (this is next to the “Screen” tab) and select the one whose value you want to change.

The variables are another element that can be used in conditions. When the expressions dialogue box opens, you will see that just as you can drag elements from the screen, you can also do the same with the variables.
Templates

The templates system is a very useful tool for rapidly creating prototypes and facilitating any global changes you need to enter. On the one hand, it centralizes the content and interactions common to several screens, and on the other it defines the default styles of the components (colors, border types, fonts) in those screens where the templates are applied. This way, there is no need to go back and generate components or groups of components that are common to more than one screen.

In the above illustration, the header and the side navigation bar are the same on all screens. If you would like to prototype it with Justinmind Prototyper, we recommend that you mount the header and the side navigation bar on a template, and draw the core content on the screens of the prototype that corresponds to them.

In the Templates tab (located below the component palette) you will find all the templates you made to build the prototype. If you double click on the name of any of them, you will be able to access the content and work on it as if it were another screen. Remember that the components and even the events that are incorporated in a template are going to be common to all the screens.

It is important to know that by default the default template is applied to all screens.

- Create the templates you need by clicking on the “+” button of the Templates tab.
- To assign a template to a screen, click on the screen with the right mouse button, select Edit and click on the template you want.

In each template, you can define default styles in two ways:

- Right click on top of the template and click on Default Styles.
- Choose the component of the screen you’re interested in and select Assign as default style in the Properties view: all those of the same type will adopt the style you have
checked. However, if you hit the top button, the styles of the component will be replaced with those pre-defined in the template.

Export your templates from one prototype to another. This way, applying your corporate image will be quicker and easier than ever.

**Tip:** Get tabs to pass from one screen to another and always highlight the one corresponding to the active screen.

- Draw all of them in a template, define a “tab-off” style, apply it and establish the links required.
- On each screen, and above the template, paste the touch tab and assign a style that corresponds to “selected tab.”
- Click on the Simulate button and verify the result.

**Masters**

The masters are groups of screen elements that can be used throughout the prototype. In addition to their obvious usefulness in grouping elements, they are convenient for propagating changes globally. If we change the content of a master, it will completely modify all the screens where it has been applied. The Masters tab is just below the Components palette and is next to the Templates tab.

- To create a master, click on the “+” button and give it a name.
- Work with a master as if it were any screen: drag any item from the palette or define events (as long as they only act on the components within the master).
- Include a master in a screen simply by dragging it from the list. Once positioned, observe how its size adjusts to the content.
- Drag them to as many screens, and as many times, as you want.
- Enter the changes you need by selecting the master with a double click in the masters list; or if you have it on the screen, double click on it. Its contents will open on the canvas so you can work with it comfortably.
- Remember that the changes you make to a master will be made on all the screens. If you want to make exceptions, use the Break Master option: you will obtain a copy of the group of items that are independent of the original master.
- As with the templates, you can export your masters to other prototypes and share them with other users.
- If you delete a master from the list, you will delete all its remarks from all the screens. But you can delete a remark from a master on a screen without affecting the rest of the prototype.

Templates or Masters?

**Templates**: use them to define the content common to various screens which always appears in the same position and never has to go over an element on the screen. For example, headers, footers or the main browser.

**Masters**: for all other cases; content common to various screens that must be viewed in different positions, or that you want to stay on the screen at certain times above the screen content. For example, drop-down navigation menus, link cases, search engines, or registration boxes.

Widgets Libraries

They allow groups of screen components to be saved so that you can use them in other screens or other prototypes. The main difference from the masters is that if you make a change to one element, it will not affect the others that you dragged to the screen. They are perfect for preparing style guides. Go to the Extras section to download all the necessary libraries to prototype the interfaces of mobile devices.

Import your libraries by selecting Import in the My Widgets, tab in the Components palette, and choose the .jpl file you need. The elements of this library will appear as lists in My Widgets. Drag them to the canvas to work with them on all the prototypes you make with this copy of Justinmind Prototyper.

Simulate Drag and Drop

In this tutorial you will learn how to use the drag and drop events in Justinmind Prototyper. You will find some examples at the end of the tutorial.

