HP TOUCHPAD

webOS Design Guidelines

Guidelines version

2011 MARCH 28 | **Release v2**

OVERVIEW

Who is this document for?

This is a preliminary version of the design guidelines for designers and developers interested in creating webOS experiences for the TouchPad device.

What is the purpose of this document?

These are general design recommendations meant to help you start thinking about how you might adapt your current webOS app design or create a new app for the TouchPad. We want to give you some general design parameters and suggested starting points for your design. Following these patterns will ensure consistency and familiarity for your application on the webOS platform.

Please use this document in conjunction with the wireframe stencils contained in the document *HP TouchPad Wireframe Stencil.* The stencils enable you to create wireframes for your application design and provide further detail about webOS Common UI components, layouts and application examples . This document provides a more general context for the webOS UI framework and provides guidance on how to put the building blocks contained in the stencils together into complete designs.

REVISION HISTORY

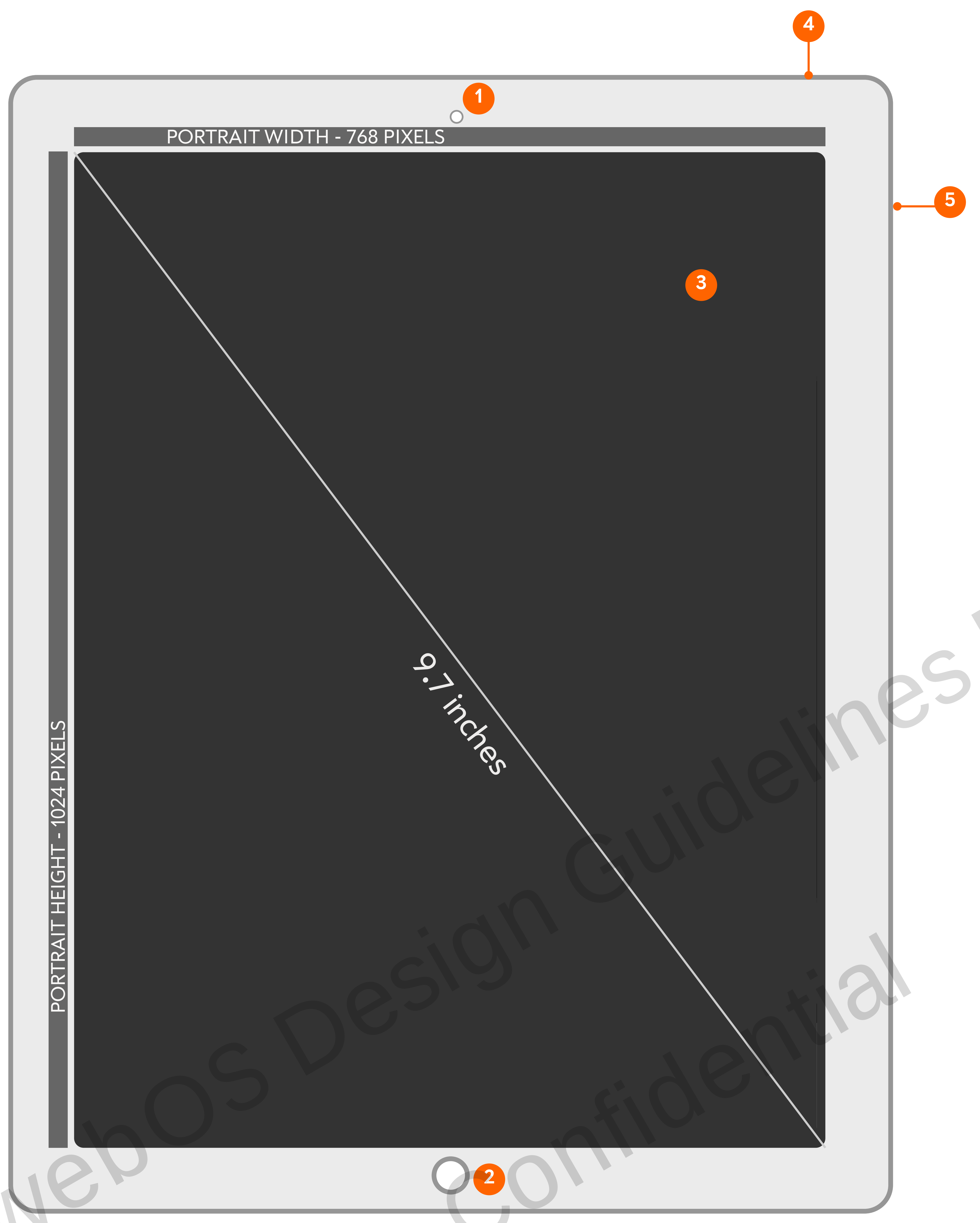
Date	Document Version	Remarks
2011-03-11	Version 1	Early access release
2011-03-28	Version 2	Released as part of the Enyo beta SDK

CONTENTS

HARDWARE FEATURES

This section describes the key hardware characteristics of the HP TouchPad that you should take into consideration when designing your application.

KEY HARDWARE FEATURES



1 Front-facing camera
2 Center button
3 Capacitive XGA display
4 Power button
5 Volume keys

DISPLAY CHARACTERISTICS

The HP TouchPad uses a 9.7 inch capacitive XGA (1024p x 768p) display. This gives a pixel density of 132 pixels per inch (PPI).

HARDWARE CONTROLS

The HP TouchPad has the following hardware controls:

- **Center button:** used to take the user to the Card view (see next page)
- **Power button:** used to switch the device on and display power management options
- **Volume keys:** use to adjust global volume levels

NO GESTURE AREA

The physical gesture area present on some webOS devices has been omitted from the TouchPad. Actions previously tied to back and forward gestures should be accomplished via direct manipulation (e.g. Sliding Panes to show/hide them - see Sliding Panes section) or explicit buttons (e.g. “OK”, “Cancel”, “Done”)

AUDIO AND VIDEO

The HP TouchPad has a front-facing 1.3-megapixel webcam, internal stereo speakers and a microphone.

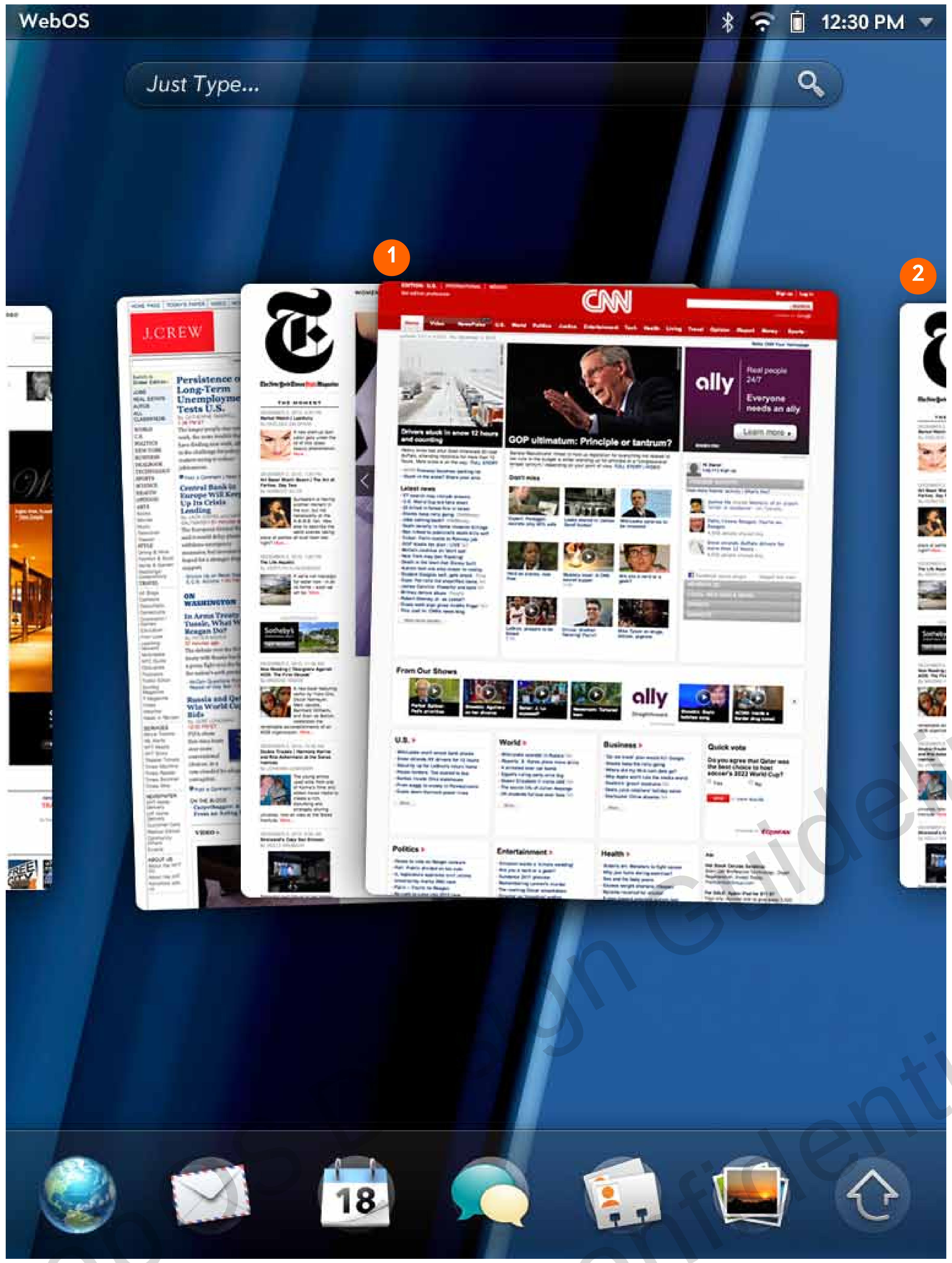
SENSORS

The HP TouchPad has an accelerometer for detecting device orientation.

FRAMEWORK FEATURES

This section describes the key features of the webOS framework that your application can utilize and integrate with.

CARD VIEW AND MULTITASKING



1 Card Stack

2 Application Card

CARD VIEW

The card view is the first view that the user sees after booting the device and can be accessed at any time by pressing the device Center button, or by swiping up from the bottom of any application view.

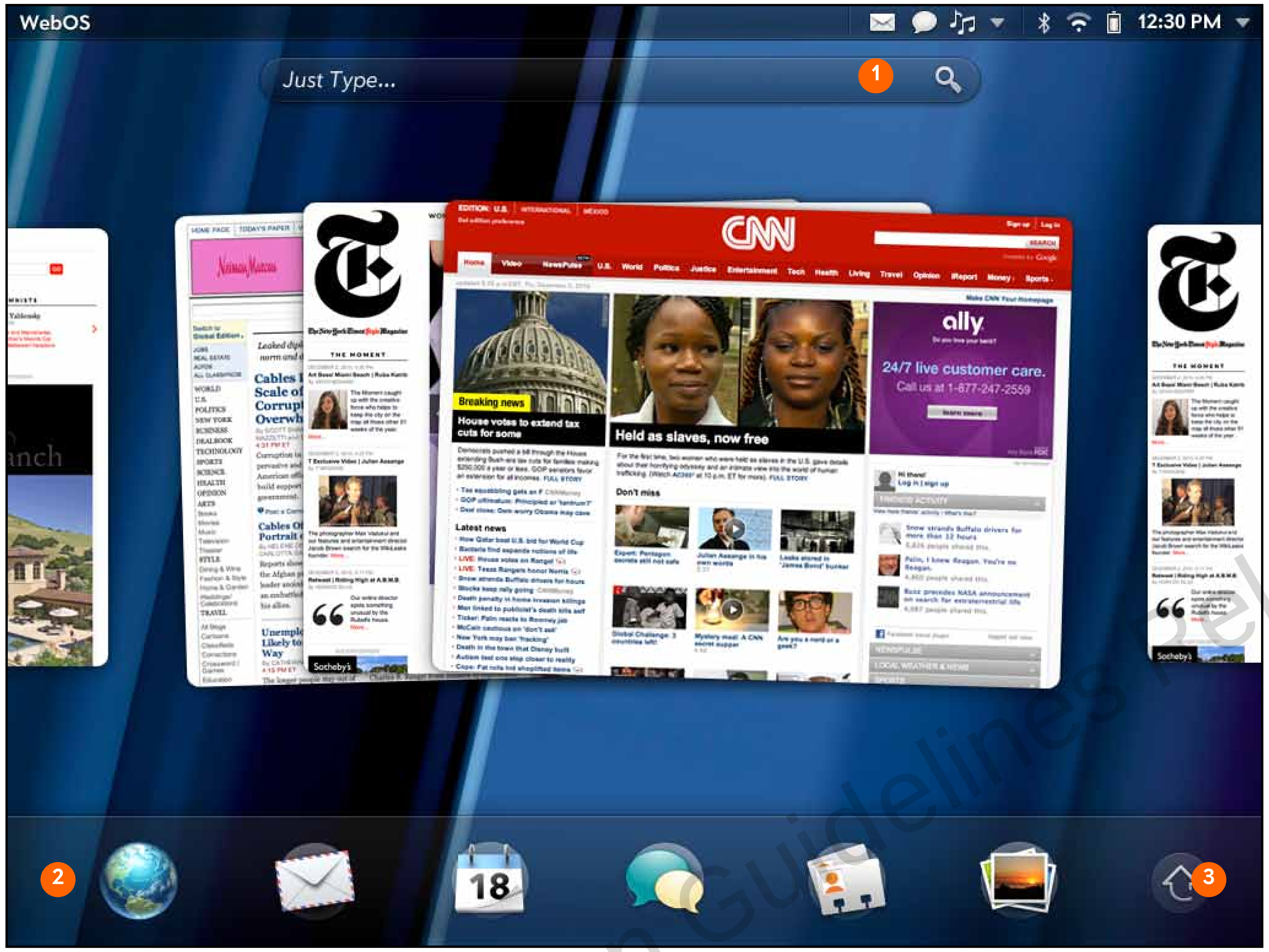
HP webOS provides a powerful multi-tasking framework that allows users to switch between concurrent activities using the device's Card view. Each activity - for example, a browser window - is represented in the Card view as a Card. In the Card view, the user can switch to another activity simply by scrolling to and tapping the card representing the activity.

Cards appear as scaled down versions of the full sized applications. Cards will continue to update in Card view to reflect the latest state of the activity in the application.

CARDS ARE ACTIVITIES, NOT APPLICATIONS

Cards are not the same as applications. An application can create several Card instances in the Card view. Each of these Cards can represent a concurrent activity within the application. For example, the email application creates a new Card for each instance of an email composer. All the Cards created from the email application are associated with each other automatically. Associated Cards are shown grouped in the Card view in **Card Stacks**.

CARD VIEW AND MULTITASKING



1 Just Type Search Field

2 Quick Launch bar

3 Application Launcher icon

MULTIPLE APPLICATION STACKS

An application can also launch Cards from other applications. For example, an application could launch an instance of the browser when the user taps on a URL. All Cards that are created by the application are shown grouped in stacks.

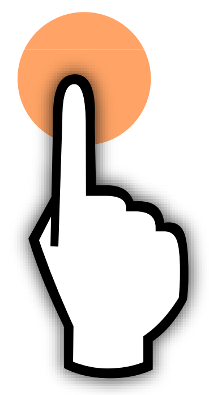
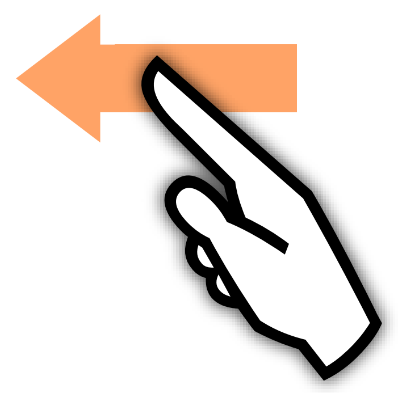
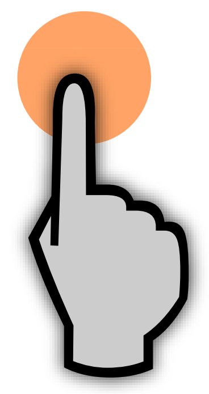
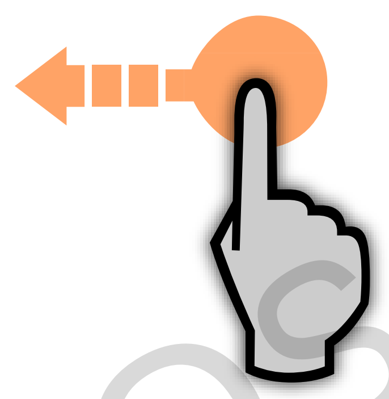
USER ARRANGEMENT

Ultimately, the user has control over how the Cards in the Card view are organized. By pressing and holding on a Card, the user can drag and drop the Cards to change the order in which the appear. Users can also re-arrange Cards within a stack, or pull a Card from a stack and show it individually. Conversely, the user can also group any Cards together into stacks.

APPLICATION LAUNCHER AND JUST TYPE

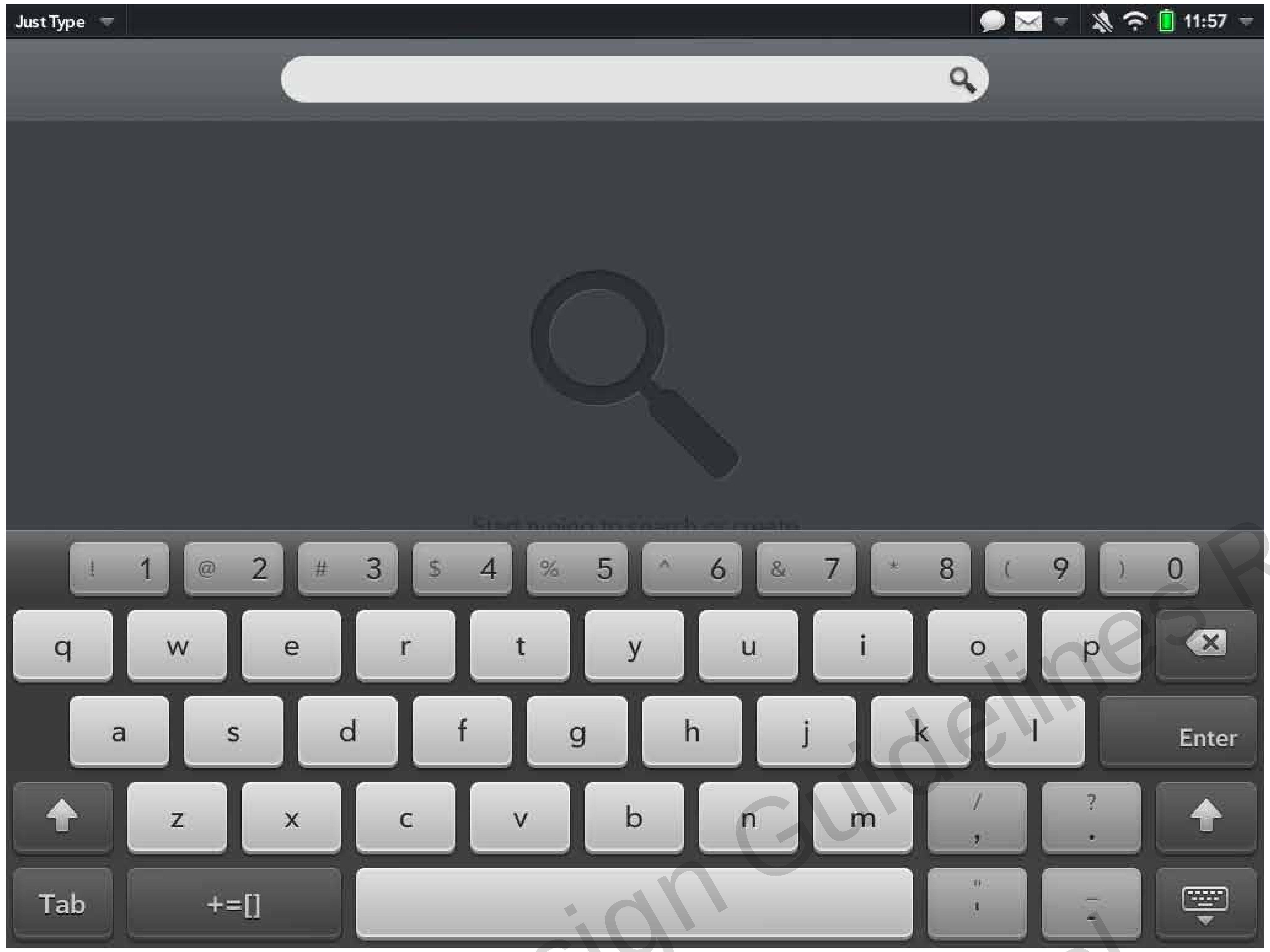
The Card view is also used as a springboard to other system features. Just Type is accessed by tapping on the search field at the top of the display and applications can be launched by tapping on the application shortcuts in the Quick Launch bar, or by opening the Application Launcher, which is accessed by tapping the Application Launcher icon in the Quick Launch bar.

GESTURES FRAMEWORK

GESTURE		TYPICAL ACTIONS
TAP		Open or launch an item; select or deselect an item; place cursor within a text field.
DOUBLE TAP		Zoom content in/out; select a word within a text field.
FLICK		Scroll left, right, up & down.
PRESS & HOLD		Enter edit/re-order mode in Card view and in grids and lists. Invoke Cut/Copy/Paste functionality in text fields and in web views.
PRESS & DRAG		Scroll content; move an object or re-size a selection.
PINCH IN		Zoom content out.
SPREAD OUT		Zoom content in.

TEXT INPUT

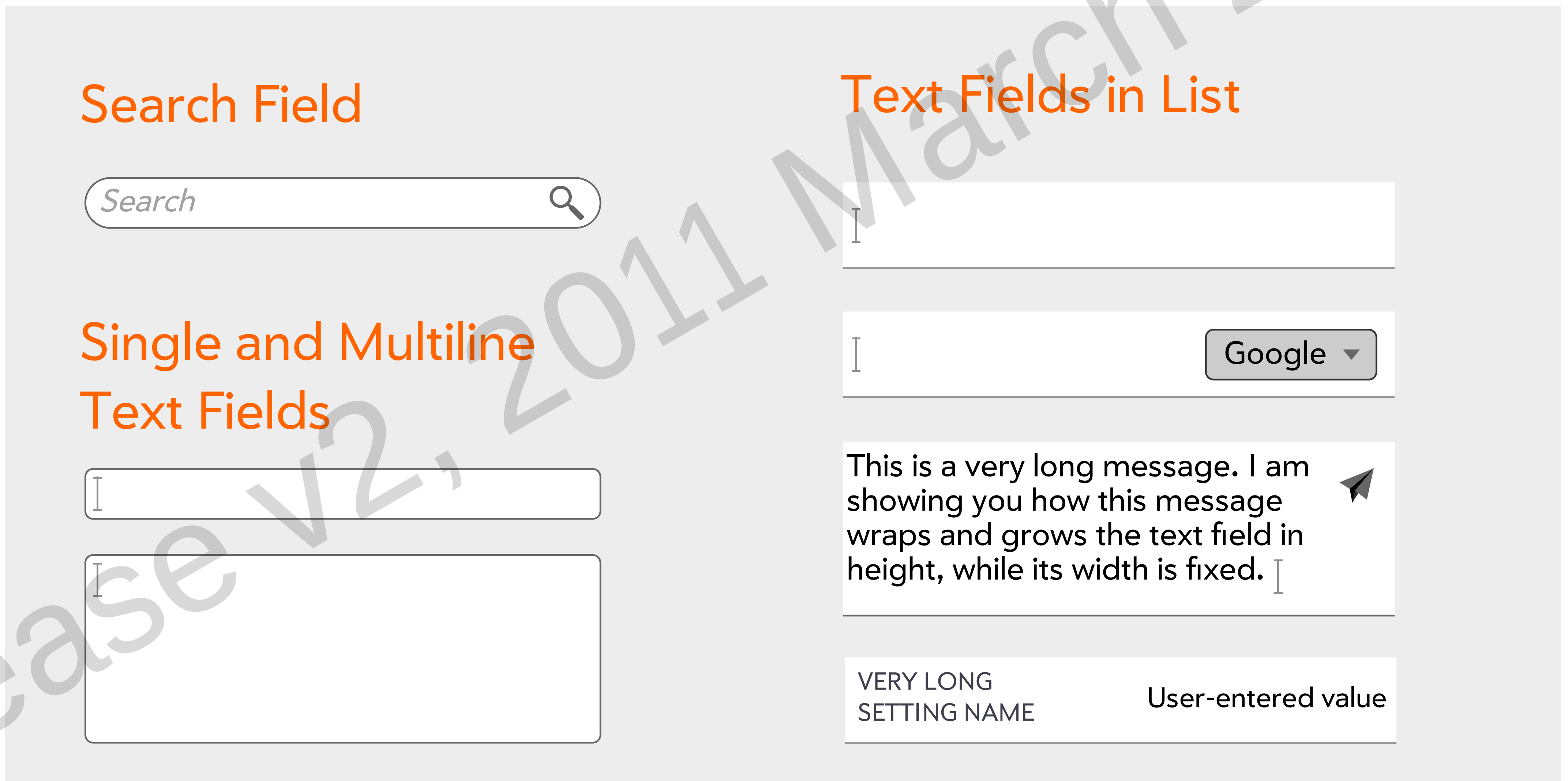
VIRTUAL KEYBOARD



DESCRIPTION

- The TouchPad has no physical keyboard - text is input instead by means of a virtual keyboard , which is invoked whenever a text field is focused. The keyboard is generally dismissed if the text field loses focus, except in the case of Interactive Pop-Ups.
- When invoked, the virtual keyboard rises from the bottom of the display and covers any content beneath. The user can still interact with content in the view that is not covered by the virtual keyboard, for example, to scroll content, or tap on buttons. The text field that is focused should always remain visible when the virtual keyboard is present. If necessary, the view will scroll automatically so that the text field is still in view.
- Note, the user can set the size of the keyboard to extra-small, small, medium and large configurations. It is recommended that you plan your application content to be laid out around the large keyboard size so that important information is not covered by the keyboard when it is invoked.
- A number of standard keyboard configurations are available: default (alphanumeric), email, form navigation and URL. Each keyboard layout is optimized for a particular type of text input. The application can specify which keyboard layout should be invoked depending on context.