Simple Drag and Drop
The 'on drag started', 'on drag' and 'on drag stop' events lets you add interactions at each state of the drag and drop. These events can be easily combined with 'move' and 'insert into' actions to prototype a wide range of drag and drop interactions.

The simplest case is to make any widget in your screen 'draggable'. That will make the widget move when the user drags it in simulation. To do so you just have to select the widget and select the 'on drag' event. Then push the 'add interaction' button and press ok with the default settings. That will create a 'on drag' -> 'move with cursor' interaction. Then push the simulate button and drag the widget wherever you want to.

The 'drop' case is defined using the 'on drag stop' event. Choose this event in the same widget you defined the drag interaction. Then push the 'add interaction' button. There select the action you want to execute when the widget is dropped. You can even constrain the 'on drag stop' execution to be triggered if you drop the widget on top of one of the other widgets in the screens. To do so just select the destination widget in the 'When drop on' option next to the 'Add condition' button.

**Drag Events**

- **On Drag Started**: Occurs when the drag begins
- **On Drag**: Occurs as the widget is dragged
- **On Drag Stop**: Occurs when the drag ends

**Actions**

- **Move**: This action moves a widget to a certain position in the screen. The movement position can be defined in two different ways:
  - **To position**: Use this option if you want to move the selected widget to a certain known position. The new position can be defined as absolute in the screen or relative to the widget that may contain the widget that is being moved during simulation. To define the position as absolute, click on the corresponding checkbox labeled 'Absolute position'.
  - **By offset**: Use this option if you want to move the selected widget by an offset defined in pixels.
  - **With cursor**: Use the move action combined with a 'on Drag' event in order to move a widget around the screen according to the mouse movement. A movement constraint can also be defined.
  - **To drag start position**: This option can be only combined with the 'on Drag' or 'on Drag Stop' events in order to move the selected widget to its initial position when the drag action started.
- Insert Into: This action inserts a specific widget into another container widget. When a widget is inserted into a container, its absolute position will remain the same if the container has a free layout. Otherwise, the widget will be inserted according to the container's layout positioning itself next to the closest container's child. Widgets which can be used as target containers for this action are dynamic panels, table cells or the entire screen. There is only one restriction for this action: container widgets cannot be inserted into any of their children widgets.

- On Drag Stop: Occurs when the drag ends.

These actions can be also executed with other events such as 'on click' or 'on mouse over'.

Effects and Animations

This tutorial will show you how to add fade and slide effects in your software prototypes.

Justinmind Prototyper provides several techniques for adding animation to a prototype. These include fade in/out and slide effects. In this tutorial, we'll closely examine each of the effects, revealing all of the mechanisms Prototyper has for providing visual feedback to the user.

The effects configuration is located inside the hide/show action in the events dialog. An effect can be associated to a hide or show action. So that if a fade effect is defined in a hide action that will simulate a fade out.

Fade

This option allows you to adjust the opacity of elements in simulation. The fade will last the time specified in 'Lenght' input. If you want to simulate a 'fade out' effect you have to select the hide/show action, then select the hide option and pick the 'fade' effect. You can define a 'fade in' effect the same way but choosing 'show' instead of 'hide'.

Slide

The slide effect animates the height and width of the matched widgets. Durations are given in milliseconds and specified in the length input; higher values indicate slower animations, not faster ones. You can add this type of effect to a hide or show action. You can define the direction for the slide effect using the 'side' select list. This 'side' refers to where the widget will slide in or out. So if you define a 'hide'+slide left' action that will hide the widget towards the left. If you select a 'show' instead, the widget will appear from the left.

Swipe Gesture

You want to simulate the swipe gesture in your mobile app prototype? It's so easy. Select the widget you want to 'swipe', then add an 'on drag start' -> 'hide + slide' interaction and that's it!
Design Patterns

These are some examples of what you can simulate with Justinmind Prototyper.

Interactive Images

In this example, we will use the Image Map widget to define events on areas of an image.

- Position an image on the canvas (in this case the Justinmind logo)
- Drag the Image Map widget over the letter “d” of the logo. Adjust its size so that it fits this letter.
- Add the Rich Text component beside the logo.
- Select the image map and create an event so that clicking the mouse hides the text.
- Click on the Simulate button and verify the results.

Upon clicking on the simulation, the letter “d” in the Justinmind logo text located beside it disappears. So it’s simple to animate areas of images and convert them into interactive images.

Changing the value of a text dynamically

We will simulate a text that, when clicked on, allows you to edit it directly, and check how the value of the original text is replaced with the one entered by a user.

- Drag a Dynamic Panels component and drag a text component in it. Remember that when you move a screen component to a dynamic panel, it is inserted in the active layer.
- Select the dynamic panel and create a new layer.
- Insert a text entry field into it and also draw a button.
- Select the button that you positioned in the second layer and click on Add Interaction (Events tab).
- Choose the Show/Hide button and indicate that you want the first layer to be displayed. Click on OK. You will see the event that you have just created in the Events tab.
- Click on the arrow located next to the “Do”. In the context menu that opened, click on Create Action to add a behavior to the existing one.
- Now choose Change Value, go to the first layer of the dynamic panel, and select the text element.
- Select the value by clicking on the Calculated option.
Build expressions from the dialogue box that has just opened. In the Screen tab, select the second layer of the Dynamic Panel and drag the text element to the top (in the dotted box).

Click on OK and return to the Events dialogue box. Click on OK again. All the necessary instructions have been provided so that when the user clicks on the button, the text that was entered goes to the first layer.

Go to the Events tab and add an interaction so that on clicking the text in the first layer, the second layer is displayed.

Click on the Simulate button and verify the result: the user has the opportunity to replace the original text with another after confirming it by clicking on the button we have drawn.

**Tooltip**

Select the element you want and enter the corresponding text in the Properties tab. So easy. If you don’t enter anything, no tooltip will be displayed.

**Customized Tooltips**

You can also customize your tooltips.

- Draw its content (for example, group together a rectangle and a rich text component).
- Select the group and mark it as Hidden in Properties so that when you do simulations, they are not visible.
- Drag any other element (for this case, an image). Select it, and in the Events tab, click on On Click option.
- The list displayed shows all the interactions that can be captured by the user. Now, we’re interested in on Mouse Over.
- Add an interaction to display the group.
- Click on Simulate. Verify how your customized tooltip is visible each time the cursor hovers over the image.

**Videos from YouTube and Google Maps**

In this instance, we will use the HTML widget. Go to the Web group of widgets.

- Drop the widget in the canvas.
- Open the YouTube page on your browser and choose the video you wish to insert.
• Click on the Insert button located below the video and copy the code that appears.

• Go back to Justinmind Prototyper.

• Select the HTML component you left on the canvas and paste the code in Properties. Then click on the Save button: the preview will already be loaded on the screen.

• You only need to resize the component so that the video reproduction will adopt the most convenient size.

Do the same with Google Maps:

• Go to the web and get the maps you need.

• Click the Link option and copy the code.

• Paste it in the HTML component as we did in the example of the video.

Scroll Bars

They are a property of the layers of dynamic panels.

1. Drag one of them to the canvas.

2. Go to Properties and indicate whether it should be horizontal or vertical.

3. Enter as much content as you want in this layer: the scroll bars will automatically update.

Conditional Drop-Downs

1. Change the contents of a drop-down menu according to what the user has chosen in another one.

2. Drag the drop-down elements to the canvas.

3. Select one of the two elements and go to Properties. Click on the Pencil icon (next to the Value drop-down) to edit its values.

4. Replace the default values by double-clicking on them and in their place type the names of countries.

5. Click on OK. Go to the Events tab and click on Mouse Click and, in the list that opens, select Change Value.

6. Then click on Create Interaction and choose Set Value from the actions on the left.

7. Select a second drop-down and add the cities of one of the countries (if you entered the USA, you can now enter New York, Washington DC and Chicago).
8. Now go to Add Condition to indicate that the three previous cities should only be displayed if the USA is clicked on in the other drop-down.

9. Release the drop-down with the names of the countries in the dotted box Double click to edit.

10. Drag over the “=” button located in the General functions tab.

11. The dotted box appears. Double click on it and enter “USA”.

12. Click on OK and do it again in the Events tab. Here you will see the description of the interaction you have just built.