TEXT FIELDS

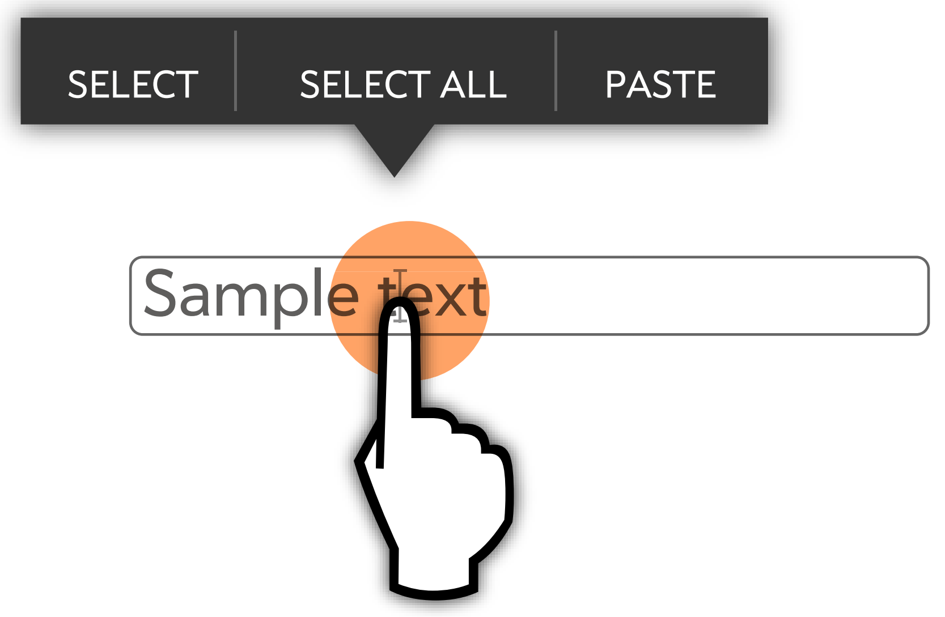


Search Field
Search
Single and Multiline Text Fields
Text Fields in List
Google
This is a very long message. I am showing you how this message wraps and grows the text field in height, while its width is fixed.
VERY LONG SETTING NAME
User-entered value

DESCRIPTION

- The webOS framework provides many ready-made text field components.
- A search field variant exists which contains a search icon and hint text.
- Text fields can be single or multi-line and can appear on their own or within List and Group Boxes.
- Text fields can contain embedded controls and labels.
- For further information, please refer to the *TouchPad Wireframe Stencils* documentation.

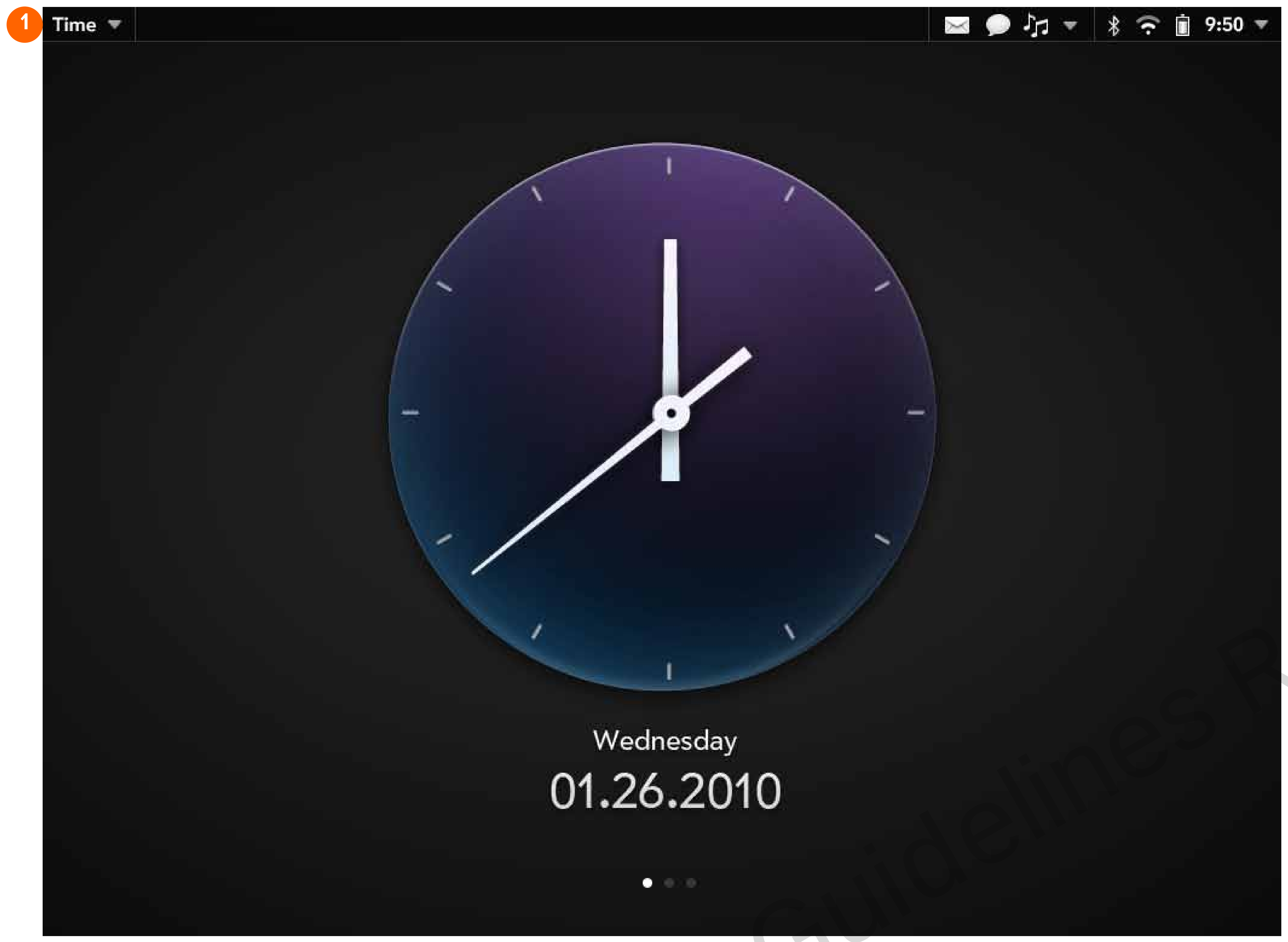
CUT, COPY & PASTE



DESCRIPTION

- Cut, Copy and Paste functionality is a built-in feature of text fields.
- A double tap or long press on a text field invokes a Context Menu that allows the user to choose select text or paste text at the cursor.
- Once a text selection has been made, the user can opt to cut, copy or paste over the selection.

EXHIBITION MODE



1 Application Menu

A NEW MODE FOR PRESENTING YOUR CONTENT

Exhibition mode is launched automatically when the device is placed on the HP Touchstone Charging Dock, or it can be launched manually from the Application Launcher. Applications can be written to run in Exhibition mode, either as an extension to your existing application, or as stand-alone Exhibition mode apps, allowing you to create all-new application experiences.

The user can switch between different applications in this mode using the Application menu. A few Exhibition mode applications will be built-in, such as a Clock application, but our hope is that it will be you, our developers, who create truly great experiences for this mode.

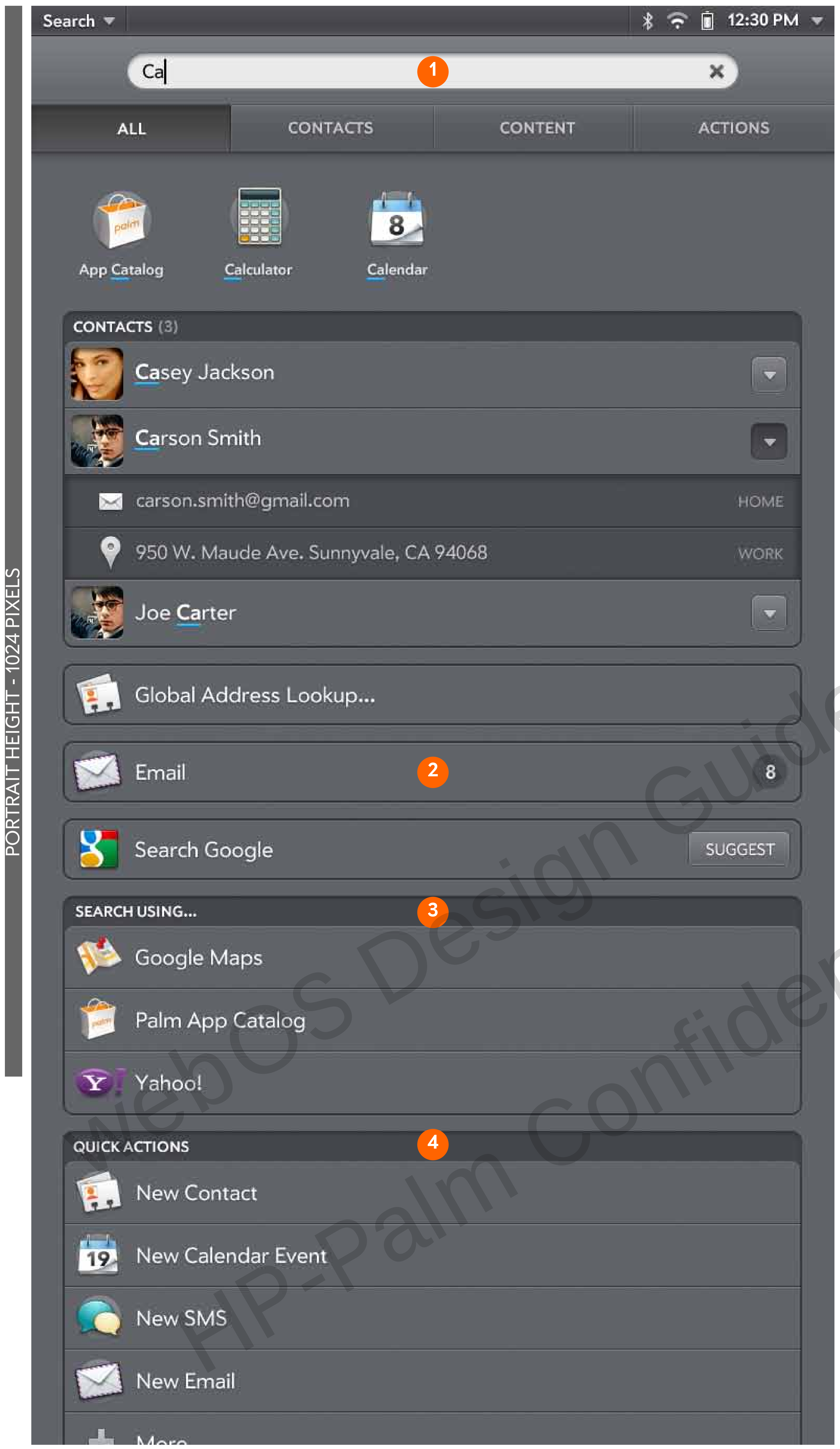
FULLY FUNCTIONAL

The full range of system functionality and UI components are available to applications in Exhibition mode. You have free reign to create as you would in any other view in your application. You can also use Exhibition mode to cross-link to your regular application and exit Exhibition mode.

DESIGNING FOR EXHIBITION MODE

Focus on creating experiences that are suitable for this mode. Users are likely to be interacting with the device only occasionally and while concentrating on other activities - therefore, design your Exhibition mode application so that it requires minimal input and to automatically refresh itself with content. Keep content visual and easy-to-scan so that it can attract users' attention, even though they may be focused on other things or may not be close to the device.

INTEGRATE YOUR APPLICATION WITH JUST TYPE



1. Just Type search field
2. Application Content searches
3. Online Content searches
4. Quick Action list

A POWERFUL UNIVERSAL SEARCH AND MORE

The Just Type application is triggered whenever the user starts to type text within the Just Type search field in the Card view. As the user types, search results are shown incrementally in the search view. Your application can harness the power of Just Type in the ways described below:

SEARCH DATA WITHIN YOUR APP

Just Type will search through application data stored locally on the device. You can expose data from your application to be searched by Just Type. If matches are found in your application data, the application is listed amongst the Application Content searches. Tapping on an Application Content search will show a full list of search results, and in turn, tapping on a search result can launch your application and link to the appropriate place.

SEARCH DATA ONLINE

Just Type can also be used to trigger an online search. By integrating with Just Type, your application will be listed among the Online Content searches. Tapping on an Online Content search will launch your application and pass along the search term.

TRIGGER QUICK ACTIONS

Just Type also displays a list of Quick Actions - such as creating new content, updating your status, etc. - all of which can be performed without the need to launch an application. Your applications can define one or more Quick Actions that link to functionality in your application and make them available to users directly from the Just Type app.

SYSTEM SERVICES

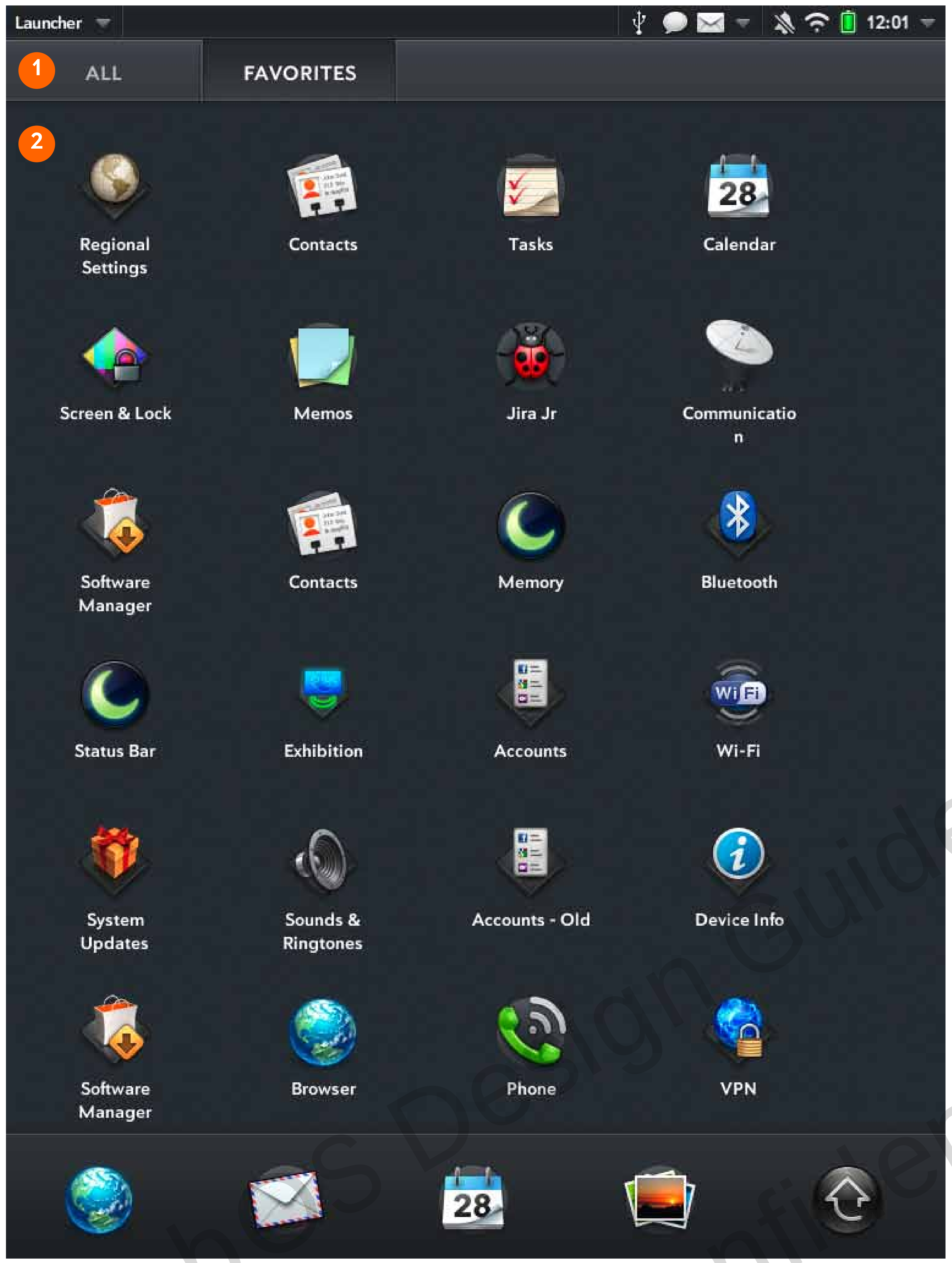
SERVICE	BRIEF DESCRIPTION
Accelerometer	Captures orientation events and accelerometer data.
Accounts	Returns information on established user accounts for use with the HP Synergy feature information manager.
Alarms	Sets a timer to activate on the device either after a specified interval or at a specified date and time.
Application Manager	Invokes default handlers for the common resource types or basic device operations.
Audio	Plays or streams audio by using common audio formats.
Browser	Loads and views the target specified by a URL.
Calendar	Various methods for accessing or creating Calendar data.
Camera	Launches the Camera application to take a picture.
Connection Manager	Gets connection status and subscribes to notifications of connection status changes.
Contacts	Various methods for accessing or creating Contacts data.
Display Manager	Gets events related to the status of the display.
Document Viewers	Launches the DocViewer application to browse and view common document file types.
Download Manager	Uploads and downloads files over HTTP.
Email	Sends an email, and includes options for pre-populating the email contents.
GPS	Gets the current location coordinates and registers for continuous updates.
Keys	Gets keypress events from headset and volume buttons.
Maps	Displays a map based on the various input options.
Messaging	Sends an IM/SMS/MMS, and includes options for pre-populating the message contents.
People Picker	Displays a list of contacts for the user to make a selection.
Phone	Launch the phone dialer with or without a pre-populated dial string.
Photos	Views an image in various common image formats.
Power Management	Enters Sleep mode after period of inactivity.
System Properties	Gets the named system properties, including device ID.
System Service	Accesses various system settings, including system Time.
System Sounds	Plays audio feedback in response to user interaction. The sounds play when the message is played, with low latency.
Video	Plays or streams video by using common video formats.
View File	Downloads and/or views a file in various formats or resource types.

This table summarizes the system services available to your application. For further information, please refer to the Service API documentation in the SDK.

INFORMATION ARCHITECTURE

This section will cover the basic building blocks and organizing principles for structuring your application

APPLICATION LAUNCHER



1 View tabs

2 Application icon grid

The Application Launcher is used to launch all applications on the device, with the exception of Exhibition mode applications. Each application is represented in the Application Launcher by an icon within a grid view. Tapping on the application icon launches the application inside a new card.

The Application Launcher shows a grid for all applications and a grid for Favorites. The user can switch between the two grids by tapping on the view tabs.

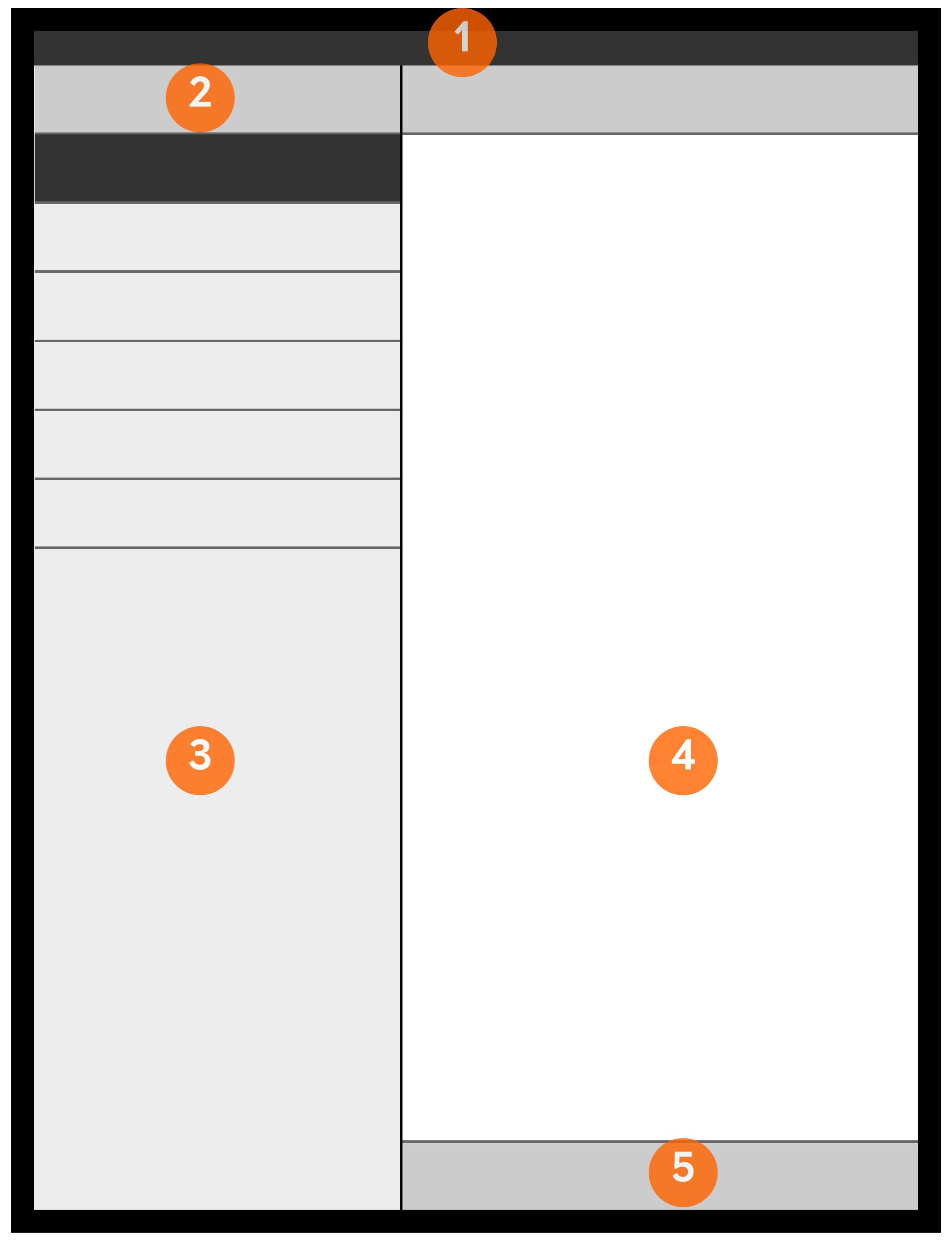
The Application Launcher can be accessed from the Card View by tapping on the Application Launcher icon, or by swiping up from the bottom of Card view.

MANAGING APPLICATIONS

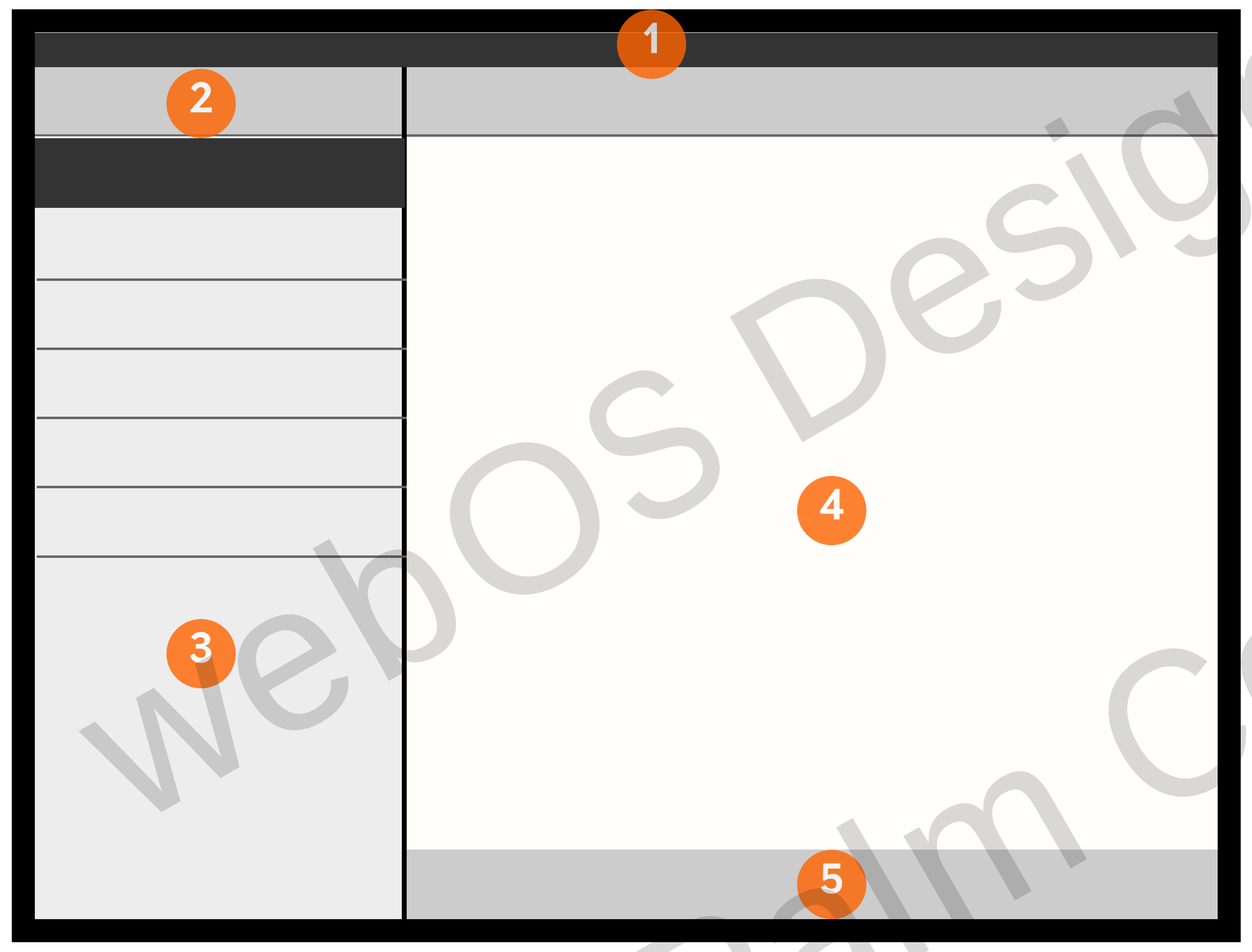
The Application Launcher also provides a range of functions for managing applications. Users can enter an Edit mode by performing a Press and Hold on any application icon. In this mode, users can re-order icons in the grid, delete applications, and add and remove icons from the Favorites tab and the Quick Launch bar.