13. By clicking on the text If not…, you can define the other cases just as you did with the first.

14. Click on Simulate to verify that each time you change a value in the country drop-down, other different ones are loaded in the city drop-down.

Jump between input fields using the Tab key

On forms with various input fields, it is very common for the user to jump from one to the other with the TAB key. Justinmind Prototyper will automatically simulate this behavior when you export your prototype in HTML format. The order is determined by the depth of the elements on the screen. But you can modify it as shown below:

- Drop various text entry fields into the canvas.
- Select the first and go to the Events tab.
- In the Click option, select on Key Pressed.
- Then click on Add Interaction so that Justinmind Prototyper shows you the Define Action screen.
- Unlike other cases, this time next to the Add Condition button, the element will appear that allows you to indicate which key will trigger the event. By default, pressing any key would be sufficient, so change this so it responds only to the TAB key.
- Click on the Pencil icon and then the tab key.
- Select the Set Focus On action and select the next text entry field.
- Click on OK and then on Simulate. Verify how the cursor jumps to the next after typing something in the first field and confirming it with the TAB key.
Simulating Data

With Justinmind Prototyper you can simulate data flow with real information. To do this, you have four components for storing it: Data Masters, Variables, Data Grids and Input Fields.

The data held in a Data Master, are called instances. They are available from any screen in the prototype and you can use them in calculated expressions, normally in combination with the functions of the Expressions Editor, grouped under the Lists tab. Unlike the Data Grids and the Input Fields, the variables contain data that will be available throughout the entire simulation and not just until you navigate to other screens.

Also, the Data Grids and the Input Fields store data. However, these components will only be maintained until you navigate to another screen where they will no longer be available.
Read Data

By adding a Data Master or a Data Grid to an expression, you are actually including the data contained in them. So, for example, the results that appear after you request a search in a Data Grid. And this way, when you drag a row to an expression, the data stored is entered. If you drag an Input Field, the value will be what your user has entered. In contrast, the variables will be taken as instances, lists of instances or texts, depending on the value type of the last event of Set Value that you have run on them.

The data stored in Data Masters, Variables, Data Grids and Input Fields can be used directly as elements of a condition or of a calculated value. But remember that only those that are saved in Data Masters or Variables will be accessible beyond the active screen.

Write Data

Use the Set Value event to modify the data of the Variables, the Data Grids and the Input Fields of the active screen. These components can receive values calculated from other data but keep in mind that in a Data Grid, the only values that can be assigned are an instance, or a collection of them. Only texts are allowed in input fields.

In the case of the Data Masters, the event Set Value is not possible. To use Create, Modify or Delete, find them in the Data Master Actions, when you need to enter data in one of them.
Dynamic Data Lists

Create a Data Master

In this example, we will simulate a list of products. Each product in the list of products will have an identifier code, a name, a brief description, and a photo.

- Click on the “+” button of the Data Master tab located at the bottom left of the screen. When the dialogue box opens, put the name in the Data Master; for example: “Product”
- With the “+” button (just below where you typed “Product”) add all the required fields and put the corresponding titles in the table on the right.

If you look closely, you will see in that alongside the Name column there is another one called Type. This sets the types assigned to the fields. Depending on the type, the information in the data lists and on the forms will be represented in one way or another. The default setting is Text.

- Select the text type in the row entitled “Photo”. Tick File in the drop-down menu that opens.
- Select the ID as the only attribute of the descriptor (use it to establish a Data Master as the data source of a drop-down item on the screen).
- Click on OK. In the Data Masters tab, what you have just created will appear.
- Drag the Data Grid component to the active screen. In the box that appears, you must specify an ID, select Data Master that will provide the data and the attributes that you will be using as columns. In this example, we will use the Data Master “Product” and we want all the attributes.
- Click on OK. It will display the space to be occupied by the table.
- By clicking anywhere on the canvas, a data grid with a dummy row will be drawn.
- Click on the Simulate button to verify the result. The table shows three rows of examples.

Let’s change these rows of examples for others that have more representative data.

- Double click on Data Master “Product”. You will go back to the dialogue box that opened when you created it. Now, however, you have a new tab available: View and Edit Instances.
- Click on it. You will see a table where each column corresponds to an attribute of the Data Master. This is where you will define the test data to be displayed in all the tables associated with the Data Master.
Enter some products and attach some images for the “Photo” field. OK.

Click on Simulate and see how the data we’re interested in appears, together with the photograph.

Customize Data Grids to form structured data lists. Continuing with the example, move the information from the cells called “Code”, “Name” and “Description” inside the column that holds the picture of the product (you will need to enlarge it to fit in all the information). Delete the remaining columns. You can also select the entire list and assign different colors for the even/odd rows.

Simulate Searches

Justinmind Prototyper has a section of functions that will allow you to perform calculations in Data Masters and in Data Grids. Use one of them now to simulate a search in the list of products.

- Draw a search box and add to it a text entry field and a button.
- Select the button and, in the Events tab, after clicking on Add Interaction, select the Set Value action.
- Select the Data Grid and click on the Calculated button.
- Note: it is important that you select the entire table, not just a cell or column: you’ll know that you did it correctly when the selection box has grouped both the header and row together.
- To continue, go to Expressions Editor, click on the Lists tab, and drag the Filter function to the top.

You will see two gaps in the function: the first gap corresponds to the data source you wish to filter; the second is the rule that will govern the filter function to decide which data should be filtered or not. Since we want to do a search that will cover all the products in this example, select the Data Masters tab and move the Data Master “Product” to the first gap. In the second, drag from the Screen tab the text entry field located next to the search button. Click on OK and OK again and then on Simulate.

Paginating Data Grids

Sometimes, the number of results of a data grid is so large that it splits into several pages. We will now explain how to simulate this type of pagination using the Index and Summary components and the listing actions. The first step is to prepare the data grid so you can use the pagination functions:

- Select the data table “Products” and click on the Properties tab.
- We are interested in “Rows Per Page:” indicate how many rows you want to display per page. In this case, 0 means no limit and therefore there will be no pagination. For this example, we’ll use 3 rows.

- Drag the Index component to the canvas and link it with the data grid using the data grid property inside the Properties tab. The index will automatically show the number of pages of the list based on the number of rows per page that you established.

- Note: this index is dynamic; if in the simulation the user makes a search, it will be updated.

Summary shows us the number of results available on all the listed pages and what values are being displayed at this time. It is configured in the same way as the index and is also a dynamic text. Finally, you can also simulate the actions: “next page”, “previous page”, “last page” and “first page”. The procedure is the same for the four options. Let’s look at the first one:

- Drag the 4 buttons and label them as “next”, “previous”, “first” and “last.”

- Select the first one and click on the Events tab, Pagination Action option and click on Next Page.

- The Expressions Editor opens. Drag the list from the screen to the gap in the expression and click on OK.

- Click on OK again in the Events dialogue.

- Click on Simulate. Check how the values in the list change each time you click on “Next.” For a better view, create a few more values in the Data Master.
Forms

Create

Do not delete the Data Master “Product”. In the next example, we will show you how to move data from one screen to another and store it at all times generating a Create Form. We’ll build a confirmation box that displays the values entered by the user; we’ll see how the new instance is created in “Product.” And finally, we’ll design a screen with the list of products.

The first step is to draw the form. Drag the screen component types Label and Text Entry.

**Tip: if you move the Data Master to the canvas, the input fields and labels are created automatically).**

We will simulate a very simple process consisting of three screens (Create Product, Confirmation and Product List). The first has a form where our user will enter the information that we want included for the new “Product.” The second will be the confirmation dialogue. The third will show all the products contained in our Data Master example, including the newly incorporated one.

The process will have these screens:
Add Product dialog:

- Create a variable for each of the data you want to store.
- Drag each item on the form to its respective variable
- **Tip:** This is a quick way to construct an event on page unload in which the value in the Input Field is stored in the variable.
- Drag the OK button to the Confirmation screen so that the link is automatically generated.
Confirmation screen:

The data in the variables is loaded on this screen. The “Confirm” button creates a new Data Master instance using the values of the screen (or of the variables).