VIEWS AND PANES

PORTRAIT LAYOUT



LANDSCAPE LAYOUT



1	Status bar
2	Pane header
3	Fixed pane content area
4	Free pane content area
5	Pane footer

The webOS framework provides a great deal of flexibility in creating view layouts for your application. You can configure your layouts with the following components.

STATUS BAR

The status bar is generally visible on most screens as it provides an area for key system indicators, such as battery and network strength, as well application notifications. See the **Notifications** section for more details. The status bar, however, can be dismissed to provide a full-screen experience. See the **Full Screen** layout section for more details.

PANES

An application can consist of one or more panes that contain application content. Application content can be freely assembled either using Common UI components provided by the webOS framework, or using custom components and application-specific content. Content within the content area is scrollable. Panes can either be fixed or free - fixed panes maintain a fixed width regardless of layout orientation whereas free panes adjust their size to fit the available space.

HEADERS AND FOOTERS

Each pane in the application can also contain a Header and Footer area to contain application controls and navigational elements. Headers can either be fixed or scrolling. Fixed headers remain anchored to the top of the pane whereas scrolling headers scroll with the application content. Footers can be used to contain a fixed tool bar that contains icon buttons.

Please refer to the Designing Your Application section and HP Wireframe Stencil document or more details on common UI components and how to layout content and controls.

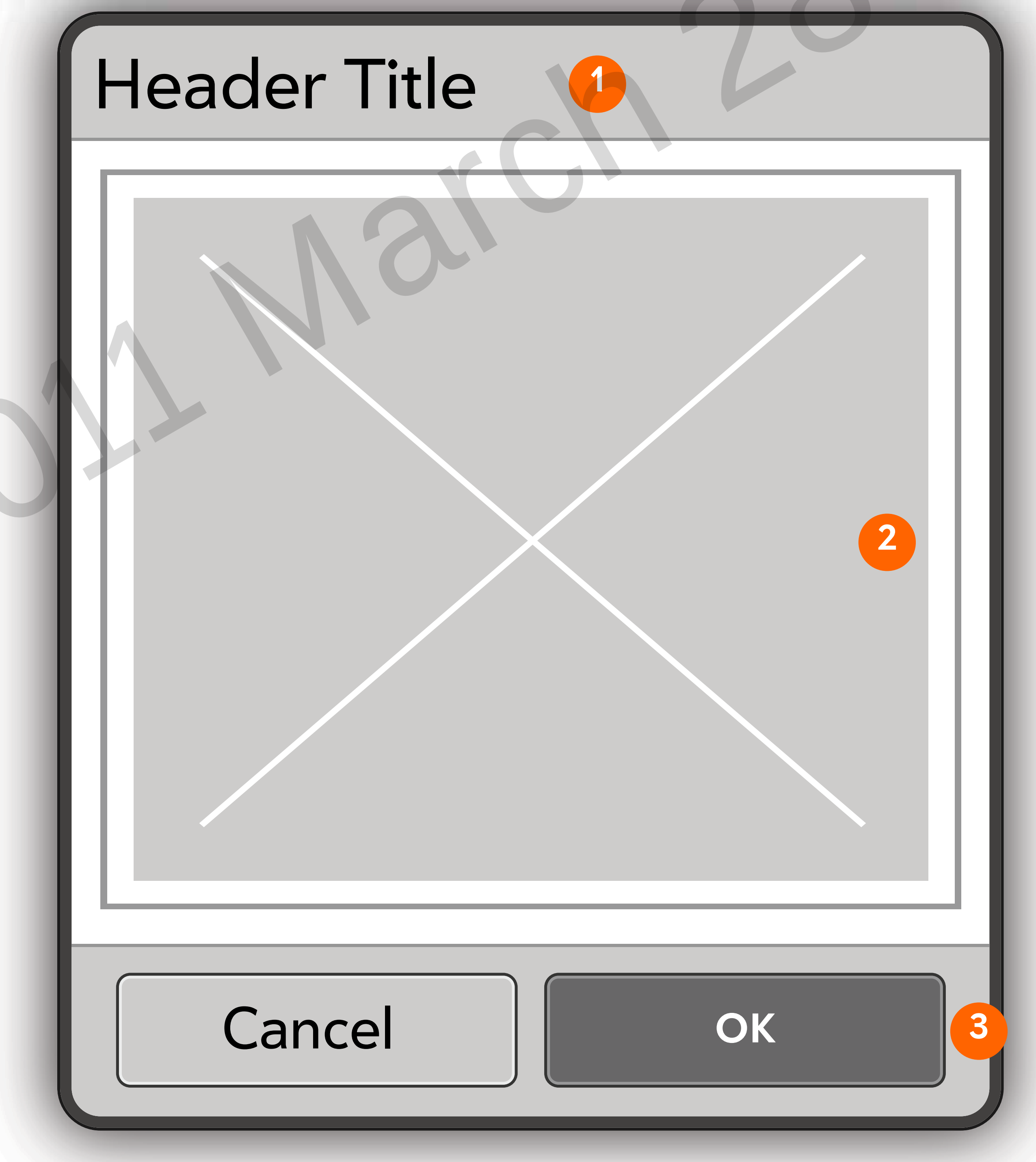
INTERACTIVE POP-UPS

What they are

Interactive pop-ups are modal elements that are displayed above a view. Its modal nature means that the user cannot interact with the view beneath while the pop-up is displayed. As the name suggests, interactive pop-ups contain interactive content that is displayed inside a pop-up. The content in the content area varies freely depending on the application and the task at hand. The content area is scrollable vertically if the content does not fit within the vertical space. Pop-ups can contain either a single level of content or multiple levels.

In addition the content area, pop-ups also contain a button area and an optional header area. Headers may sit in the chrome or scroll with the content. They may contain a title or instructions for the user. They also may or may not be editable.

Buttons always sit in the chrome and cannot be scrolled. Buttons may be stacked or placed side-by-side. In general, actions buttons (buttons that trigger certain actions, e.g., Print) are stacked and navigation buttons (buttons used to confirm/cancel or navigate between scenes) are shown side-by-side. If buttons are displayed side-by-side, no more than 2 buttons should be shown in the pop-up.



1 Header area - Title or instructions

2 Content area

3 Button area

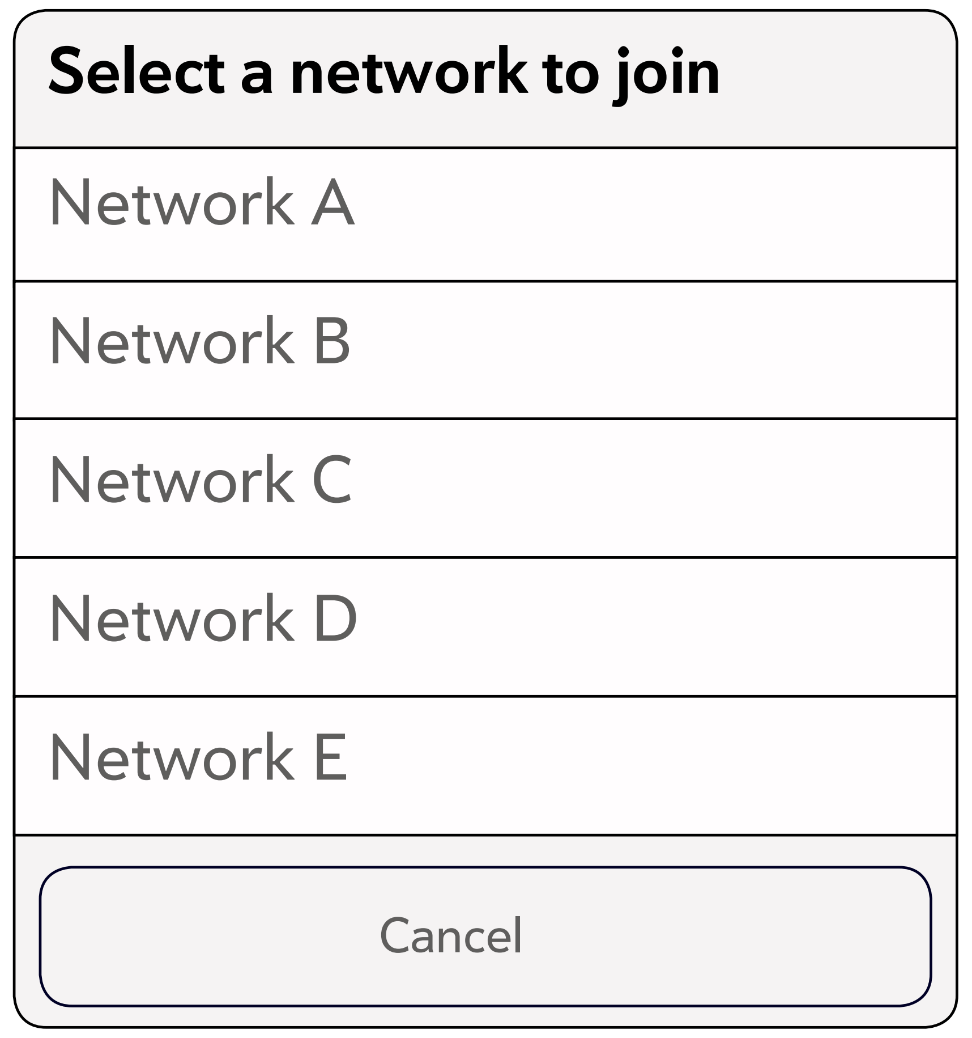
When to use

Use interactive pop-ups to perform small, self-contained, sub-tasks that the user must perform to accomplish the main task at hand. These sub-tasks should be relevant to the scene from which they are launched. On tablet devices that have large screens, navigating away from the scene to perform these smaller tasks can be disorientating. Interactive pop-ups are therefore allow the user to remain within the context of the scene that they are in.

Nevertheless, avoid the temptation to do contain too much interaction inside a pop-up. Although multiple scenes are supported, interaction inside a pop-up should be brief and contained. An interactive pop-up should not be the main way for the user to interact with the application.

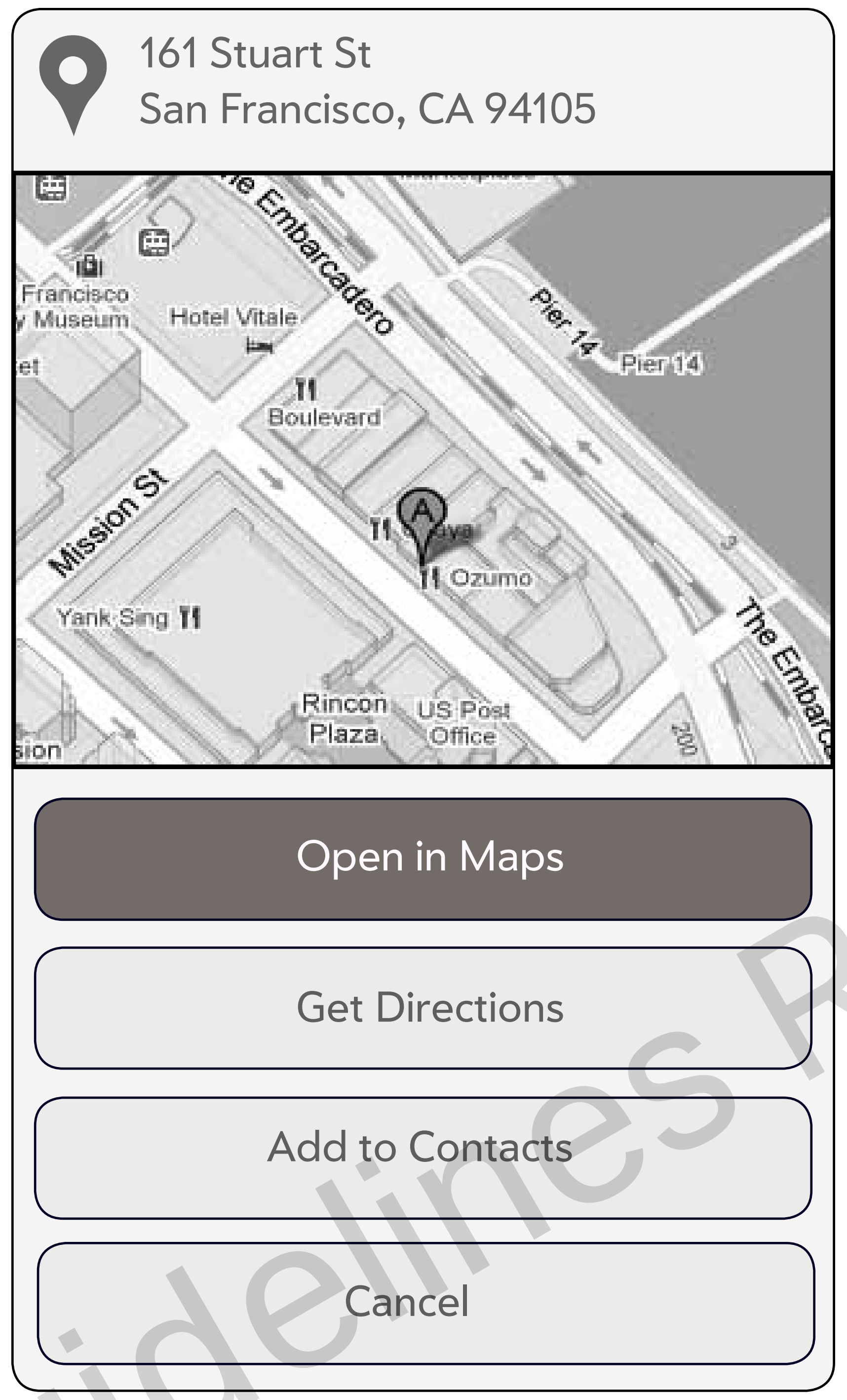
INTERACTIVE POP-UPS: EXAMPLES

SELECTION POP-UP



- Use this style of pop-up to allow the user to make a single selection from a range of options
- A header can be used to describe the selection list
- The user selects by tapping on one of the items in the list
- Selecting the item also confirms and closes the pop-up
- Dialogs of this style should contain a button to dismiss the pop up without making a selection

MULTI ACTION POP-UP



- Use pop-ups to give the user a selection of actions.
- Use vertically stacked buttons to present these choices to the user
- A key action can be shown using a Primary button - there should only be one Primary button in the pop-up
- Dialogs of this style should contain a button to dismiss the pop up without saving changes

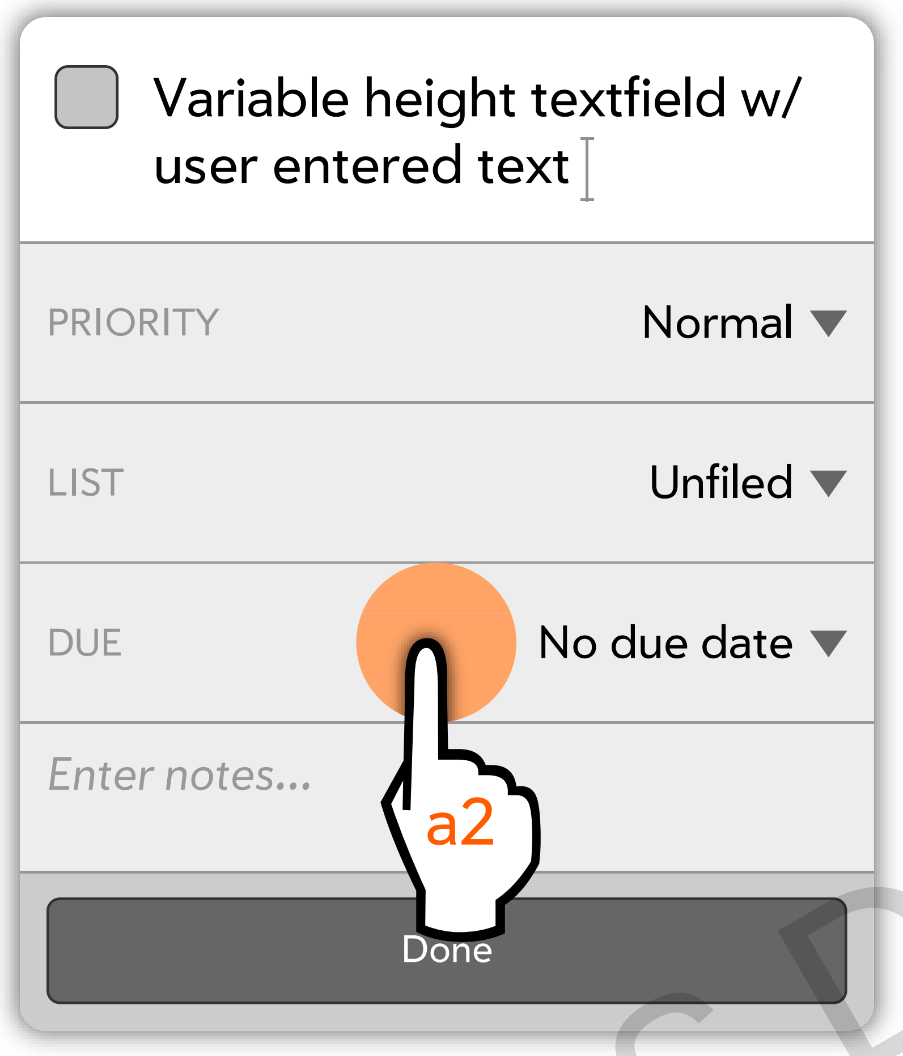
INTERACTIVE POP-UPS: FORM FILLING AND TEXT INPUT

A common usage for interactive pop-ups is to enable users to fill out form elements or to input text. In the example shown below, an interactive pop-up is shown containing various form elements. Selecting these form elements can navigate users to secondary views to fill out further information, but navigation is based from the initial interactive pop-up containing the form elements.

If the user is required to enter text, the Virtual Keyboard is invoked as usual to enable text input. The Virtual Keyboard should not cover the Interactive Pop-Up when it is displayed. The Interactive Pop-up should be repositioned and in some cases may need to be re-sized to ensure that the Virtual Keyboard and Interactive Pop-up do not overlap.

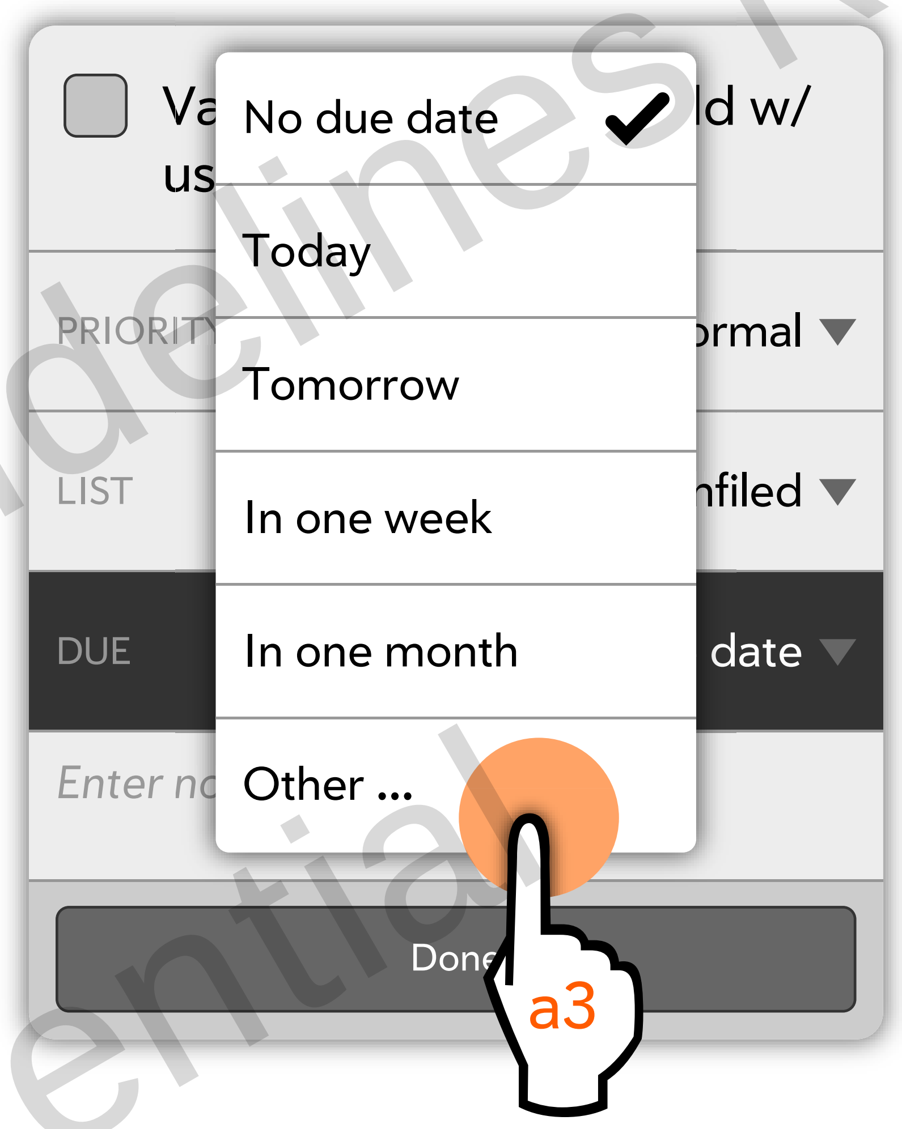
Interactive pop-ups can also be used for longer interaction sequences that guide the user through several views within a task flow. See the *Task flow* page within the *Navigation Patterns* section for further details.

a1. Pop-up w/ form elements



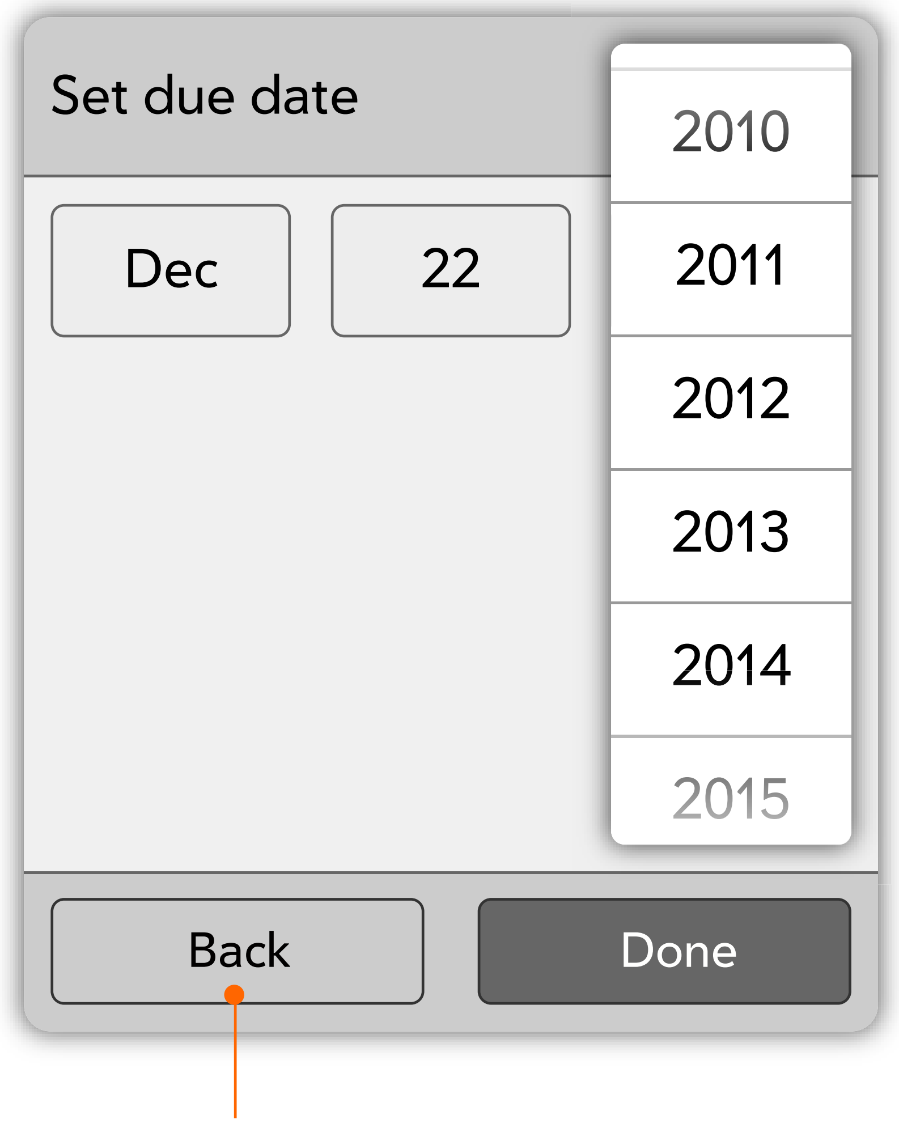
- The initial view in the interactive pop-up can be freely constructed from form elements
- The view should contain a button to confirm and close the pop-up (‘Done’)

a2. List menu on top of pop-up



- It is possible to use components that create additional layers above the pop-up (e.g., List selectors, integer pickers)
- When additional elements are shown above the pop-up, the user may tap anywhere outside the element to dismiss it

a3. Date picker with list menu



Place ‘Back’ navigation buttons on the left

- If the user drills-down to another level within the pop-up, a button should be shown in the button area to allow the user to either back step to the previous view (‘Back’). The Back button should be placed on the left of the button area.
- A button should also be shown to allow the user to confirm and close the pop-up (‘Done’)