- Drag each variable to the item where you want it displayed. This way you will create events quickly. When loaded, the screen displays the value stored in the variable.
- Select the button and click on the Add Interaction, then select the Data Master action option. Push the New button and drag to the expression gaps the variables corresponding to each attribute of the Data Master “Product”.
- Add another action to enable navigation to the List of Products screen.
Product List screen:

Finally all the products are displayed on this screen. In simulation mode, when the user confirms a membership, it will appear in the list. There is no need to create any event since by default all the values of our Data Master “Product” will be displayed.

Edit Form

Start the flow with a data grid in the canvas. We’ll extract one of the rows and move it to another form where the details will be displayed. Then we’ll add a button to confirm the changes that replace the original data.

- Go to the Product List screen from the previous example.
- Add a data grid that shows only two columns: one with the photograph and the other with the name of the product
Add a click event to the product name and choose the Set Value action. Select the variable Product and for the value, drag the row from the data grid.

**Note:** the understanding is that the row the user has clicked on will be put in the variable.

Add a second event to the product name that allows you to link to the Edit Product screen (you can only pass a selected row from one screen to another in the form of a variable).

**Note:** the variable stores all the information of the instance displayed in the row (in this case, all of the product information) and not just the information displayed in the columns. Therefore, on changing the screen, the variable contains a product, not an image and a text.
Design a form that shows the values of the row selected in the Product List screen. Remember that this row is available in the “product” variable, but you must choose which information you want to be displayed for each field.

- Generate an event on load page → Set Value in each field and use the Select function from the Lists tab to select the value.

- **Note**: the Select function allows you to say which attribute of a Data Master instance is to be used.

- Drag the product variable to the first gap and to the second drag the attribute of our Data Master product that corresponds to the selected field.

- Apply the same process to the rest of the fields.

To let the values entered by the user go through the confirmation screen, drag the fields to the variables. This way they can be stored. Finally, make the OK button you drew previously link up with the Edit Product Confirmation screen.
Edit Product Confirmation screen:

- Retrieve the values of the variables in the texts using on load page ➔ Set Value events.
- Create a new interaction in the Confirm button with the Data Master action Modify.
- Drag the “product” variable to the first gap. For the rest, move the variables.
- Establishes a link with the Product List screen.
- Click on the Simulate button. If you follow the flow from the Product List screen, you’ll see how the data updates in the data grid each time you enter a change.

Delete Form

To create a form that allows the user to delete a line in the data grid, repeat the steps you followed for the create form. Instead of indicating the Data Master action New, choose Delete. Although you can also avoid having a Delete screen and allow this action to be executed directly on the row in the data grid.

Completing a form with the data from a line on a list

There is another way to bring a line from a list to another screen and display its data in different fields of a form. Let’s see how.

- Create all the necessary events so that your user can select a row in the data table. Although you can do this in many ways, this time we will use the Radio button option.
- Draw a product data grid on the canvas.
Drag the Radio button component to one of the cells. This will allow us to distinguish which row will be displayed on the next screen.

**Important: we want Radio button, not Radio buttons**

- Select it. In the Events tab, click on Add interaction \(\rightarrow\) Set value. Choose Product in the Variables tab.
- In the lower section, click on the Calculated button to open the expressions dialogue. In it, you must indicate that the variable assumes the value of the row that contains the Radio button selected.
- Drag the Filter function (in the Lists tab). Put the whole data grid in the first gap. In the second, specify the condition for filtering: In this case, drag the Radio button.
- Click on OK. Click on OK again.

The event just generated will act in such a way that when a user clicks on the Radio button, the row in which it is located is copied to the Product variable.

- Draw a button and associate a link to another screen.
- In this other screen, design a form that will show the product’s details. To do this, drag text labels.
- Select the label in which you want to display the product identifier and choose the on page load event. Click on Add Interaction \(\rightarrow\) Set Value.
- Select the label and click on the Calculated button at the bottom. In the Expressions Editor, drag the Select function located in the Lists tab, to the dotted area.
- In the first gap, put the Product variable (this is in the Variables tab).
- Go to the Data Masters tab drag the attribute you want to display in that label. Click on OK.
- Click on OK again in the Events dialogue.

For the rest of the labels, repeat the same steps. To continue, double click on the Product List screen and then on Simulate. When selecting a Radio button in the data table and clicking on the button, the screen will show the details of the selected row.

**Log in using Data Masters**

In this example, we will simulate a registration form to validate that the username and password correspond to one of those already stored in a Data Master. We will call this one “User.”

- Create the “User” Data Master
• Add two attributes, one called User Name and the other called Password.
• Select the descriptor and click on OK.
• Move the new Data Master to the canvas and draw the fields on the registration form.
• Drag the button to the canvas.
• Link it with the screen that must show whether the username and password are correct.

Now you must add a condition in the Expressions Editor to compare the data entered by the user with those contained in the Data Master. Navigation will only be allowed if they match. Click on the Add Condition option in the events tab and follow these steps:

• Move the Filter function in the Lists tab to the expression.

  ```
  Filter (___ , ___)
  ```

  In the first gap, drag the Data Master “User”.

• **Filter** (Data Master User , ___)

  In the second gap, put the “=” function of the General tab.

• **Filter** (Data Master User , ( ___ = ___ ) )

  In the first gap of the “=” function, put the Username attribute of our Data Master.

• **Filter** (Data Master User , ( User Name = ___ ) )

  In the second, drag the input field that allows you to enter the username.

• **Filter** (Data Master User , ( User Name = ☐ ) )

  Drag the “&” function between the last parenthesis and the right parenthesis of the “=” function so that the first parameter takes the whole “=” function.

• **Filter** (Data Master User , [ ( User Name = ☐ ) & ___ ] )

  In the second gap, add another “=” function and repeat the steps you followed for Username.

• **Filter** (Data Master User , [ ( User Name = ☐ ) & (password = ☐ ) ] )

This expression returns the users who have this username and password. We need to convert it into a condition that will determine if there is any.

• Drag the Count function in the Lists tab in front of the Filter expression
\textbf{Count} [ \textbf{Filter} (Data Master User , \left\{ ( User Name = \square ) & ( password = \square ) \right\} ) ]

- Move the “=” comparator in the General tab and enter 1 in the last gap.

\textbf{Count} [ \textbf{Filter} (Data Master User , \left\{ ( User Name = \square ) & ( password = \square ) \right\} ) ] = 1

You now have an expression that covers all the values in the Data Master “User” and chooses those that contain a username and password that matches those entered in the form. It then counts how many users comply with this condition and if there is one, it OKs it.

\section*{Shopping Cart}

In the second example we will simulate the shopping cart design pattern used widely by online stores. We start from a list of products that the user can go to and add to his shopping cart. We will also add a button that confirms the purchase and will display the purchase details with the final price.

- Create a Data Master called “Product” with the following attributes: Name, Description, Photo and Price. All these must be text type except the Photo which will be a file type. And click on OK.

- Edit the Data Master and enter a few products with their corresponding prices using the View and Edit Records tab.

- Move to the canvas a Data Grid and select Data Master “Product”.

- Add a button to the row with the text “Add to cart.” To indicate that each time the user clicks on it, the product shown in the row is added to those already included in the “Product” variable, go to the Events tab, click on Add Interaction \textrightarrow Set Value and choose the “Product” variable from the right-hand panel.

- Click on the Calculated button and drag the “Product” variable to the gap in the expression.

- Drag the “+” function from the Lists tab so that the variable remains as the first parameter. Drag the row of the data grid as a second parameter.

- \textbf{Important}: you will know that you have selected the row because the cursor will change into a right pointing arrow.

- Click on OK in both Expressions Editor and Events.

In another screen, will we simulate a list with the products selected by the user and the total order price, with and without VAT.

- Now go to another screen of the prototype and drag a data grid component to it.
- Choose Data Master “Product”, select the screen’s background and, in the Events tab, select the on page load event.
- Click on Add Interaction → Set Value.
- In the right-hand panel, select the entire data grid and click on the Calculated button.
- Drag the “Product” variable into the gap. Click on OK. In the events dialogue, click on OK again. From this point, when the screen loads the data grid, it will show only the content of the “Product” variable.

Now we have to simulate the total price of the products and apply the VAT rate.