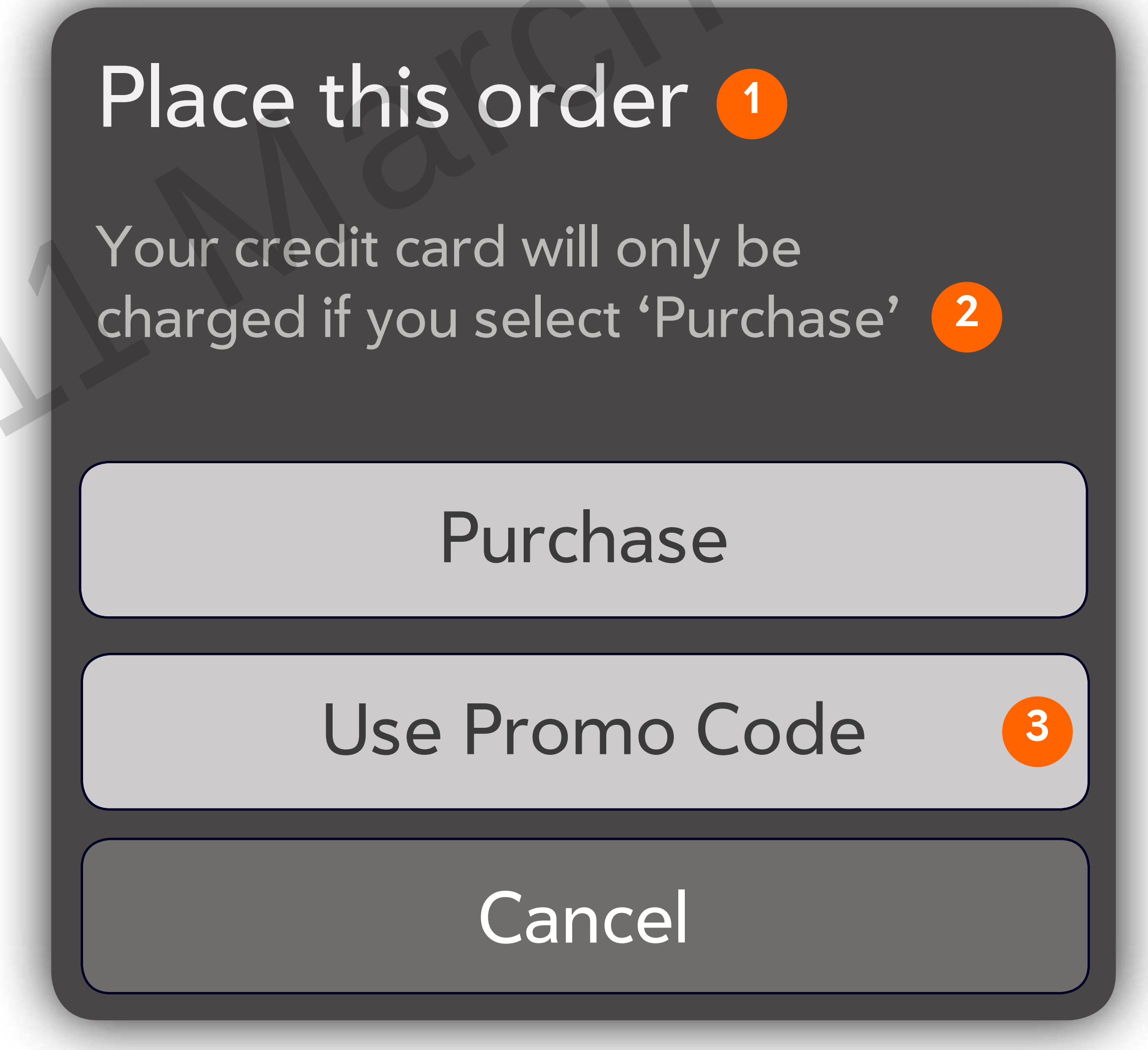
DIALOGS

What they are

Like interactive pop-ups, dialogs are also modal UI components. When a dialog is displayed, the user cannot interact with the scene below.

Unlike interactive pop-ups, however, dialogs do not generally contain an interactive content area. Dialogs consist of a title, some body text and buttons. Navigation between different levels of content is not possible with dialogs. The title is shown using large text and the body text is shown using smaller text.

As with interactive pop-ups, the buttons may appear side-by-side or may be stacked vertically, and buttons handling navigation should be shown side-by-side, whereas action buttons should be stacked vertically.


Place this order 1
Your credit card will only be charged if you select 'Purchase' 2
Purchase
Use Promo Code 3
Cancel

1 Title

2 Body text

3 Buttons

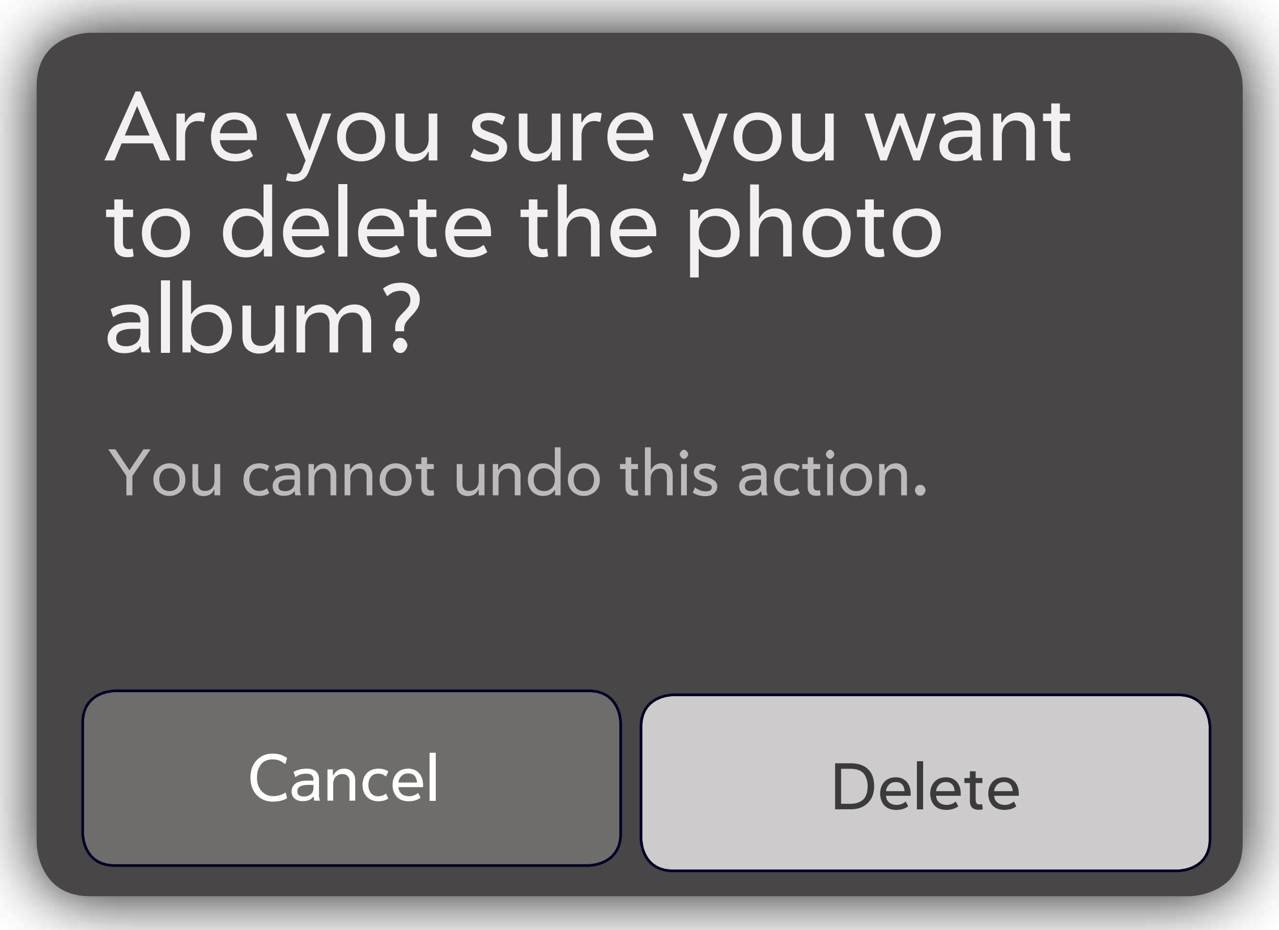
When to use

Use dialogs to prompt the user to perform simple and quick interactions, such as confirming an action, or to inform the user. They should be used when your application requires confirmation or disambiguation from the user. For more complex and lengthy interaction sequences, or to display interactive content, consider using an interactive pop-up.

As dialog components are modal elements, they are disruptive to the user's flow, and so should only be used when it is important to interrupt the user's attention. To notify the user in a more discrete fashion, consider using a Notification.

DIALOGS: EXAMPLES

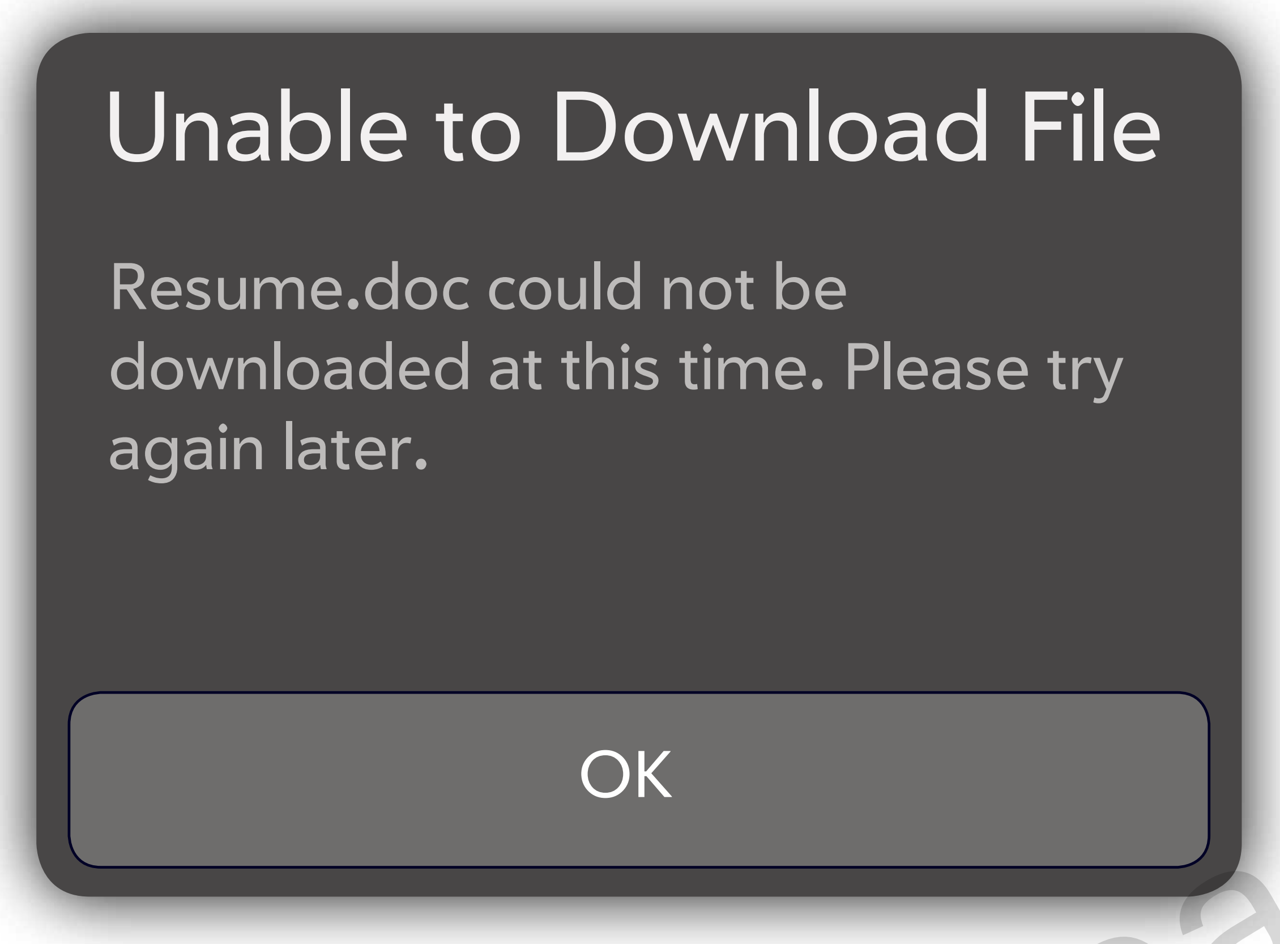
CONFIRMATION DIALOG


Are you sure you want to delete the photo album?

You cannot undo this action.

- Confirmation dialogs enable users to confirm an action. Confirmation dialogs are typically shown only for irreversible actions.
- Confirmation dialogs should contain two buttons - a cancel button shown on the left, and a button to confirm the action on the right
- The button confirming the action is shown on the right of the button group

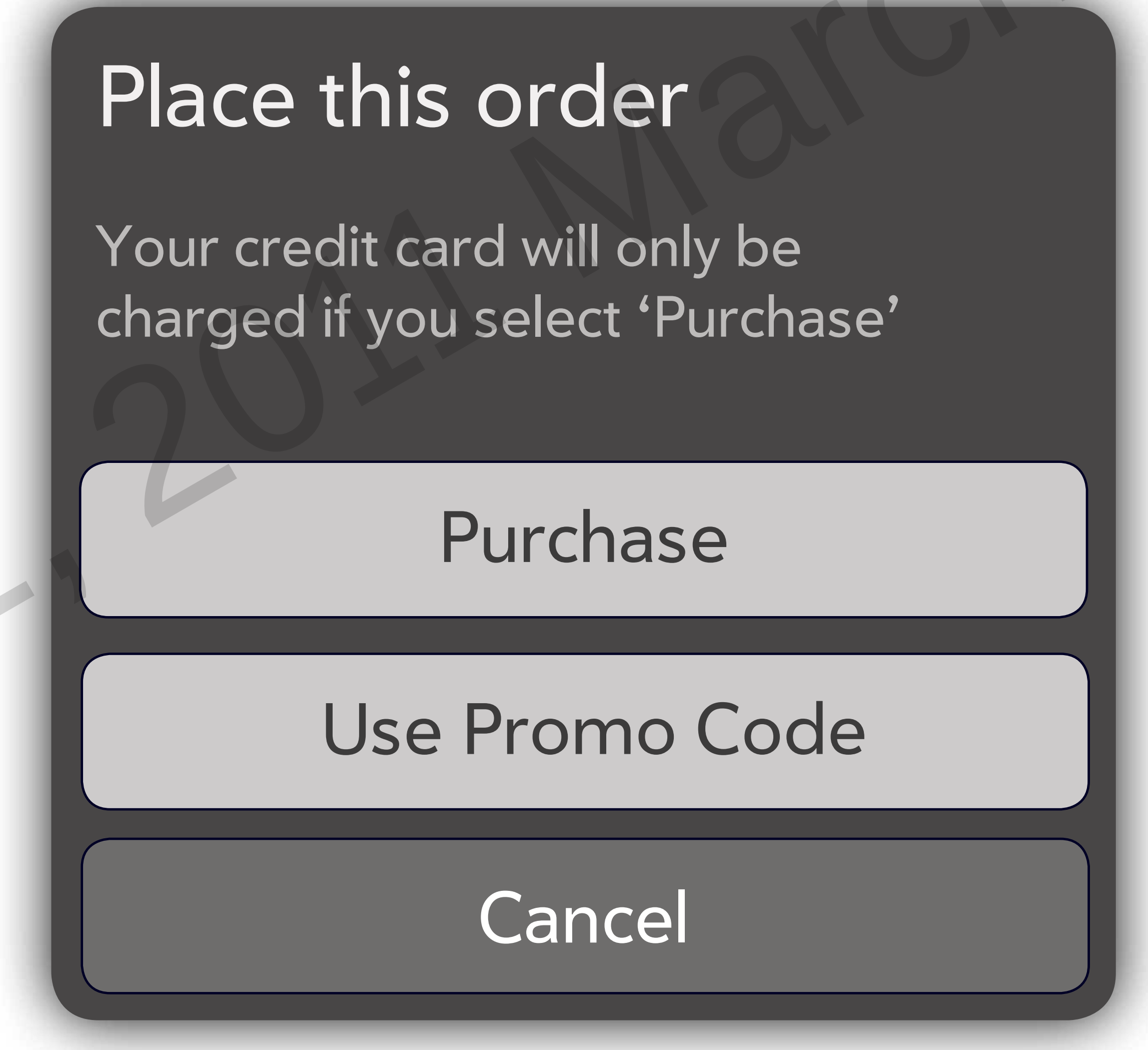
INFORMATIVE DIALOG


Unable to Download File

Resume.doc could not be downloaded at this time. Please try again later.

- Informative dialogs are used to inform the user of important or time-critical information
- They contain a title and body text to inform the user and a single button to dismiss the dialog

ACTION DIALOG


Place this order

Your credit card will only be charged if you select ‘Purchase’

- Action dialogs are used to present a list of actions to the user within a task flow
- These actions should be presented using vertically stacked buttons
- A Cancel button must also be shown to allow the user to dismiss the dialog

NOTIFICATIONS

What they are

Notifications are subtle, non-modal elements that are used to inform the user of events. When a notification is received it does not prevent the user from interacting with the rest of the interface. An application can publish a notification to the user if the application is running in the background or even if the application's card has been dismissed altogether by the user from the card view. Three types of notification can be published by an application:

- Banner only - when received, a banner containing text animates into view in the view's status bar. The banner animates out of view after a time-out.
- Banner notification - as per a Banner only notification, but after the banner text has been displayed, an icon from the application remains in the Icon Summary area in the status bar.
- Notification icon - when received, an icon appears in the Icon Summary area in the view's status bar without showing any banner

If icons are displayed in the Icon Summary area, a user can tap on the area at any time to launch a Dashboard component, which shows a listing of all active notifications received on the device. When the dashboard is opened, the user can either dismiss notifications by swiping them away, or tap on a notification to trigger an action associated with the notification.

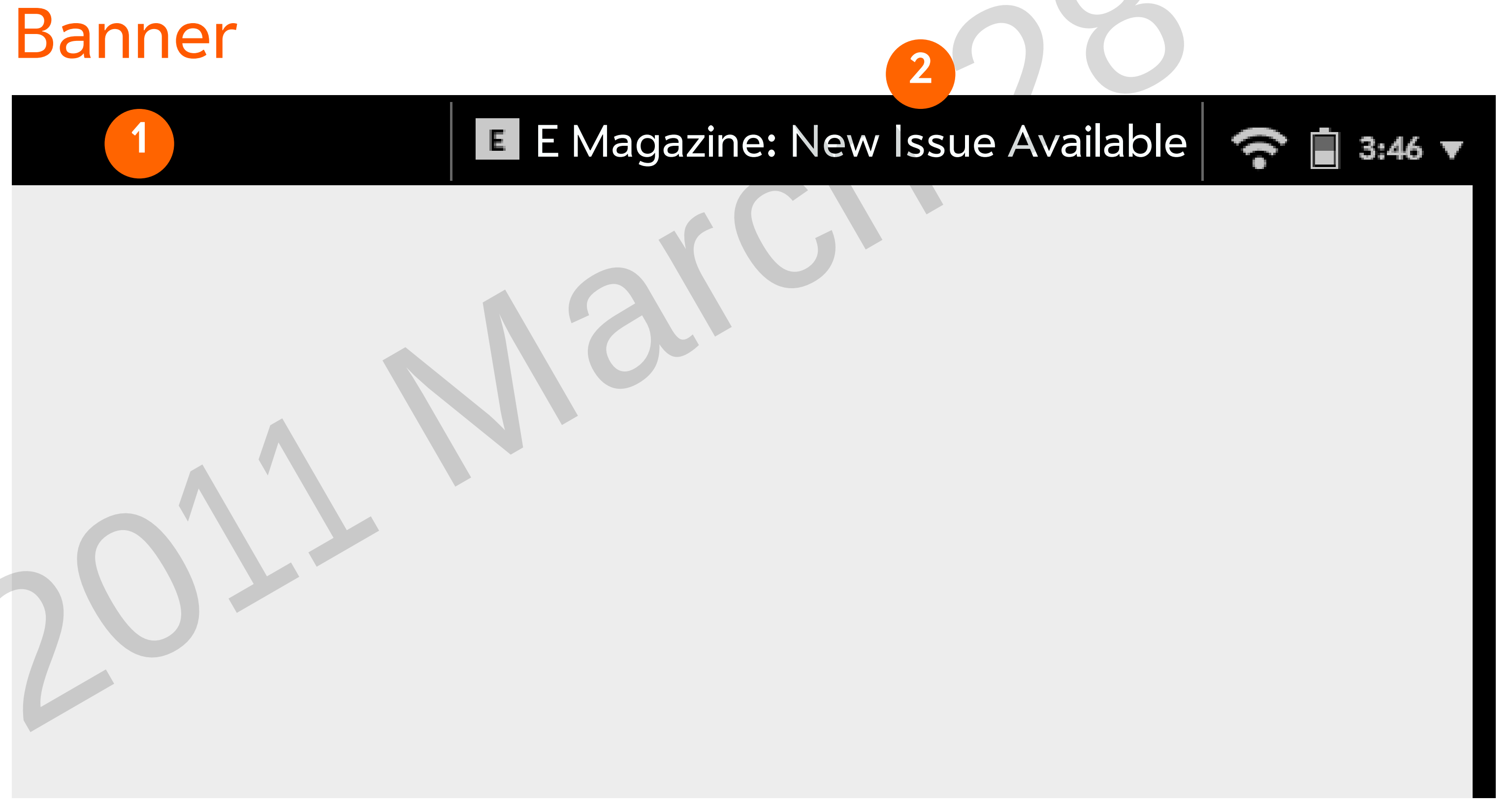
When a notification is received, the application can specify whether a sound is played by the device, or whether the device vibra is activated, or both. The application can customize the sound that is played by the device (functionality to be confirmed).

If several notifications are received by the same application, they are shown grouped together in the dashboard as one item. An indicator shows the number of notification received from the application.

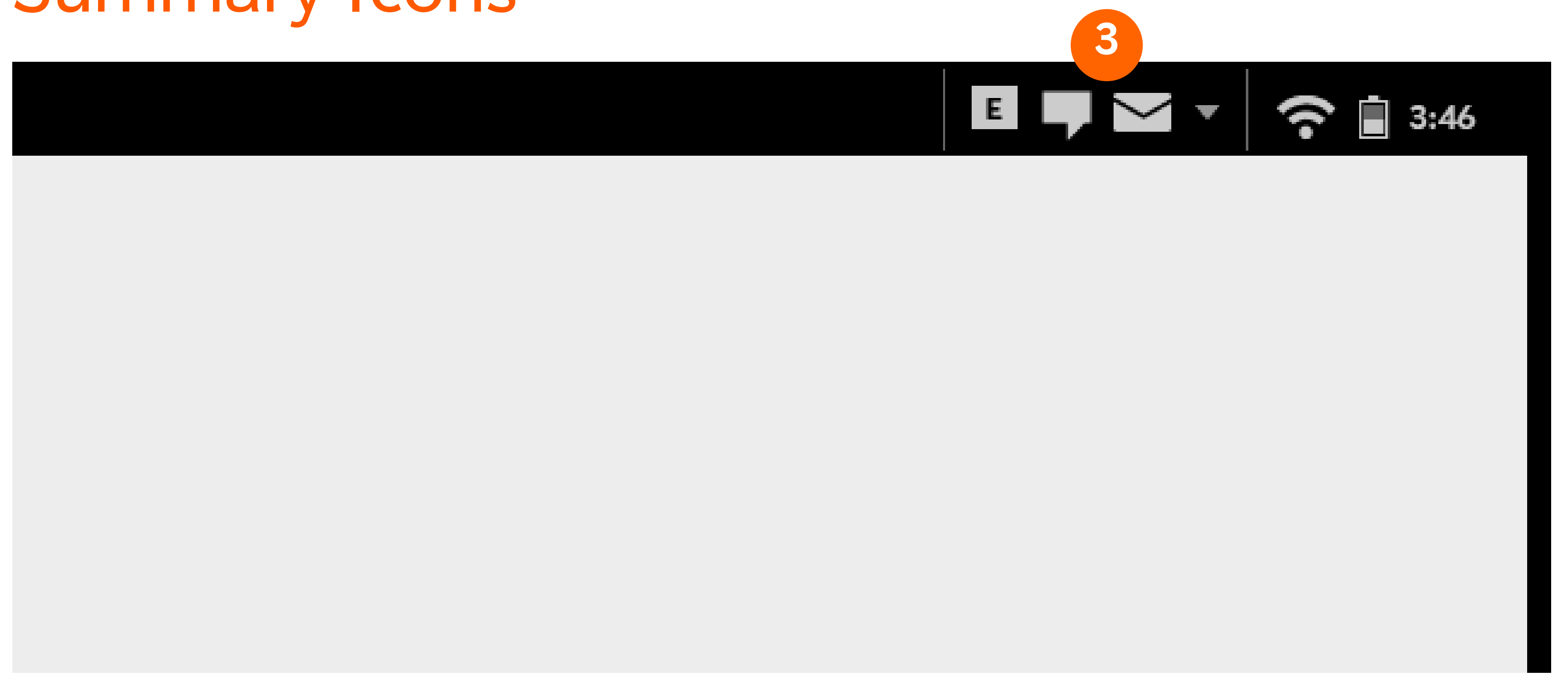
When to use

Use notifications to inform the user of an event or multiple events in a discrete and non-disruptive way.

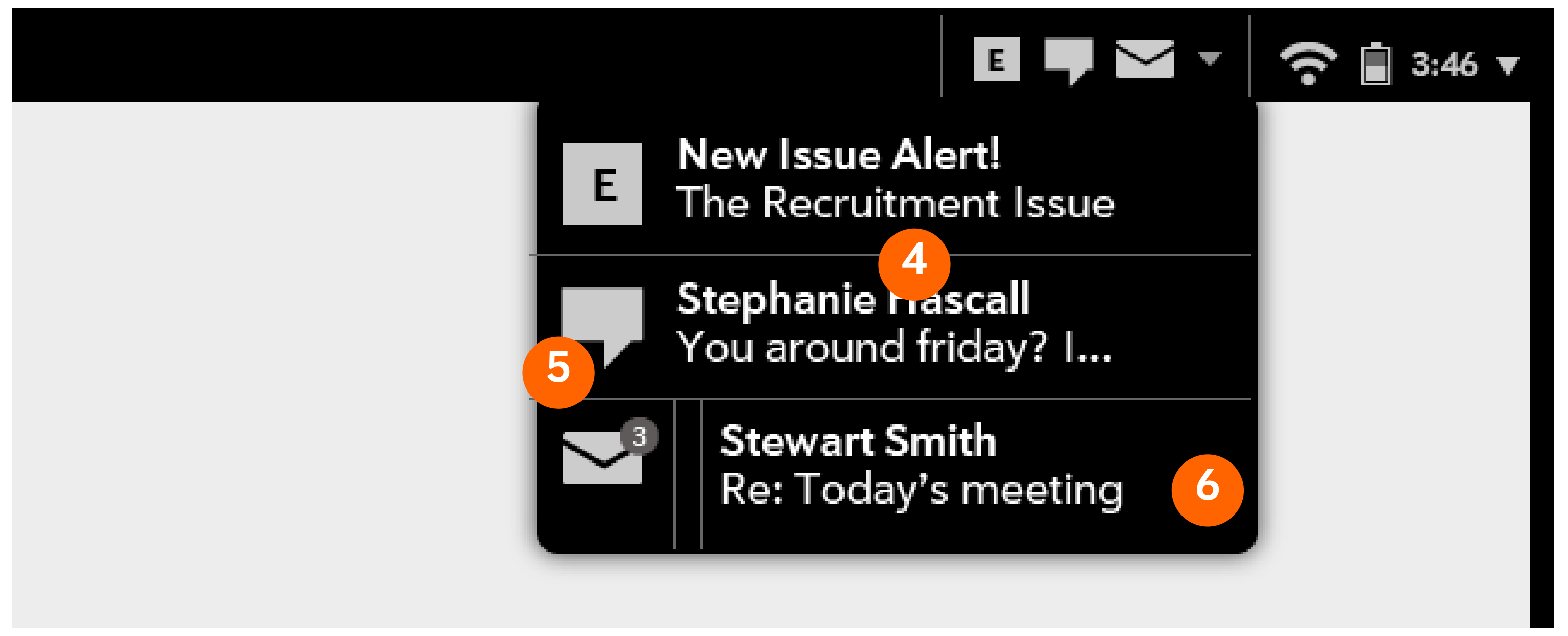
Banner



Summary Icons



Dashboard



New Issue Alert!
The Recruitment Issue

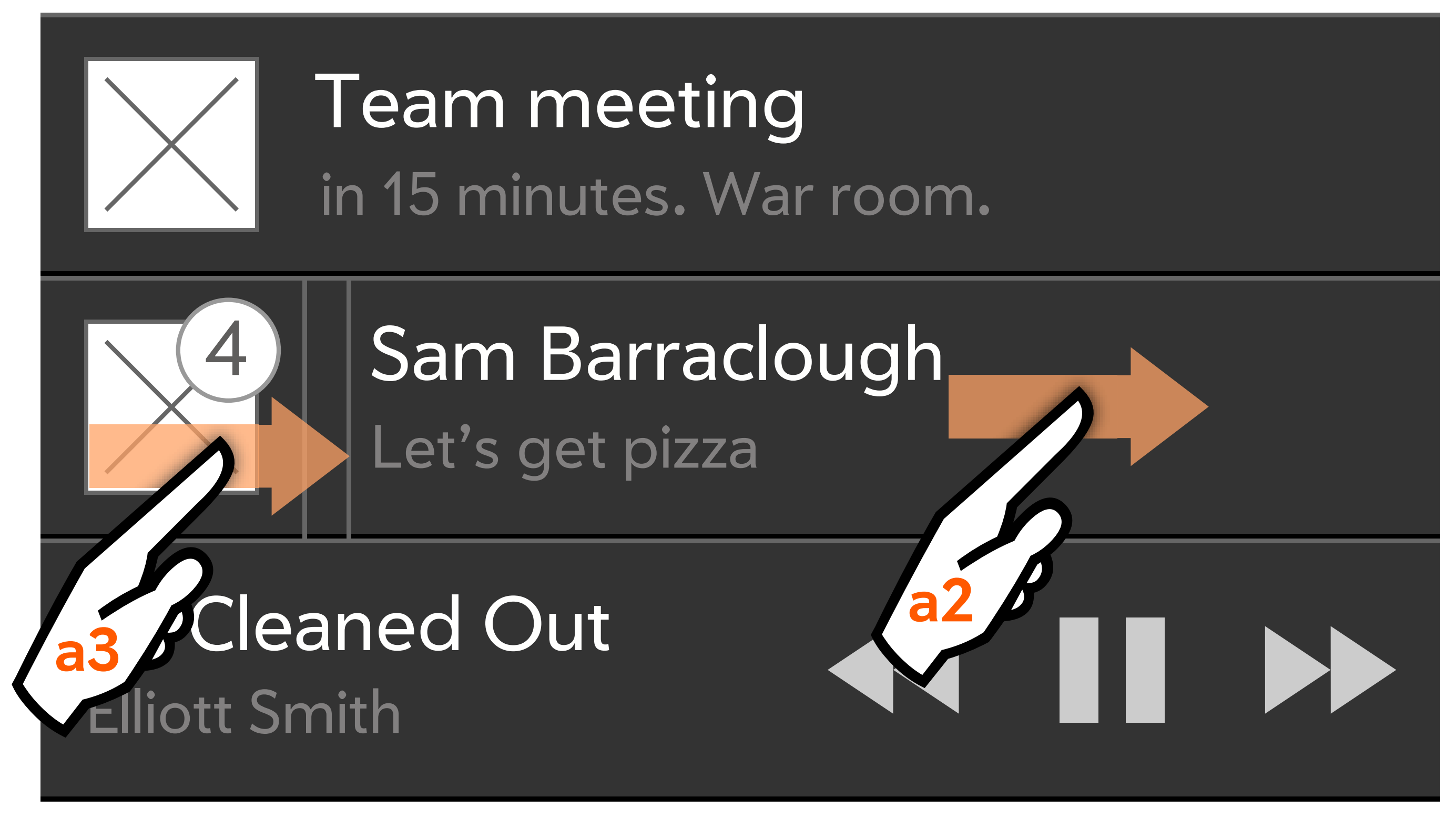
Stephanie Hascall
You around friday? I...

Stewart Smith
Re: Today's meeting

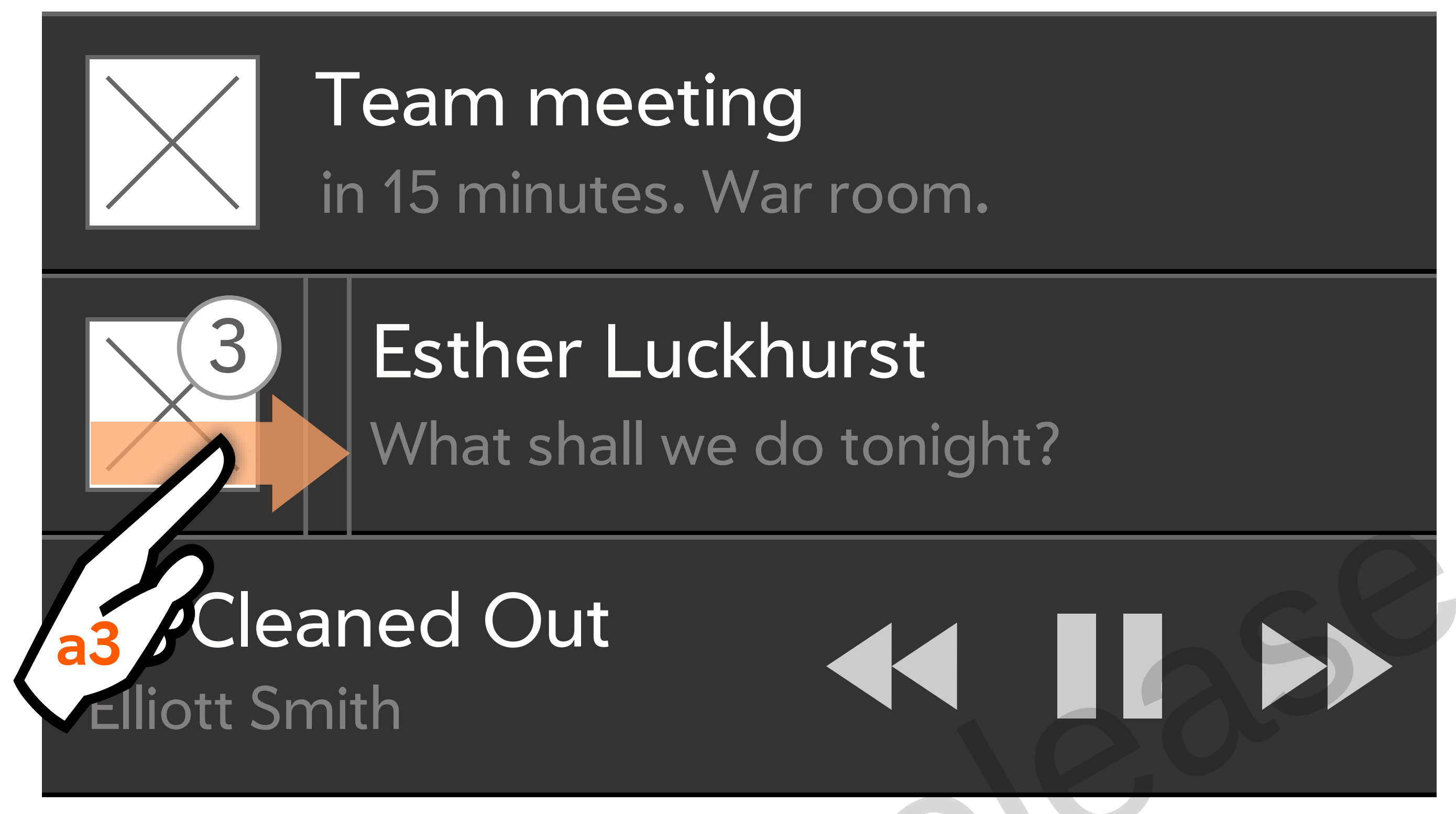
1	Status Bar	4	Dashboard
2	Banner Text	5	Application icon
3	Icon Summary Area	6	Grouped notification

GROUPED NOTIFICATIONS

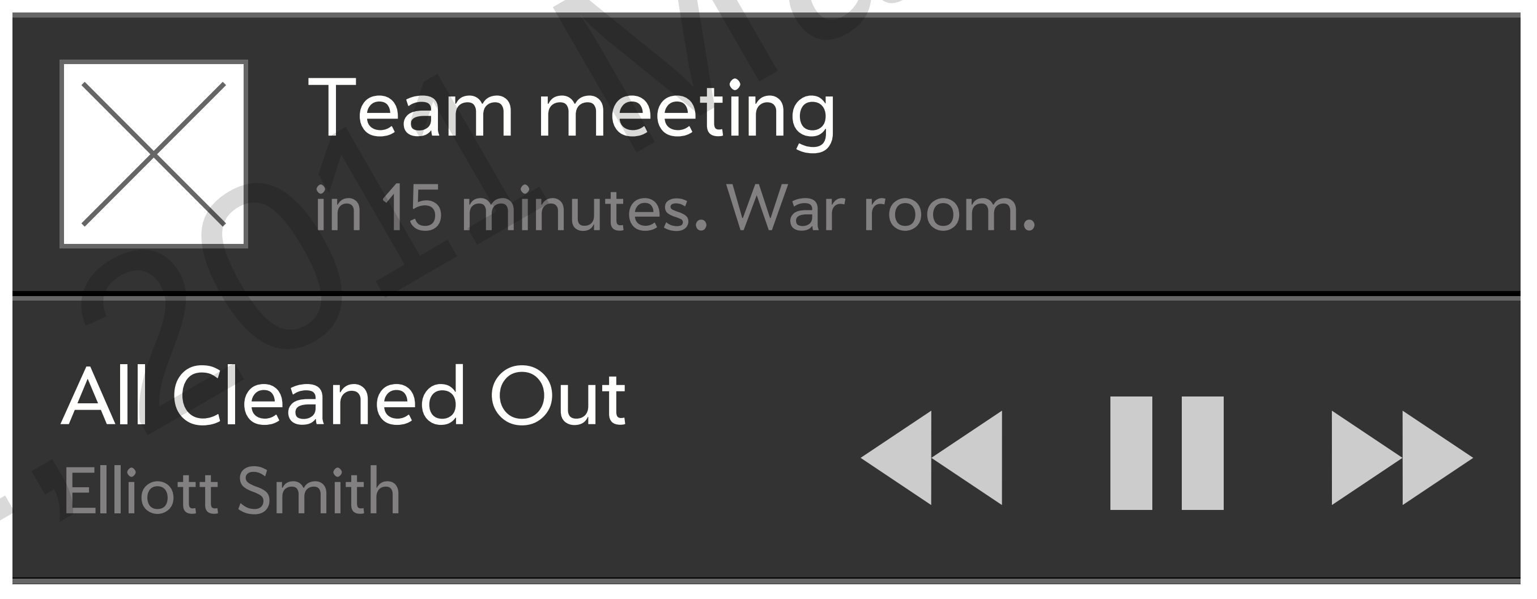
a1. Group of 4 notifications



a2. Group of 3 notifications



a3. Grouped notification dismissed



- The user can dismiss individual notifications by swiping them away to the right
- Grouped notifications can be dismissed by swiping away the application icon

ALERTS

What they are

Similar to Notifications, Alerts are also non-modal elements used to inform users of events. Alerts, however, are more prominent and noticeable and should be used when it necessary to get the user's attention. Alerts are displayed above application content until actively dismissed by the user or until they time-out. The time-out schedule can be configured by the application that triggers the alert.

As with Notifications, an application can publish an alert to the user if the application is running in the background or even if the application's card has been dismissed altogether by the user from the card view.

The design of the alert is at the discretion of the calling application - however, please note the following guidelines:

• include an application icon to inform the user which application is publishing the alert
• include an action button that allows the user to dismiss the notification
• provide other action buttons that allow the user to perform other actions associated with the alert event

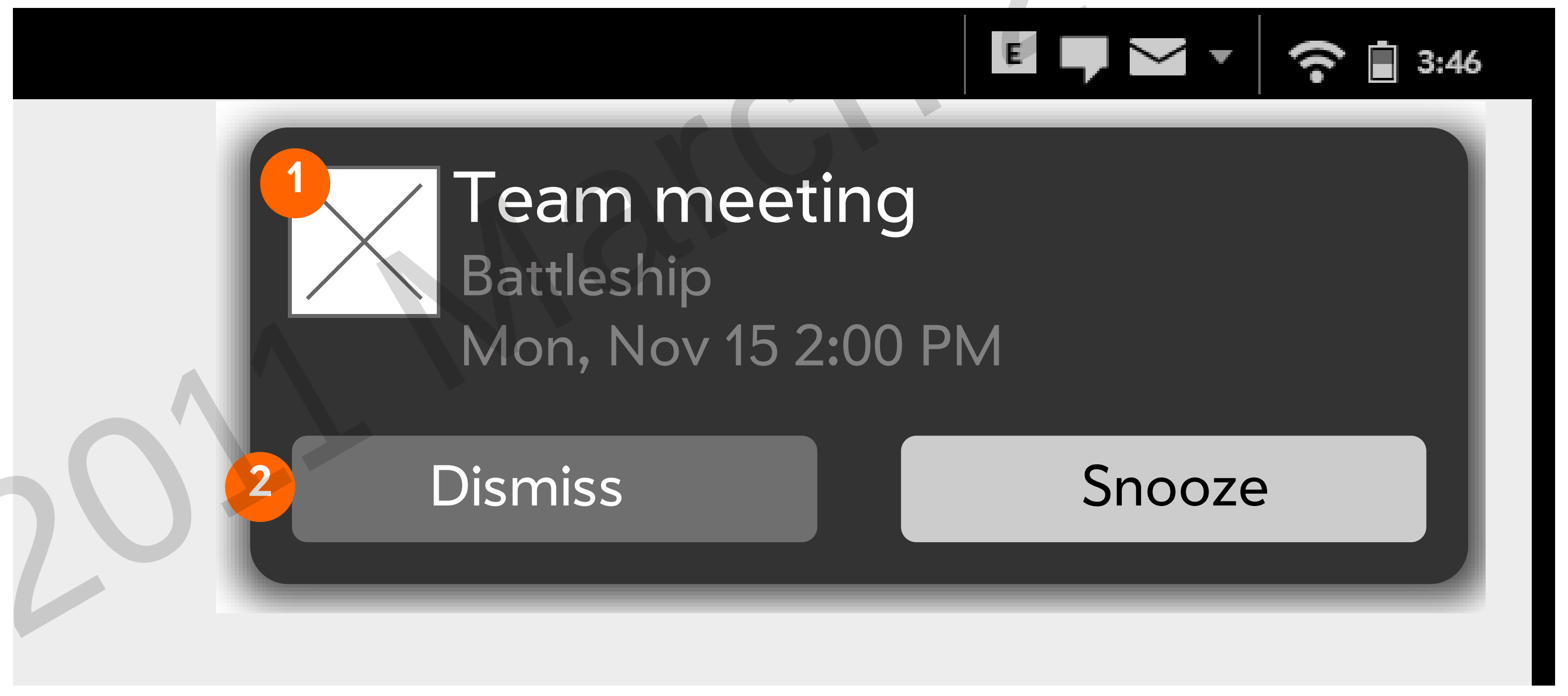
As with Notifications, when a notification is received, the application can specify whether a sound is played by the device, or whether the device vibra is activated, or both. The application can customize the sound that is played by the device.

Unlike Notifications, only one alert can be shown at any one time. If more than one alert is active at one time, the most recent alert is shown and other alerts are shown as the more recent alerts are dismissed.

When to use

Use Alerts to inform the user of an important or time-critical event. Use alerts only if it the event is important enough to disturb the user.

Alerts



1 Application icon

2 Action buttons

APPLICATION MENU

What it is

The application menu is an application-specific menu that is available from any view on the device. The application menu consists of a title area and a menu that contains the menu commands. The menu is launched by tapping or swiping down on the title area which is displayed in the top-left area of the application chrome. The application should provide an application name that is displayed in the title area. The application menu is dismissed by tapping outside the menu area.

The menu is not dynamic and the same menu items must appear any time the menu is launched from within the application. If some menu items are not applicable in the context from which the application is launched, they should be grayed out.

The application menu also contains Global Commands - these are common commands that are available across different applications. The Global commands are shown beneath the other commands and are separated from them using a menu divider. The Global commands are:

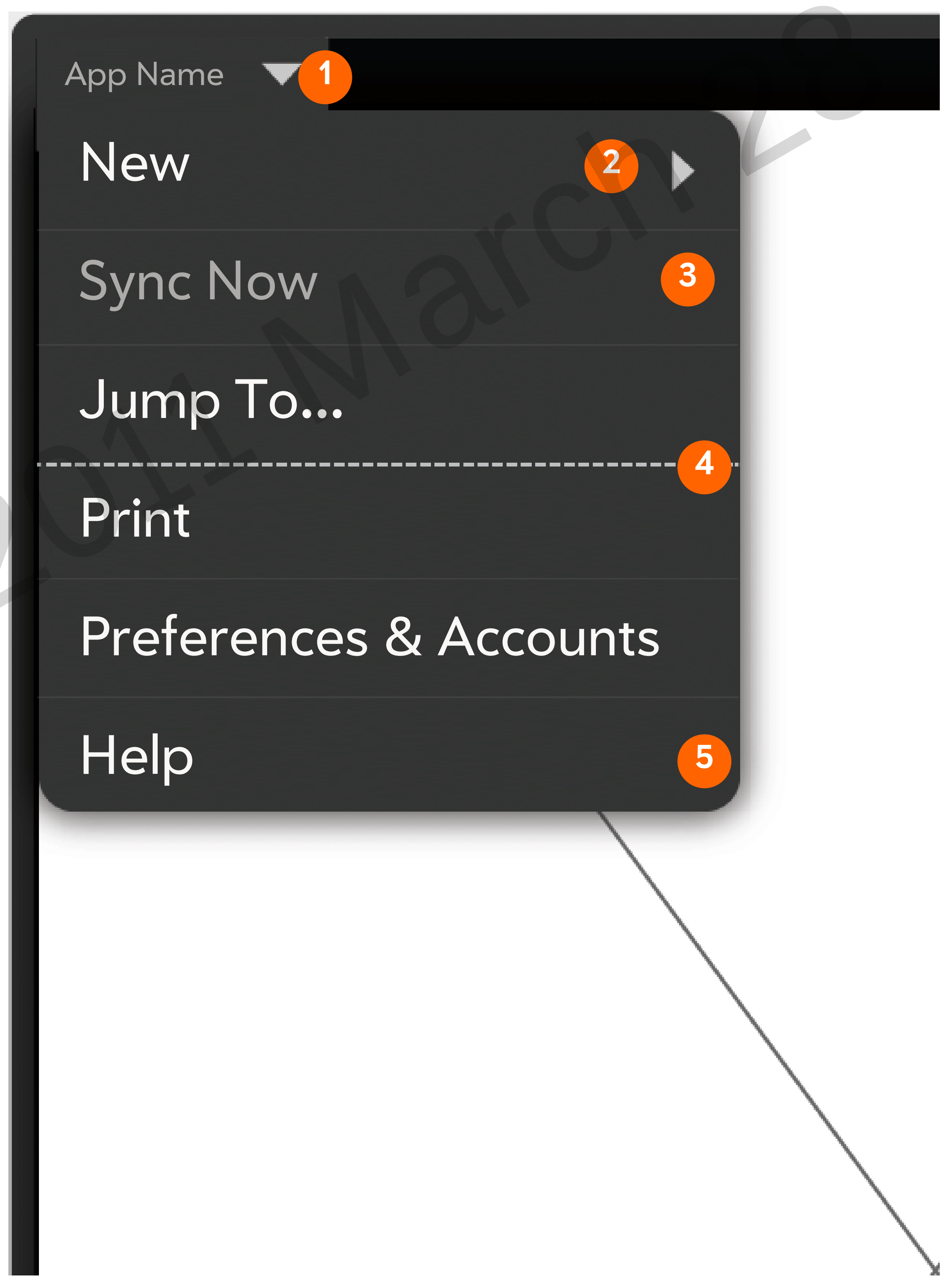
- **Print** - optional and can be switched on by the application if relevant.
- **Preferences & Accounts** (or 'Preferences' if the application does not use accounts) - launches a view for managing application preferences, or accounts, or both.
- **Help** - launches a view providing user guidance for the current view.

An application menu can contain sub-menus. Menu commands that launch a sub-menu are shown in the menu with a right-pointing arrow to indicate that tapping on the item will launch a sub-menu.

When to use

Use the application menu to display general commands that are specific to the application or the application view as a whole, rather than for commands that are specific to a particular object in the view.

In general, use the application menu for actions that are only used occasionally within the life cycle of the application. More frequently-used commands should be visible within the view.



1 Title area
2 Sub-menu command
3 Disabled command
4 Menu divider
5 Global commands

DESIGNING YOUR APPLICATION

This section describes the key interaction patterns and design guidelines that will help you design your application on webOS.

THE DNA OF WEBOS

A focus on people
...rather than the technology

Task-oriented interactions
...enable you to seamlessly use other applications without losing context

Simple, clear design
...with a focus on enabling important features for users

Physical metaphors
....with intuitive, natural interactions that just make sense

Unobtrusive notifications
....that don't take away users' focus

Unified data
...makes it easier to find information with minimal navigation

DESIGN PRINCIPLES

Everything is obvious

- Visual cues support the intended interactions.
- You should not need a tutorial to use our system or any applications.
- Information is displayed in an organized manner.
- Don’t rely on a physical back gesture area.

Simple, natural gestures just work

- Some examples: Turning pages, side-swiping through single-page emails in Portrait. Scrubbing up/down through lists to quick scroll.

Advanced interactions are supported when it makes sense

- As secondary, super-cool ways to do things. Never the primary or only way.