- Drag to the canvas three Label type components.
- Select the background and click on the arrow next to the word Click in the Events tab.
- Choose the Add interaction → Set Value option
- Select the first label and click on the Calculated button.
- Drag the Sum function, which you will find within the Lists tab, to the gap of the expression.
- Drop the entire data grid in the first gap.
- Note: if you correctly dragged the table below the value of the first gap, the text “data grid” will appear.
- In the second gap, drag the “Price” attribute from the “Product” Data Master. Click on OK.
- Click again on “OK” in the Events dialogue. In simulation mode, the label will show the summary of the products’ prices in the data grid.
- Click again on the word Do and select the Set Value option.
- Choose the third text that you left on the canvas and click on Calculated.
- Drag the “%” function to the expression (located within the General) tab. Put the second label in the first gap and the first label in the second (the one which contains the total).
- Position the “+” function so that the whole “%” function remains as the first parameter of the total. Drag the first label to the second gap. Accept
- Click again on “OK” in the Events dialogue. This label will display the price with the % applied to it.
- Edit the second label value and enter 18 (or the VAT rate that corresponds to the products you’re working on).
- Establish a link from the product screen/display to “Content of shopping cart”.

- Simulate.

Click on a few Add to Shopping Cart buttons. Click on the link and see how the data grid contains only the products you have selected in the previous screen, and that the labels show the total price. Moreover, you already know how much the package will cost including VAT.
Scrolling and Push Content

When the dynamic content exceeds the space we created for it, it is possible that it will overlap other elements close to it. To avoid this, you can use the scroll bars or the “push” feature.

Scroll Bars

The properties of dynamic panels include vertical and horizontal scroll bars. When they are activated, you will see that they update automatically each time you insert content in their layer. For example, drag a rectangle component inside so that part of it lies outside the panel area: it will activate by itself without you doing anything. This is a very useful option when you want the data grids to show a lot of lines and paginating is not convenient.

Push Content

This is the second option. Observe the distribution properties of a dynamic panel: if the vertical distribution is activated, the elements in this panel will be displayed one beneath the other. If any of them are not currently visible, the ones behind it will take up their space.

- Take a dynamic panels component to the canvas.
- Re-size it so that it takes up most of the available space.
- Create a Data Master and drag a data grid inside the dynamic panel.
- Move another dynamic panel inside the first one so that it is below the data grid.
- Within this second panel, put any screen component (a text, for example).
- Add a button in the row of the data table and click on Add Interaction (Events tab).
- Push the Data Master action Delete button and drag the row to the delete expression.
- Open the Data Master and enter a few examples of instances.
- Click on Simulate.

By pressing the button you created, the row is deleted in the data grid and the text below takes up the space that is freed up. Use this same concept with the events Show/Hide or with panels containing layers of different sizes.
Scenarios to Validate

**Justinmind Prototyper** offers a system that allows you to define navigation flows in a conceptual manner. We will call these flows “Scenarios” and you can work with them in the Scenarios tab, located next to the User Interface.

To create one, click on the “+” button located in the right-hand panel. All the ones that the prototype contains will appear here. Opening one is as simple as double-clicking on its name. In fact, the operation of the scenario editor and the distribution of information are very similar to that of the screens.

On the left you have a list of shapes or components that you can drag to the scenario. More specifically, you can move Screens, Decisions or Actions. You can also move screens created in the User Interface using the list located under the shapes list. Join the ones you moved with the arrow tool in the toolbar. Click on it and you will see how the cursor adopts the shape of an arrow. When you drag a shape on top of another with this tool selected, it creates an arrow linking the two and draws a navigation flow.

You can begin to define a prototype creating different navigation scenarios and then create a simulation that will adjust to them from the User Interface tab. Each time you create a screen whose name coincides with that of a Screen shape in a scenario, you will see a preview instead of an icon.

**Note:** if you click on the right button of the mouse on a Screen type shape in a scenario, you will create an equivalent screen in the simulation.

When two Screen type shapes are linked by an arrow and both are actual screens in the simulation, a warning symbol will emerge. This indicates that there is no event in the first screen to cause the navigation described by the arrow. If you add one and return to the scenario, you will see that the warning symbol has disappeared.

When you export the contents of a prototype to MS Word, all the information relating to the scenarios will be included.