Important actions are within immediate reach

- They’re immediately visible, and easy to tap them while holding the device comfortably.
- Other actions can be found easily
- Secondary actions can be found. They might require a little exploration or additional reach - and that’s okay.
- Less important or destructive functions are harder to reach or may require additional steps to access

Applications behave safely and reliably

- Applications are useful, enjoyable, and they don’t crash.
- When applications permit you to do something slightly destructive (like deleting something), they warn you before they allow you to proceed.

Content, functionality and interactions should be appropriate to the form factors of the device

- How do you hold the device? What types of activities are important for the form factors of the device?
- How large is the screen? What density of information and controls are reasonable for the screen real-estate?

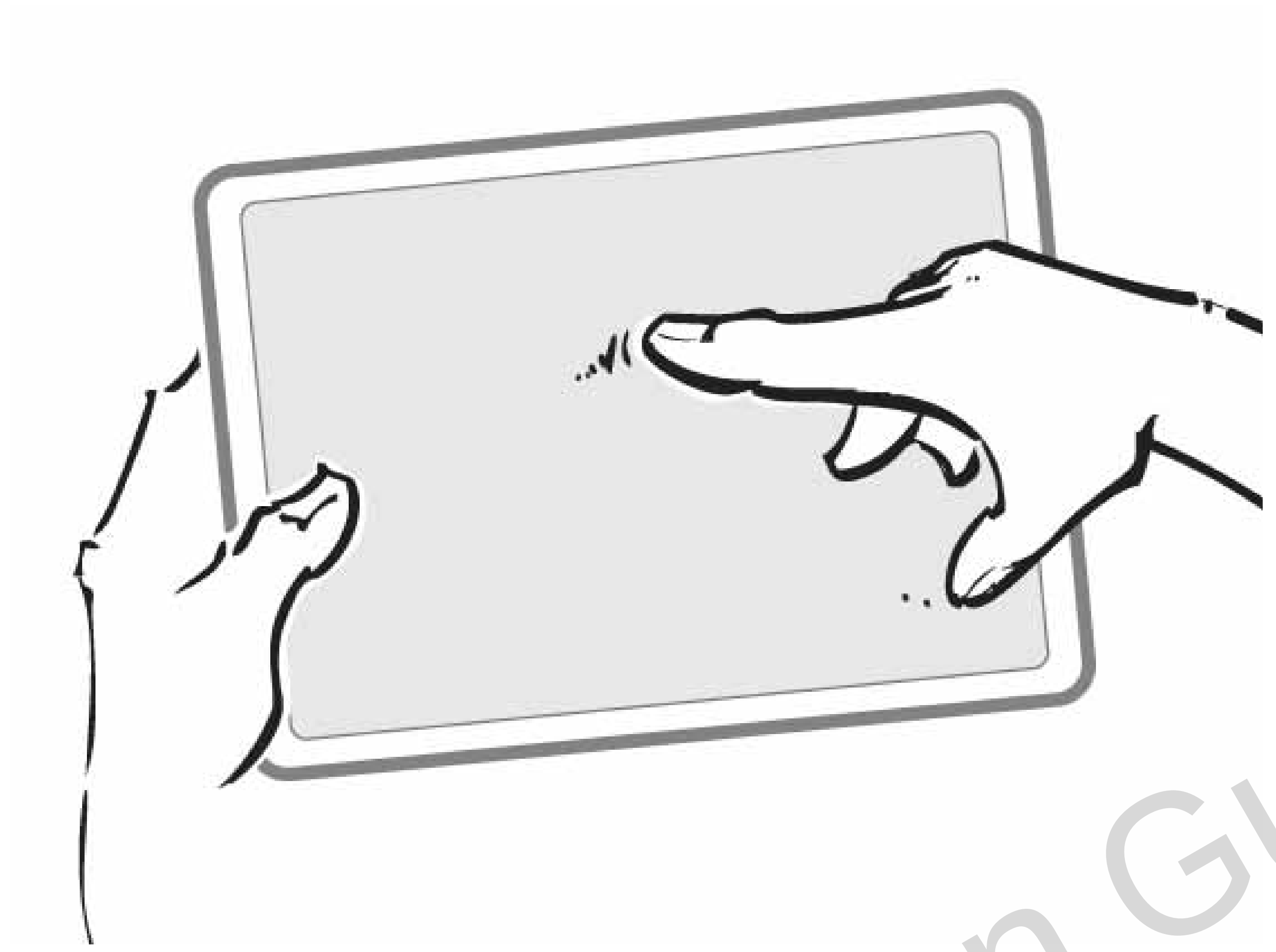
KEY DESIGN DIFFERENCES BETWEEN PHONE AND TABLET INTERACTIONS

PHONES	TABLETS
Single, focused area for content.	Potential for **multiple areas** of content where appropriate.
Limited on-screen actions, exposing only highest priority functions.	Ability to **expose a greater number of on-screen actions** that are high priority.
Mainly **list-based navigation** optimized for the small screen.	Potential for more fun, intuitive navigation with **greater emphasis on visual objects.**

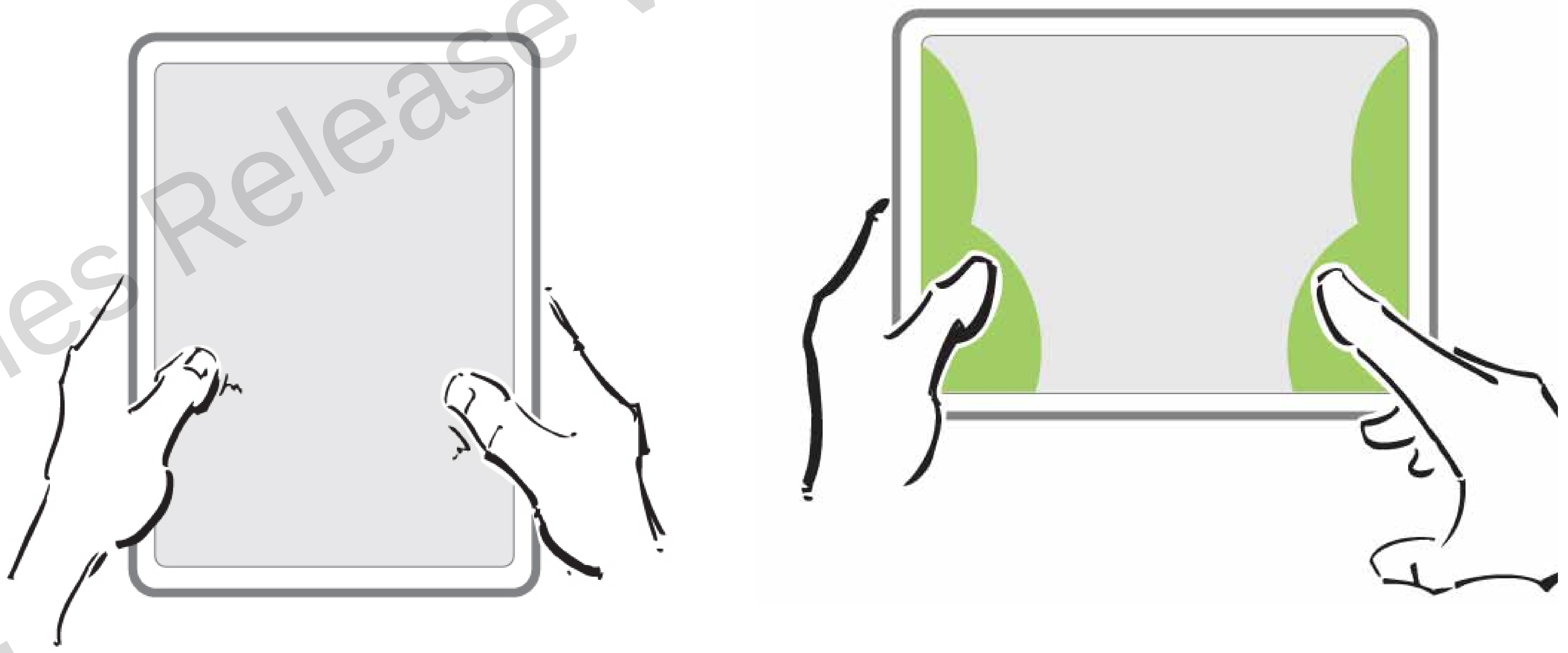
Note for current webOS developers:
The physical gesture area present on current webOS phones will be omitted from the TouchPad and future phones. Actions previously tied to back and forward gestures should be accomplished via direct manipulation (e.g. sliding UI panes to show/hide them) or explicit buttons (e.g. "OK", "Cancel", "Done")

LAYING OUT CONTENT AND CONTROLS

Consider how people will hold the TouchPad



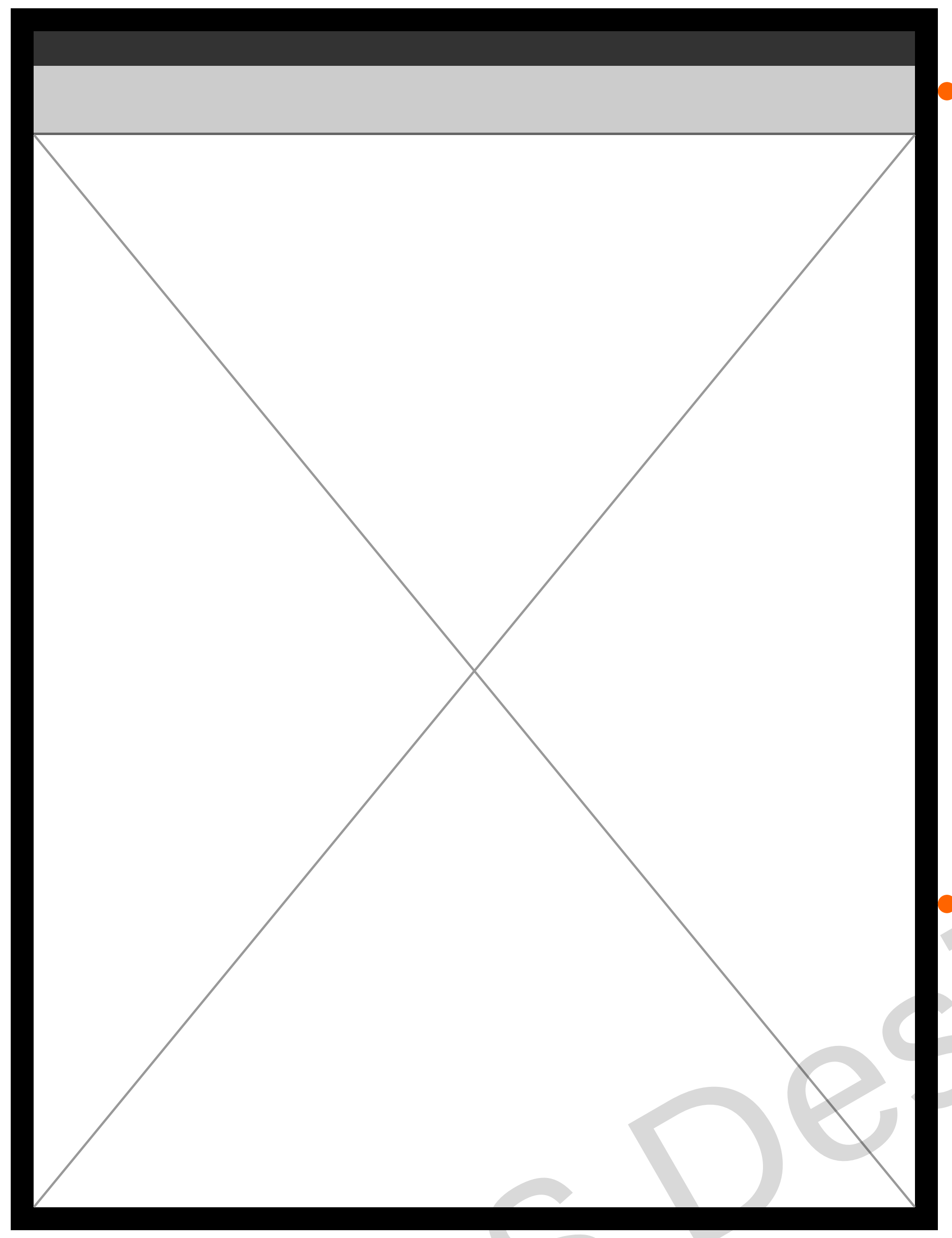
- Tablet has no conventional grip
- Any area is potentially a touch target



- Consider adding controls to the sides as these are easily thumb-able areas
- Avoid placing destructive controls on the sides as these areas are also easy to tap by mistake when gripping the device

Create logical groupings of navigation and action controls

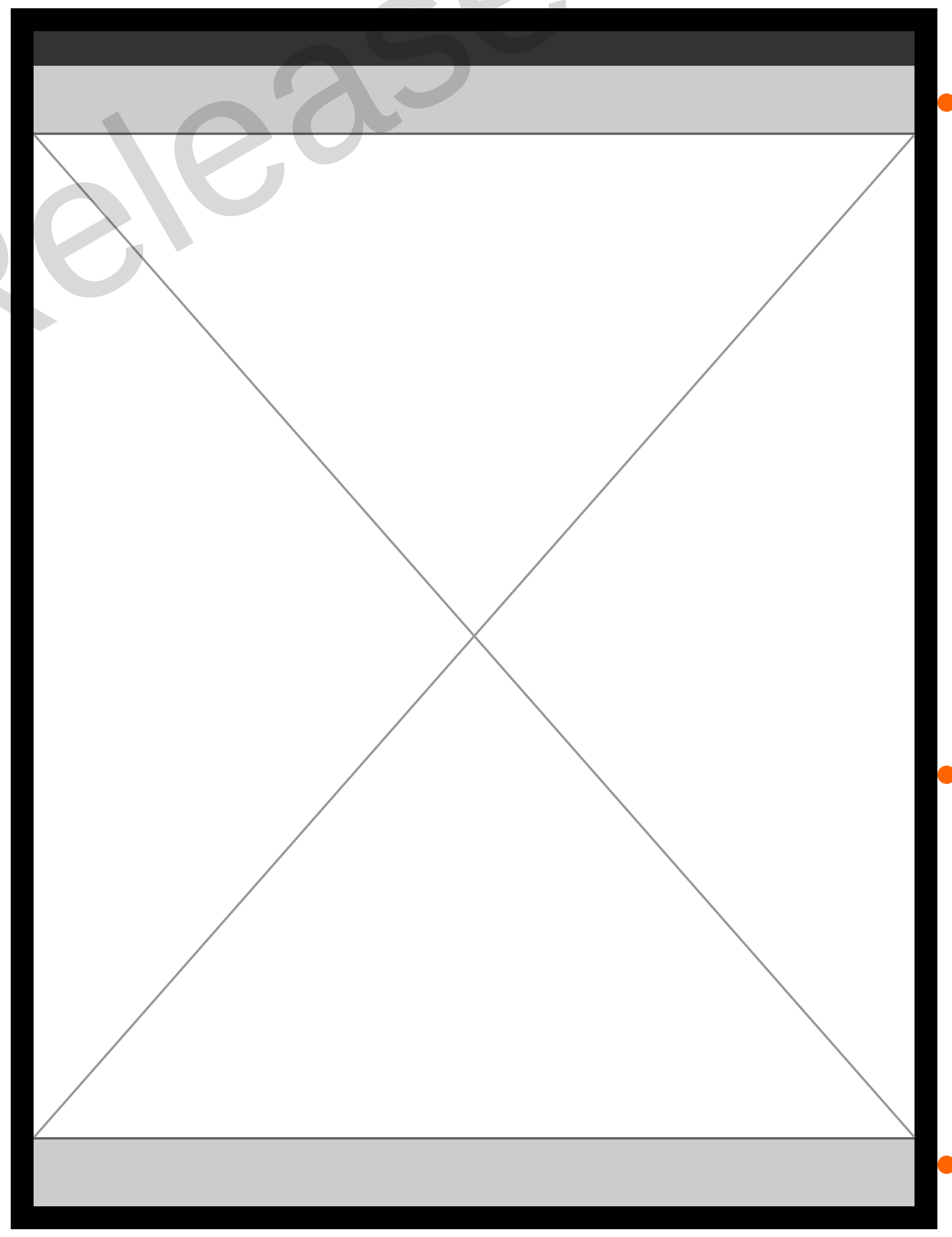
PANE WITH HEADER ONLY



HEADER AREA:
Use for:
- View title
- Search
- View options
- Navigation
- Actions and controls

CONTENT

PANE WITH HEADER AND FOOTER ONLY



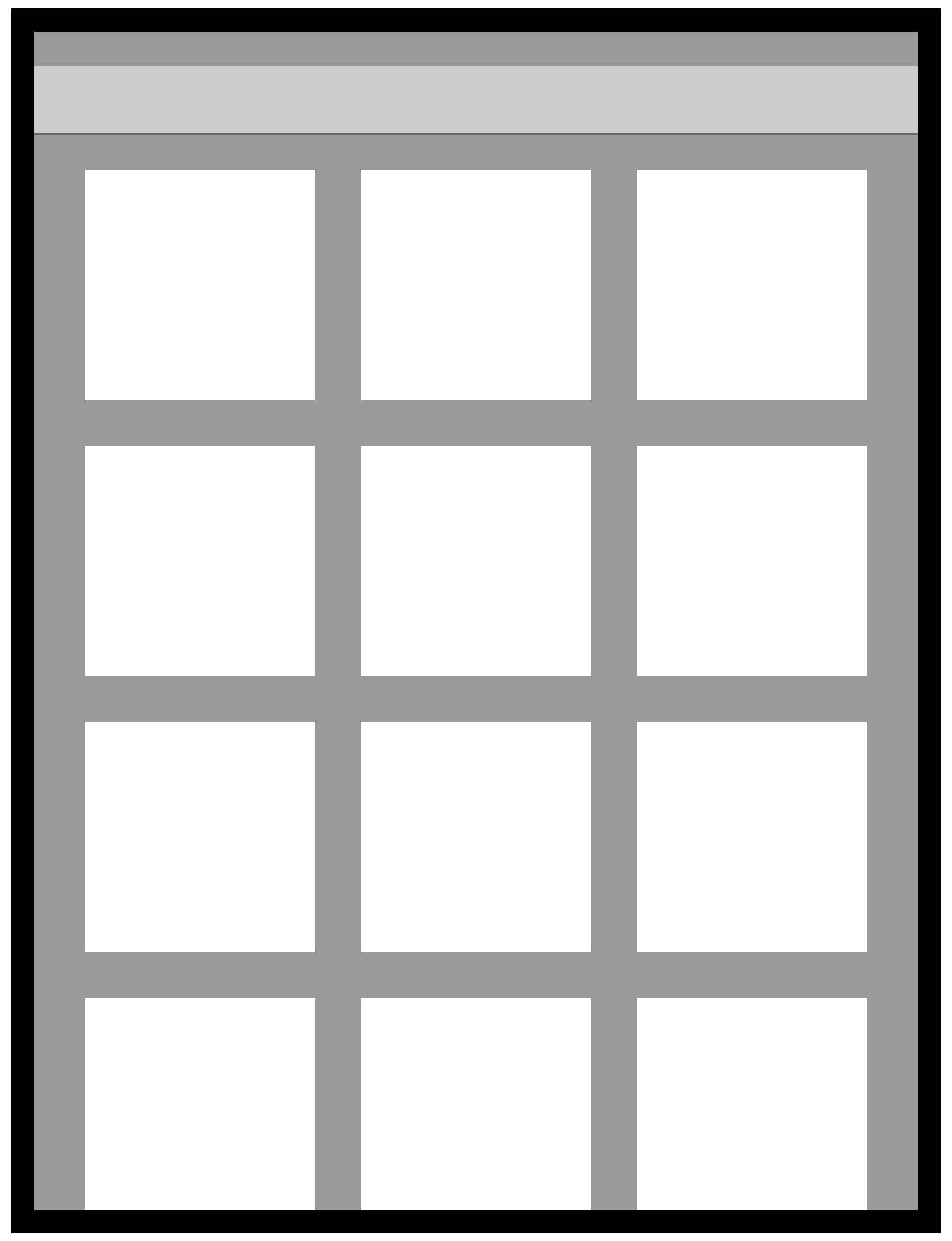
HEADER AREA:
Use for:
- View title
- Search
- View options
- Navigation

CONTENT

FOOTER AREA
Use for actions and controls

LAYOUTS: SINGLE PANE AND SPLIT VIEW

SINGLE PANE



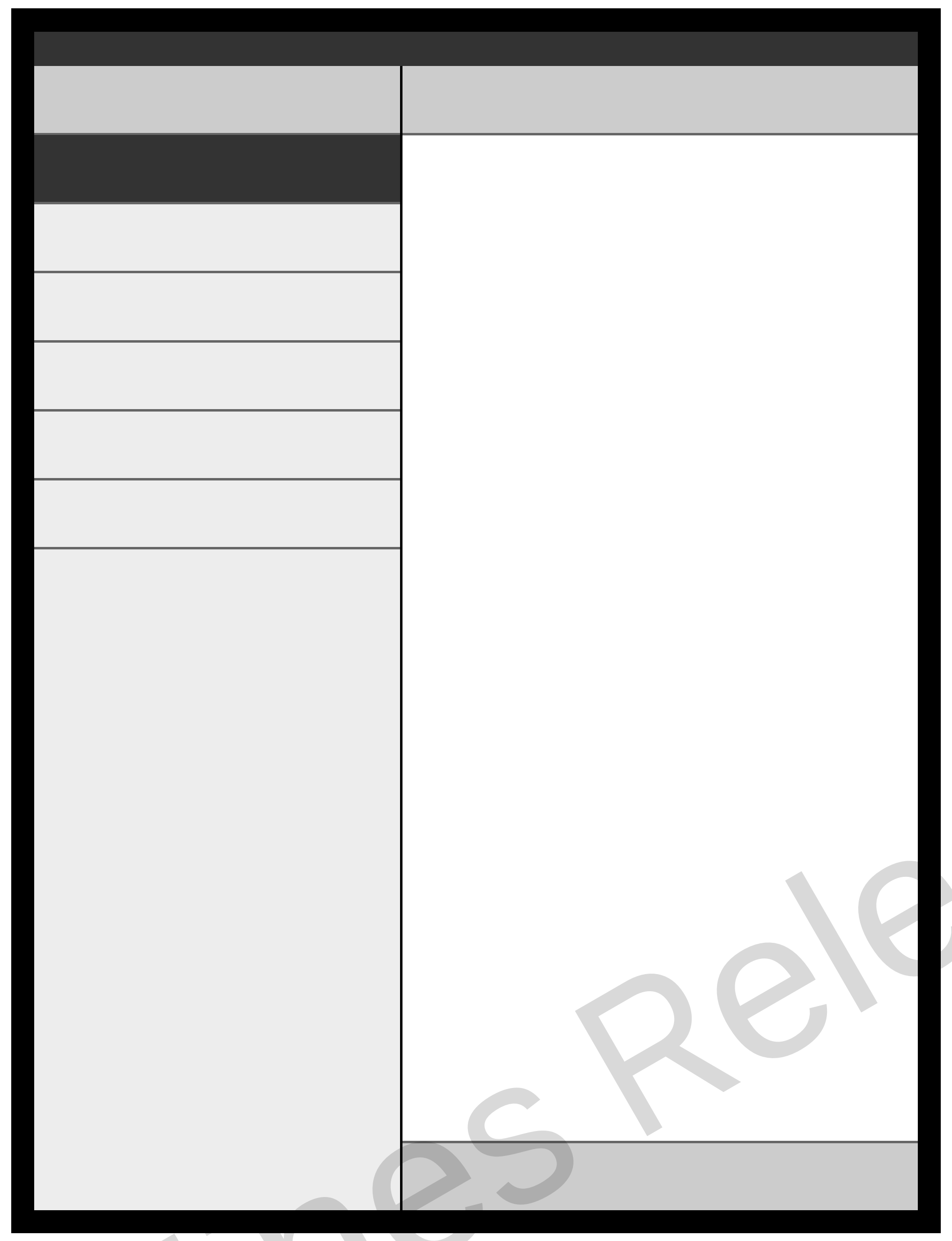
DESCRIPTION

- This is the simplest layout available
- Use the header area for:
  - Navigation functions, such as tabs for switching between content
  - A search field for searching through application content
  - Viewing options - e.g., use buttons to select what content is displayed in the content area, such as grid vs. list, or day/week/month
  - Labels to describe the content in the content area
  - Place action buttons on the right side of the header area if a footer area is not employed in this layout
- The content area is typically used to present visual content that makes full use of the larger canvas, such as image grids or memo notes. For displaying list-based information, use a different layout

APP EXAMPLES

- Memos
- Calendar

SPLIT VIEW (FULL WIDTH PANES)



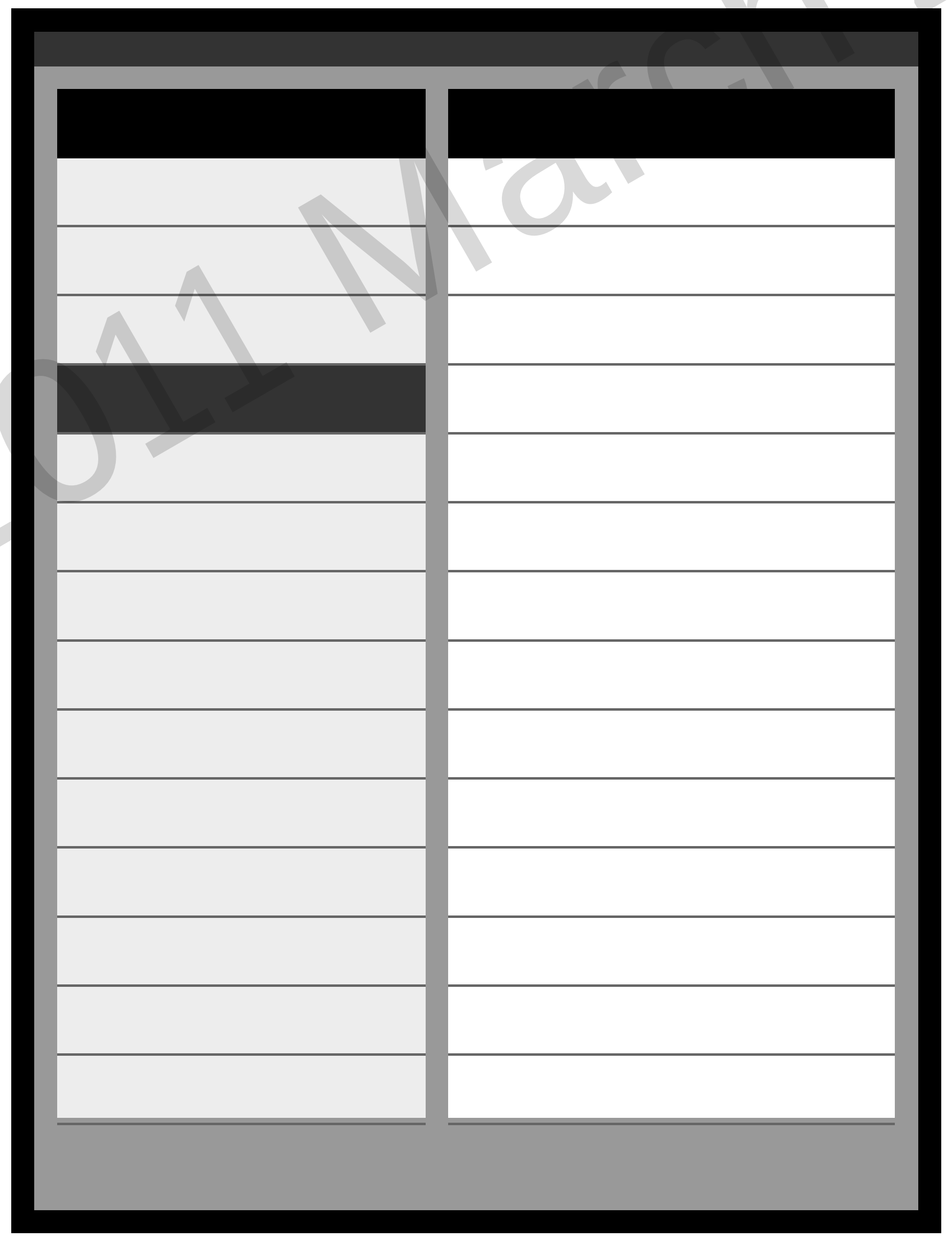
DESCRIPTION

- Use this layout to structure your application into two pane - the left hand pane is fixed in width whereas the content on the right hand pane is free
- The left pane contains a list of items. One item in the list should be selected and the right pane should show content that is related to the selected item
- Each pane can contains its own independent header and footer area which contains controls that relate to the pane

APP EXAMPLES

- Photos and Videos

SPLIT VIEW (SEPARATED PANES)

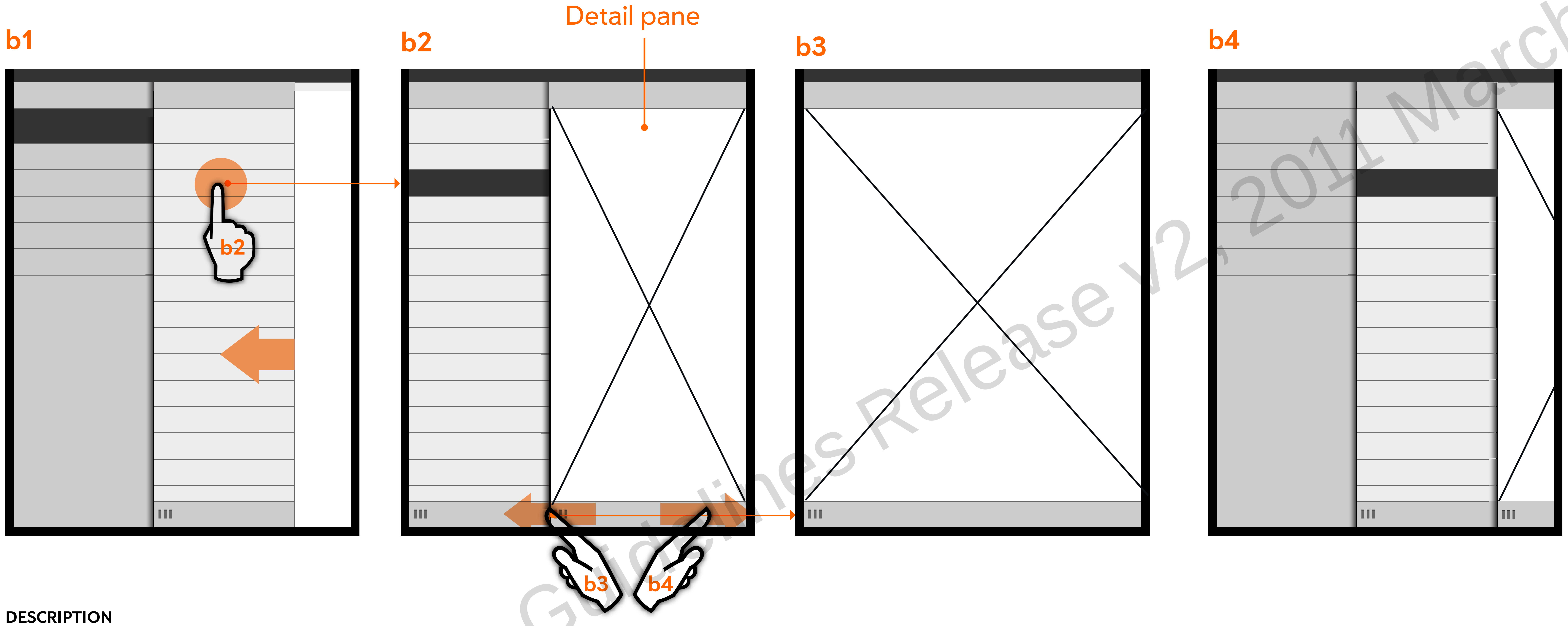


DESCRIPTION

- A similar layout to Split View (Full Width Panes) but the panes have a border around them and therefore a separated from each other.
- The view as a whole contains a footer area for general actions that is shared between the two panes

LAYOUTS: SLIDING PANES

PANE NAVIGATION



DESCRIPTION

- The Sliding Panes layout uses multiple panes to organize content hierarchically - each level in the hierarchy is represented by a pane. The topmost level corresponds to the pane furthest to the left and lower levels slide into view from the right and on top of the panes below.
- New panes are added dynamically as the user drills down through the hierarchy - when the user selects an item in a pane, a new pane slides in from the right. There is no limit to the number of panes that can be stacked in this way.
- The currently selected list item within a pane is shown highlighted.
- Usually, up to 3 panes are visible at any time. In portrait, however, only a portion of the third pane is visible. If the user selects an item from a pane that is not the left most pane, the pane should automatically slide over to the left and a new pane will slide in from the right to make sure that the new pane is visible.
- Eventually, when the bottom of the hierarchy is reached, a detail pane is revealed.
- Generally, panes are 320 pixels in width but the detail pane can vary its width to fill the remaining space and can fill the entire width of the screen if it is the only pane shown. Content within the pane should re-flow to fit the available space.
- Each pane can contain its own header and footer area

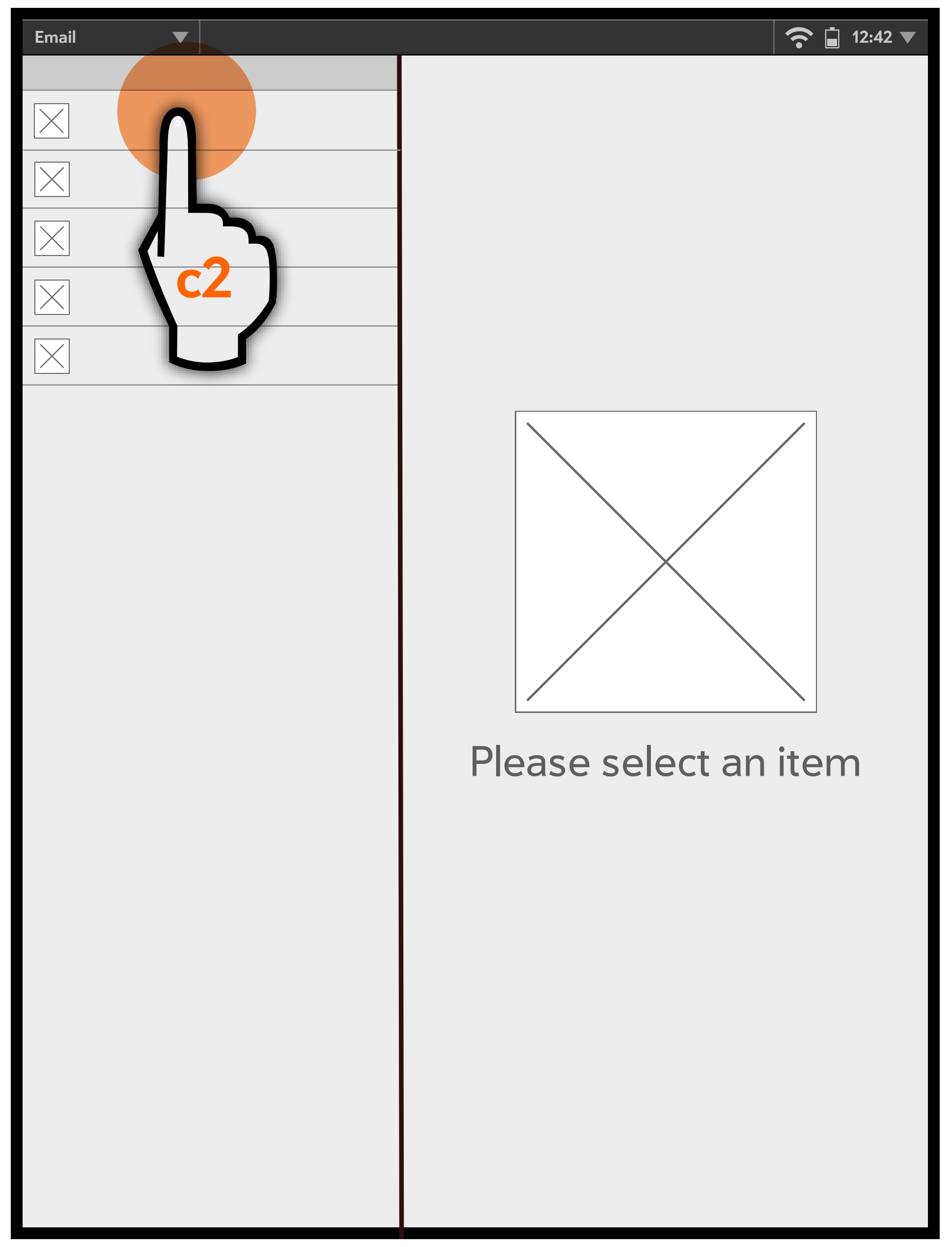
APP EXAMPLES

- Email

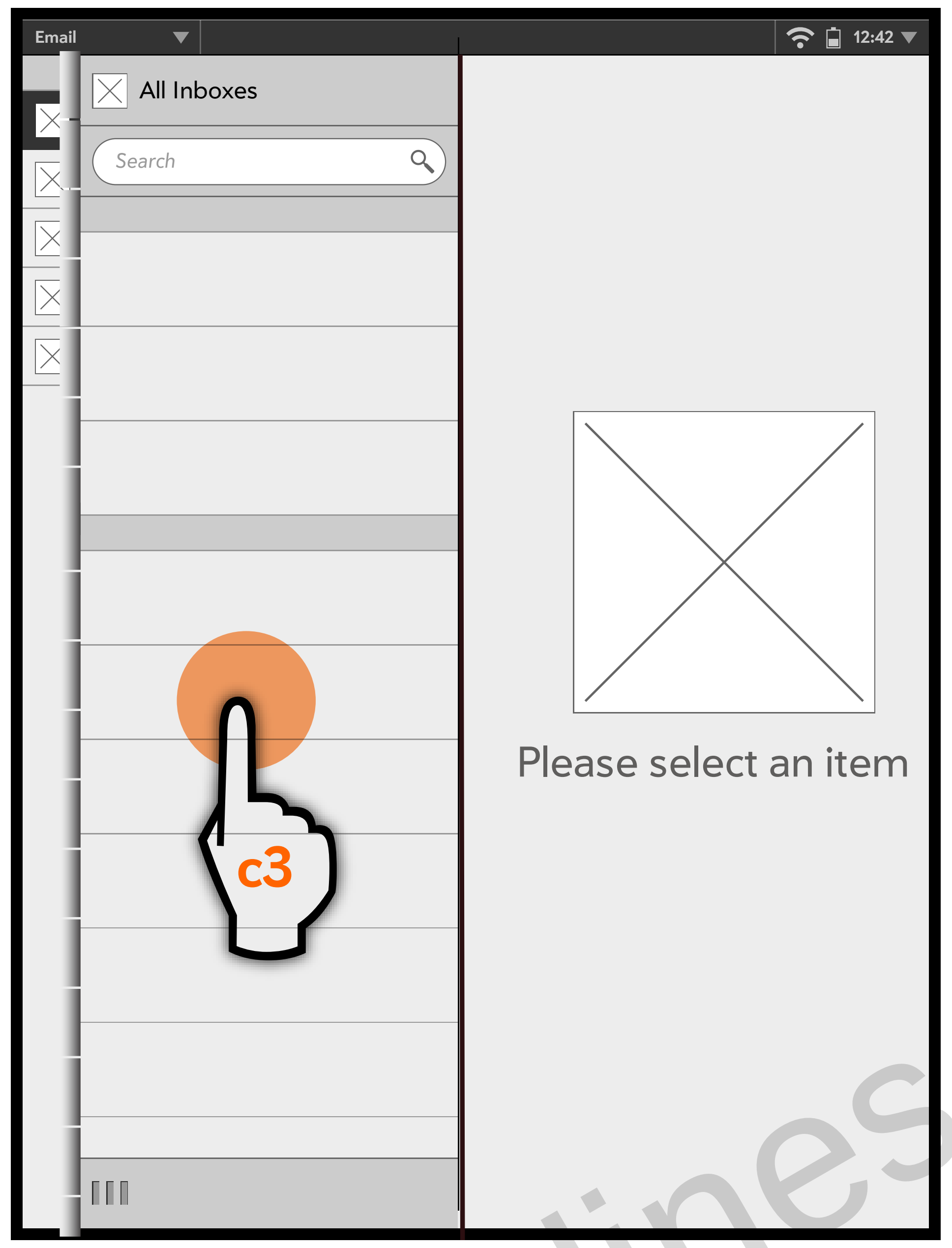
LAYOUTS: SLIDING PANES

PANE PEEKING

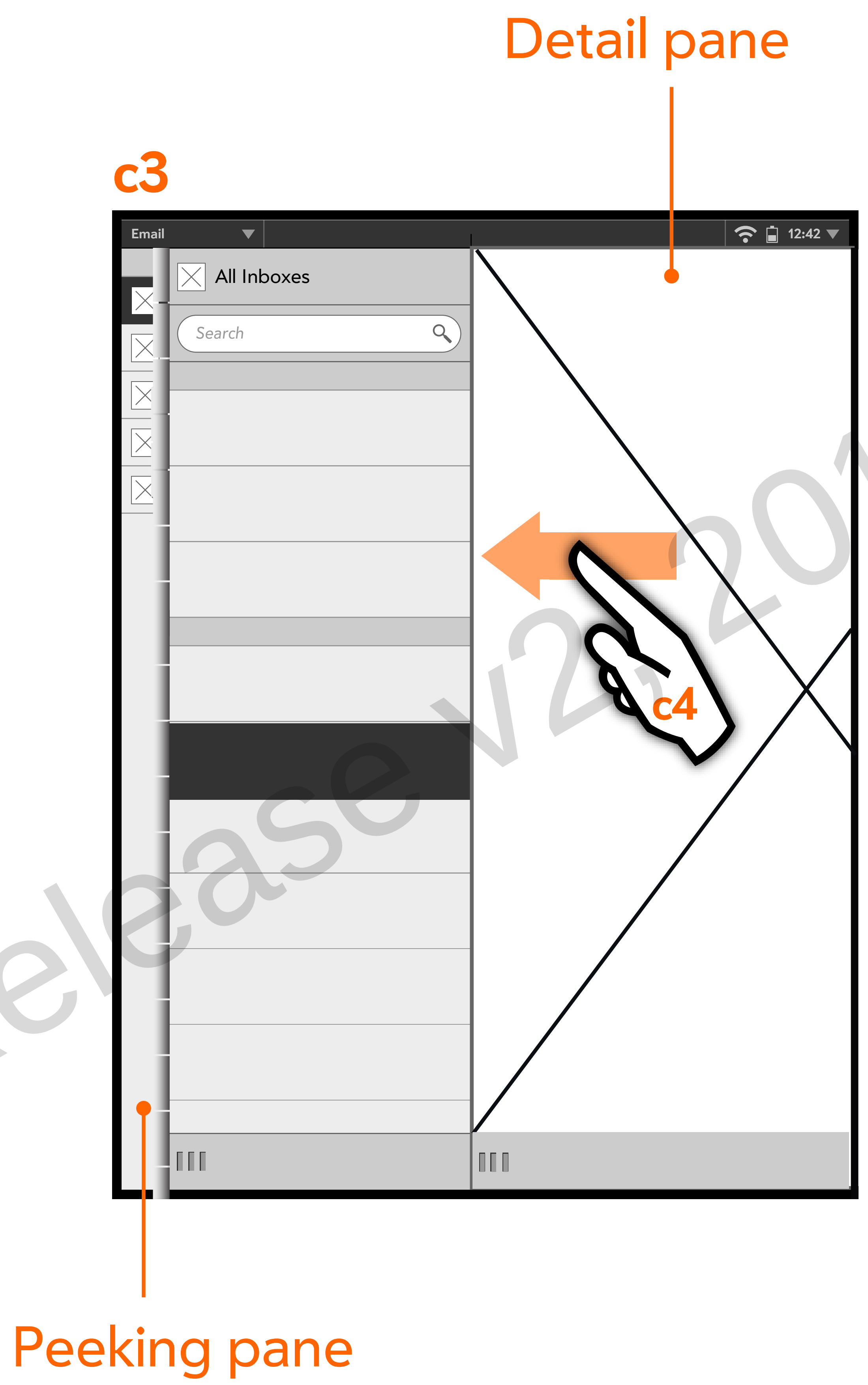
c1



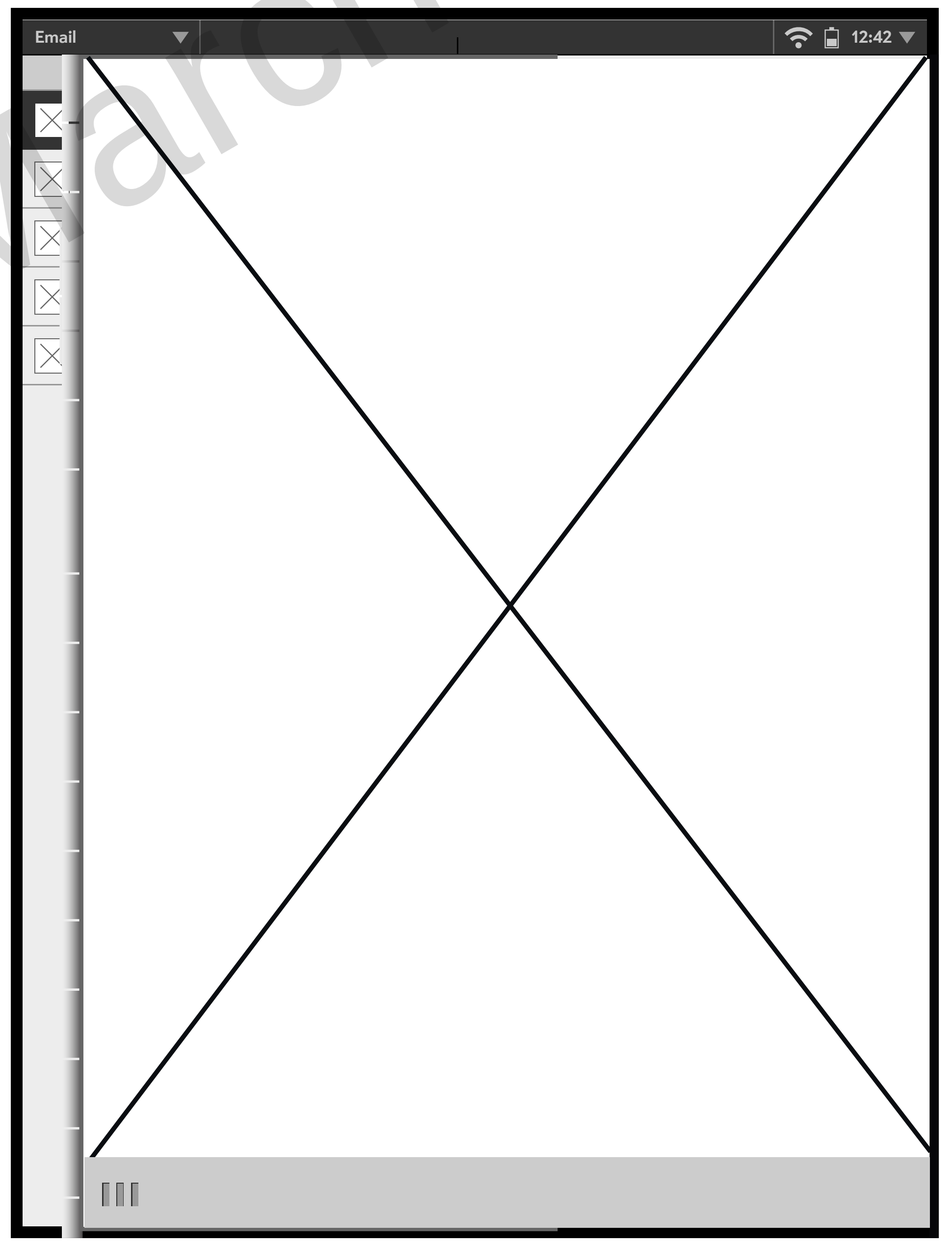
c2



c3



c4



DESCRIPTION

- Panes can be designed such that a portion of the left-most pane (50 pixels is recommended) always remains visible
- This allows the user to always have a ‘peek’ of the top-level navigation options and easily switch between them no matter how deep the user is within the hierarchy

It is recommended that peeking is only employed if the following conditions are met:

- There is minimal content in the peeking pane (no scrolling)
- There are no collapsing or expanding list items in the peeking pane
- Icons are easy to recognize
- There is a strong use case for being able to always access content in the peeking pane

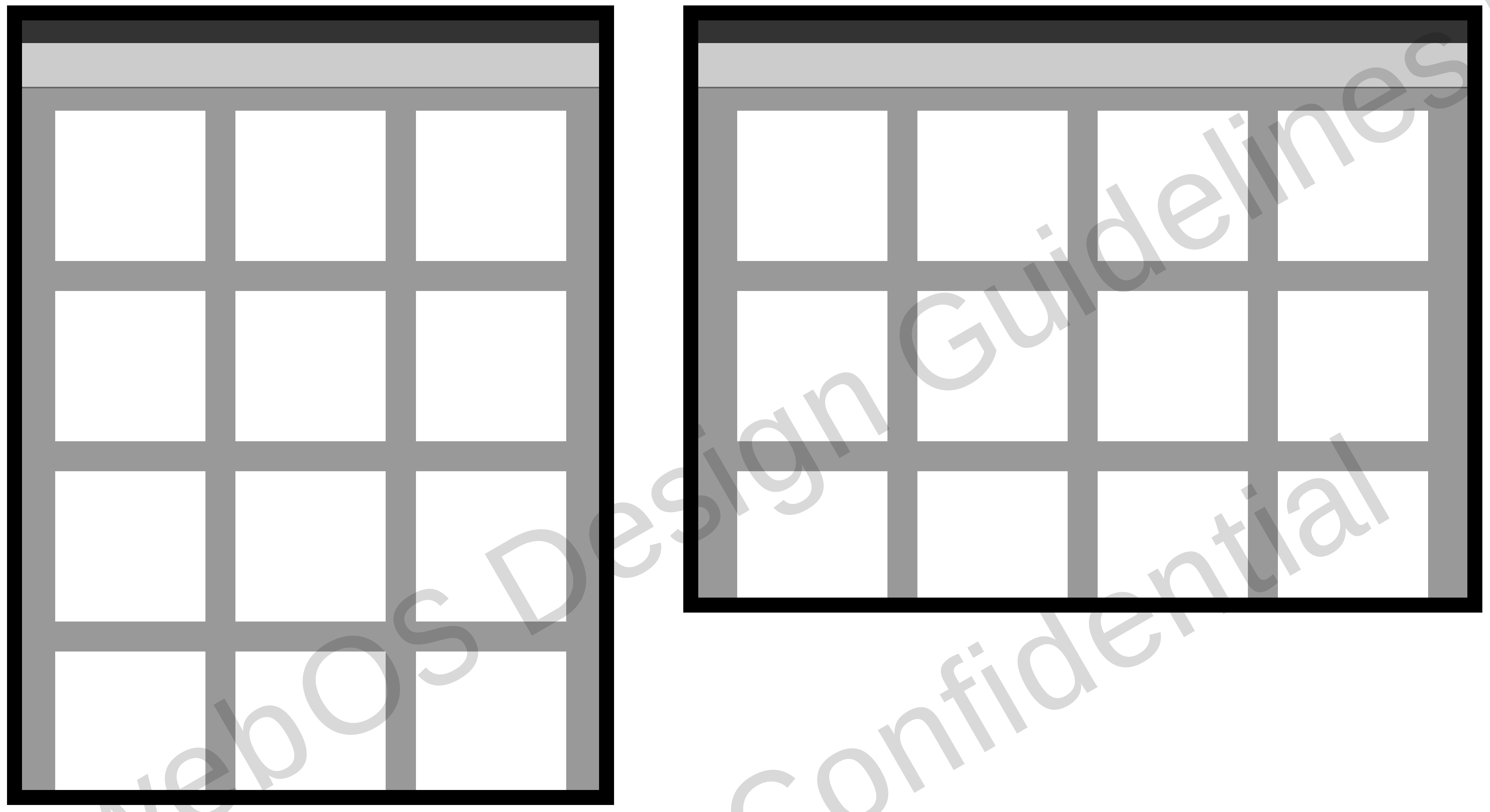
PORTRAIT AND LANDSCAPE LAYOUTS

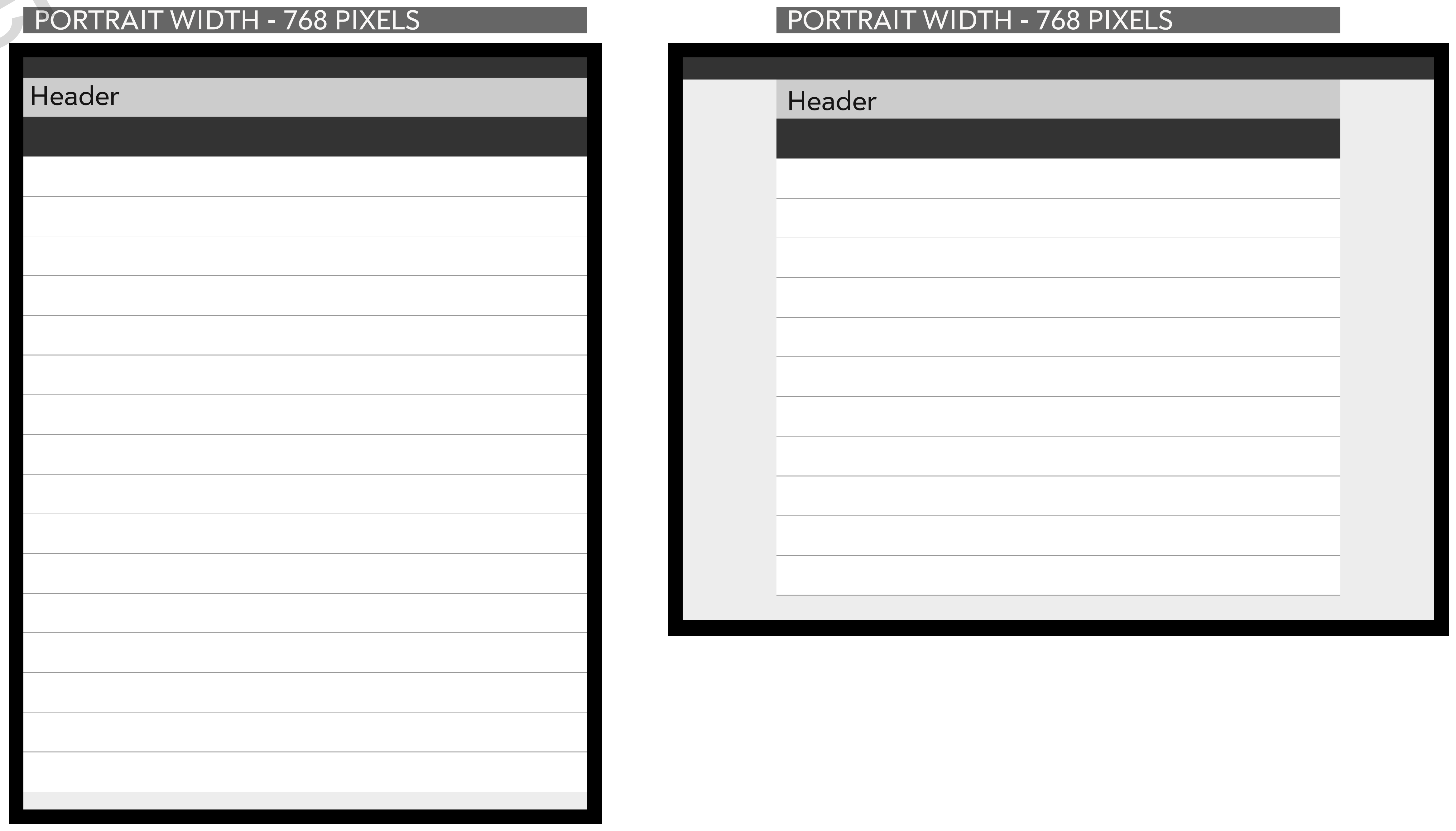
The device accelerometer can capture orientation events to detect whether the device is being held in a portrait or landscape orientation. Your application has the choice to constrain itself to a particular orientation, or to readjust its layout to match the device orientation between portrait and landscape. It is recommended, however, that applications generally provide both a portrait and landscape layout.

Panes will automatically re-size to fit the available width in the view unless they are fixed in width. Consider how your application content will appear within the view's panes in both orientations and rearrange the layout of the content if necessary to maximize the amount of content displayed in the available space.

Note, Header and Footers adjust to fit the full width of the pane that they are in. **The items that appear in the header and footer areas should remain the same even if the available space may change between orientations.** Switching between orientations should also not introduce new content or functionality.

SINGLE PANE LAYOUTS





DESCRIPTION

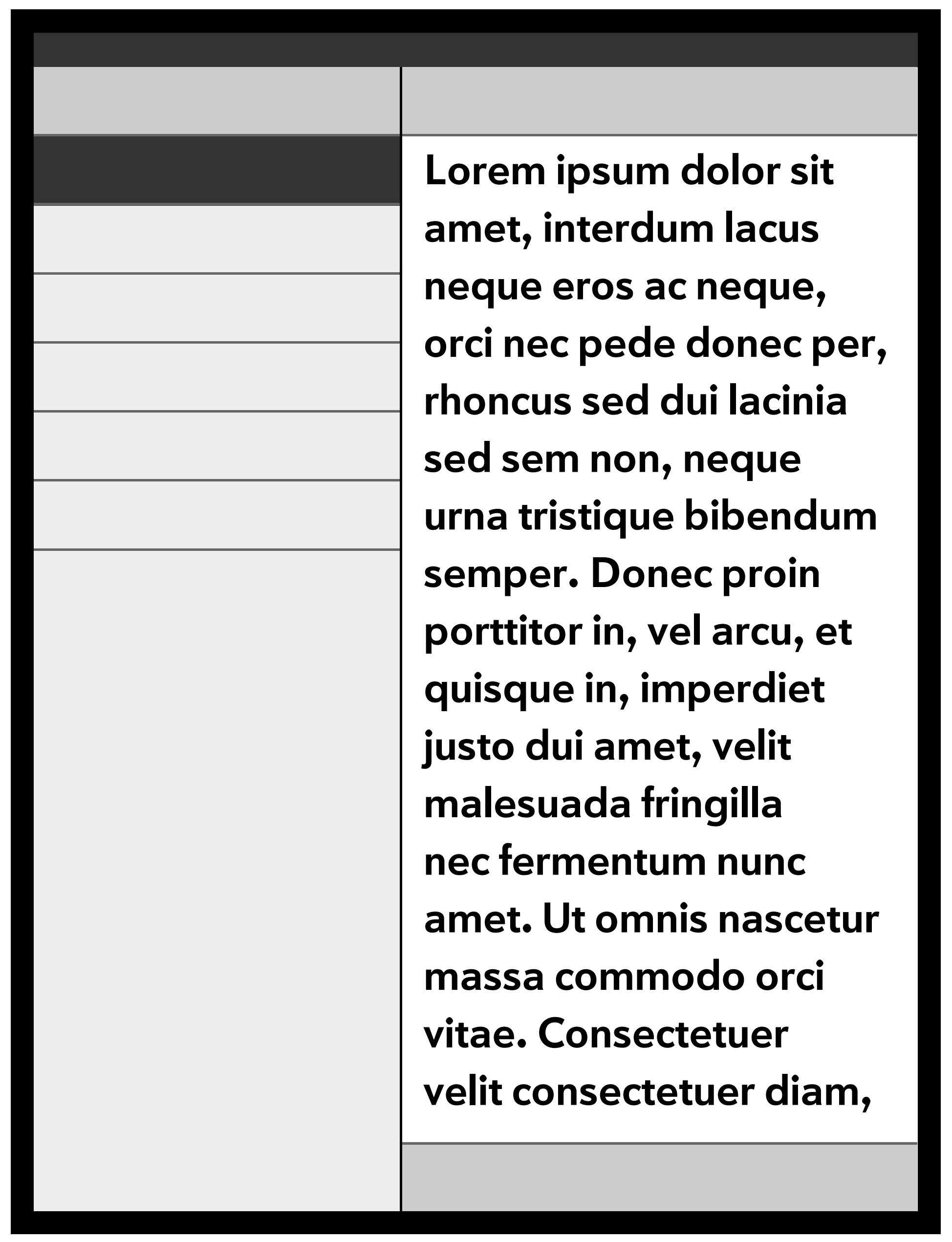
- Grid-based layouts scale well between portrait and landscape. A grid-based layout should re-order to fit as many grid items as possible within the horizontal space within the pane.

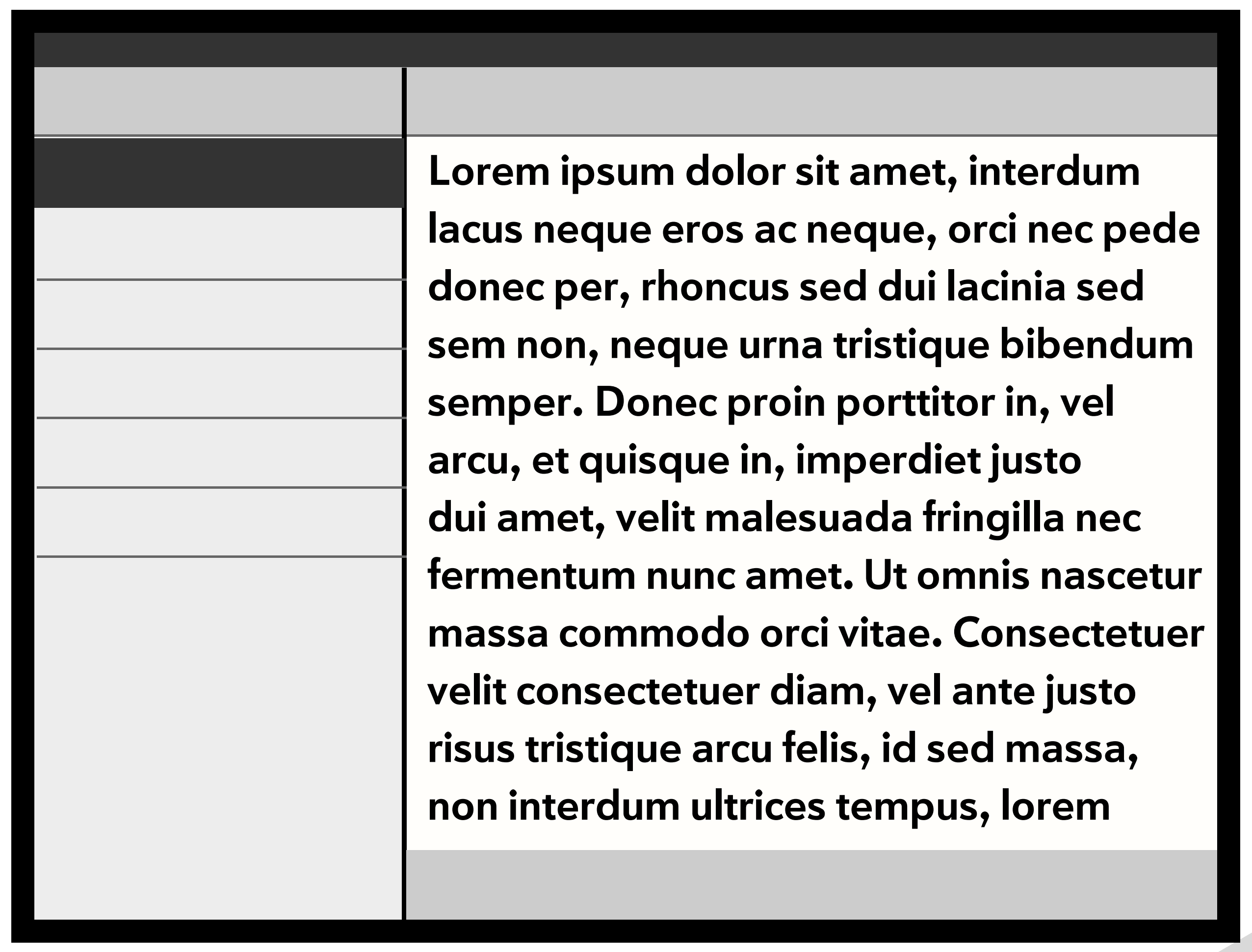
DESCRIPTION

- A single column layout does not scale well between portrait and landscape. This layout should be avoided but when it is necessary, do not scale the column to full-width in landscape.
- The overall width of the column should be maintained between landscape and portrait orientations.

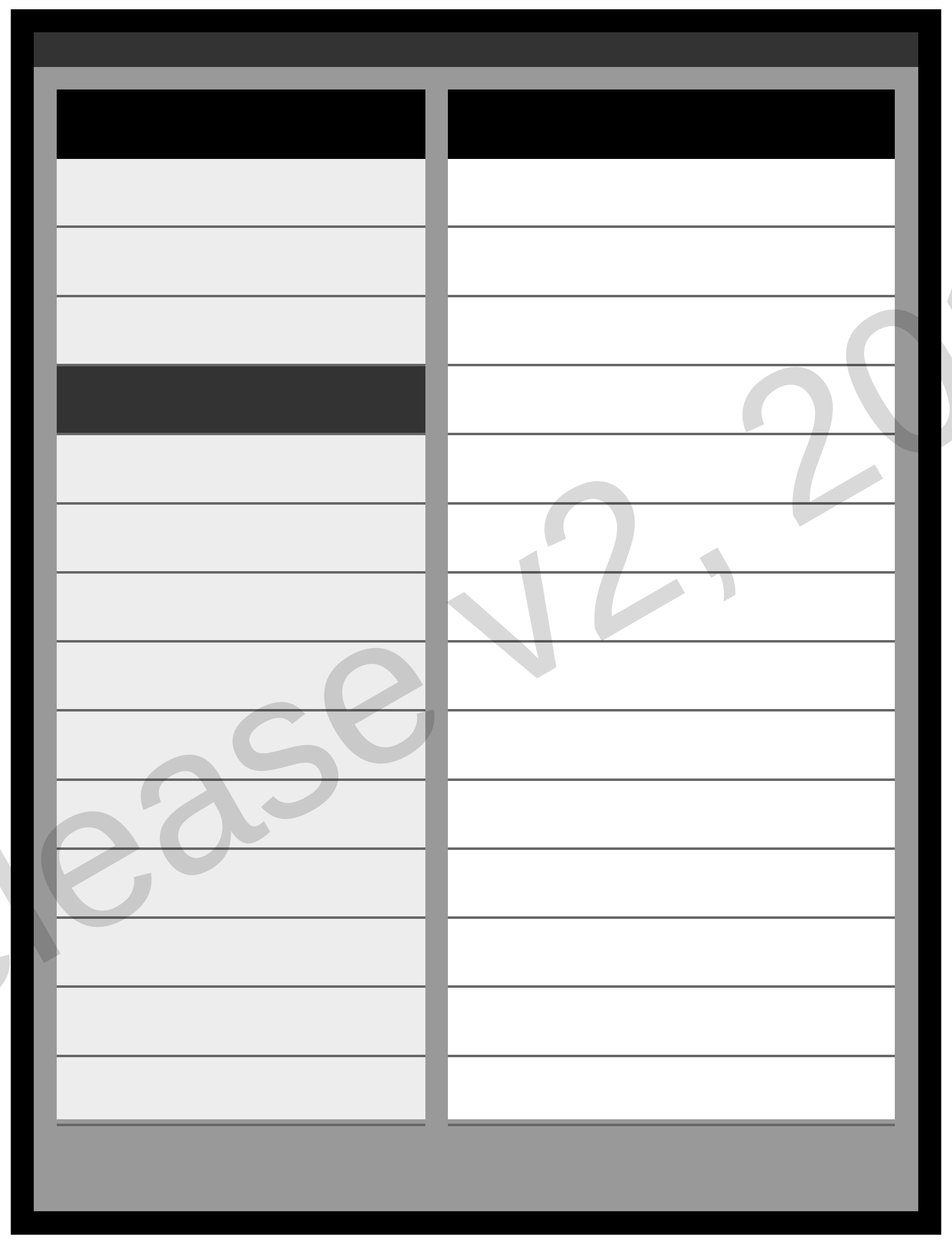
PORTRAIT AND LANDSCAPE LAYOUTS

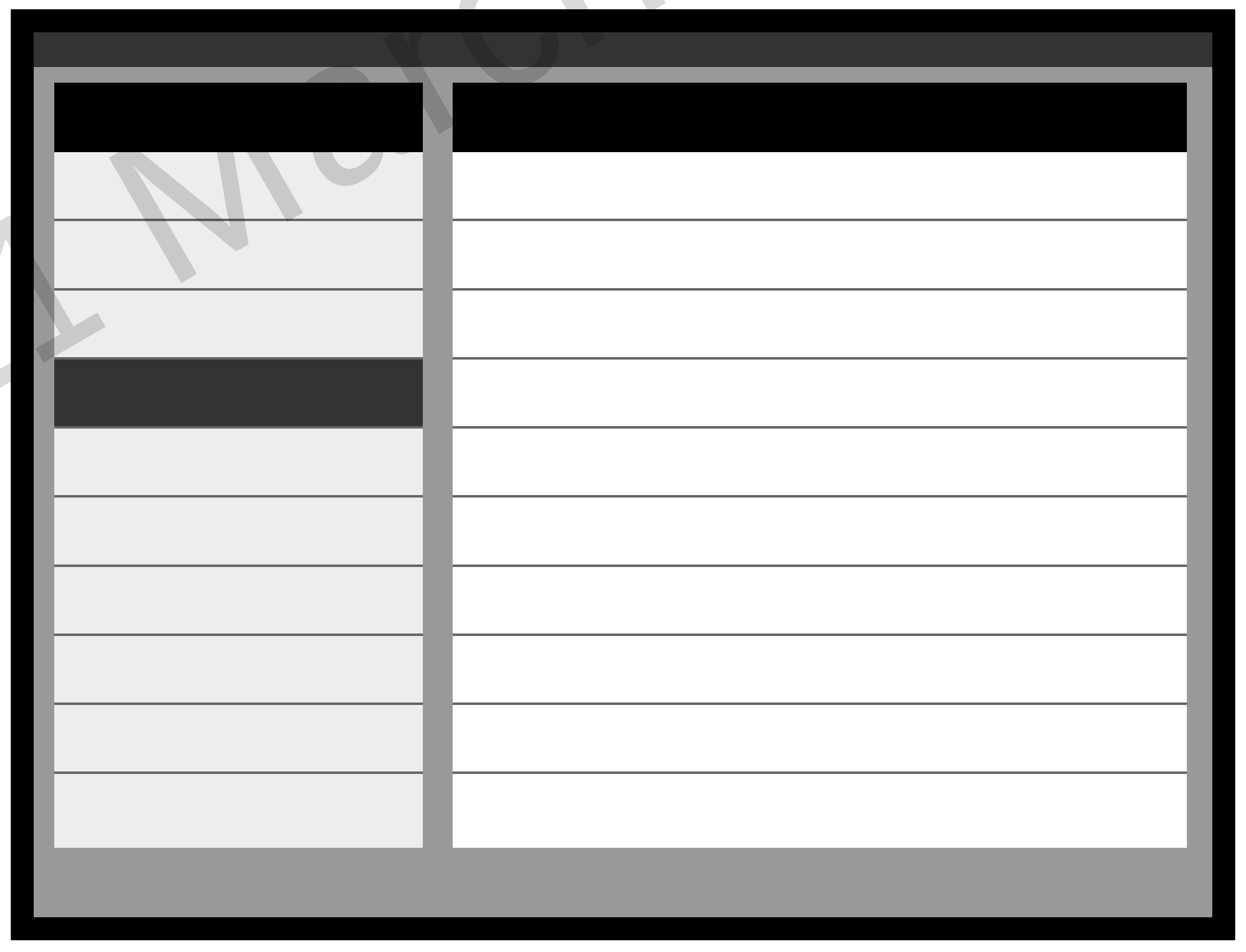
SPLIT VIEW (FULL WIDTH PANES)


Lorem ipsum dolor sit amet, interdum lacus neque eros ac neque, orci nec pede donec per, rhoncus sed dui lacinia sed sem non, neque urna tristique bibendum semper. Donec proin porttitor in, vel arcu, et quisque in, imperdiet justo dui amet, velit malesuada fringilla nec fermentum nunc amet. Ut omnis nascetur massa commodo orci vitae. Consectetuer velit consectetuer diam,


Lorem ipsum dolor sit amet, interdum lacus neque eros ac neque, orci nec pede donec per, rhoncus sed dui lacinia sed sem non, neque urna tristique bibendum semper. Donec proin porttitor in, vel arcu, et quisque in, imperdiet justo dui amet, velit malesuada fringilla nec fermentum nunc amet. Ut omnis nascetur massa commodo orci vitae. Consectetuer velit consectetuer diam, vel ante justo risus tristique arcu felis, id sed massa, non interdum ultrices tempus, lorem

SPLIT VIEW (SEPARATED PANES)





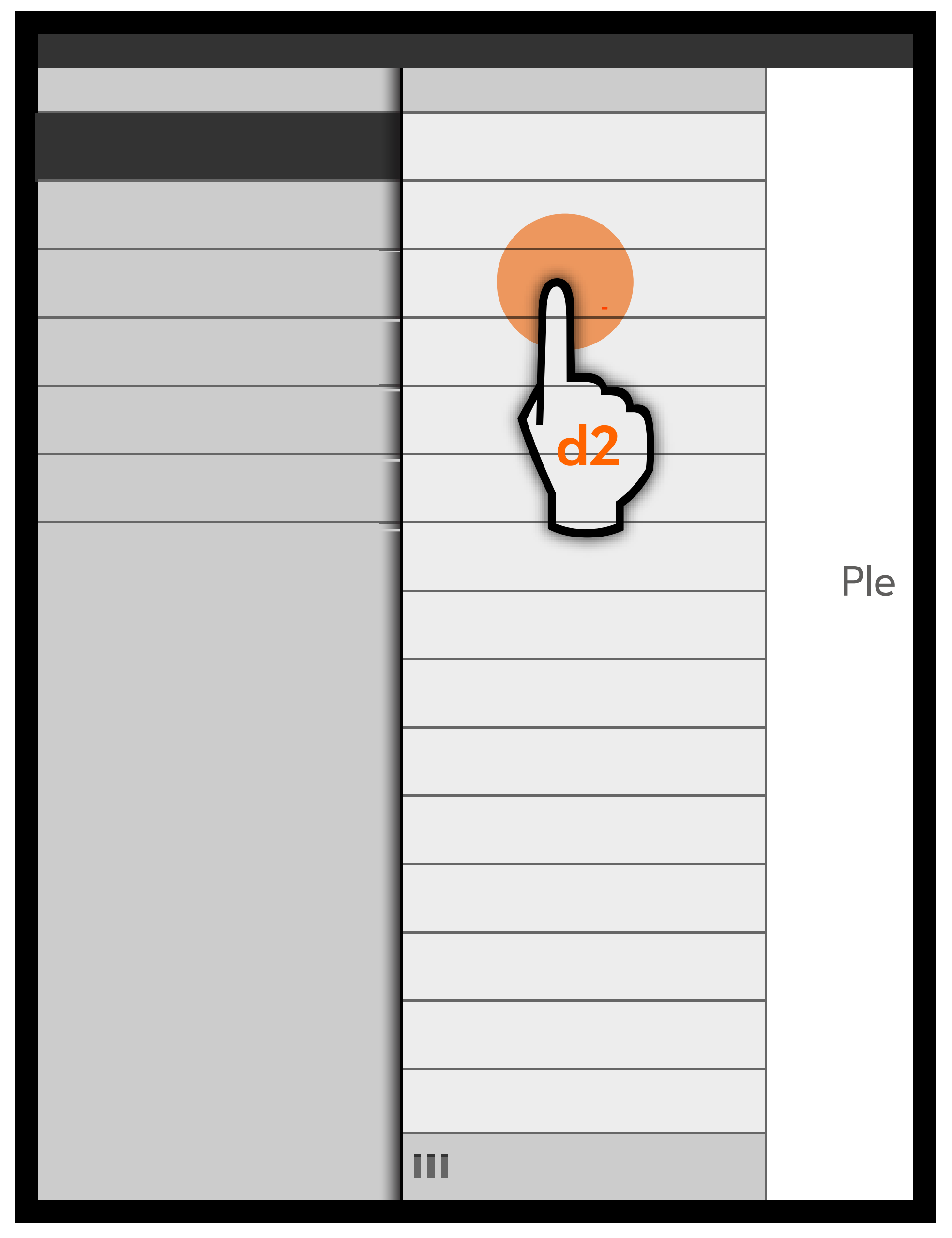
DESCRIPTION

- A split view contains a fixed pane on the left and a free pane on the right.
- The fixed pane remains the same width regardless of orientation and the free pane fits the available space
- The content in the free pane should re-flow to fill the available space when the orientation changes
- Users will typically change to a landscape orientation to see more content in this layout - try to fulfil users' expectations by ensuring that the amount of content displayed in landscape is maximized

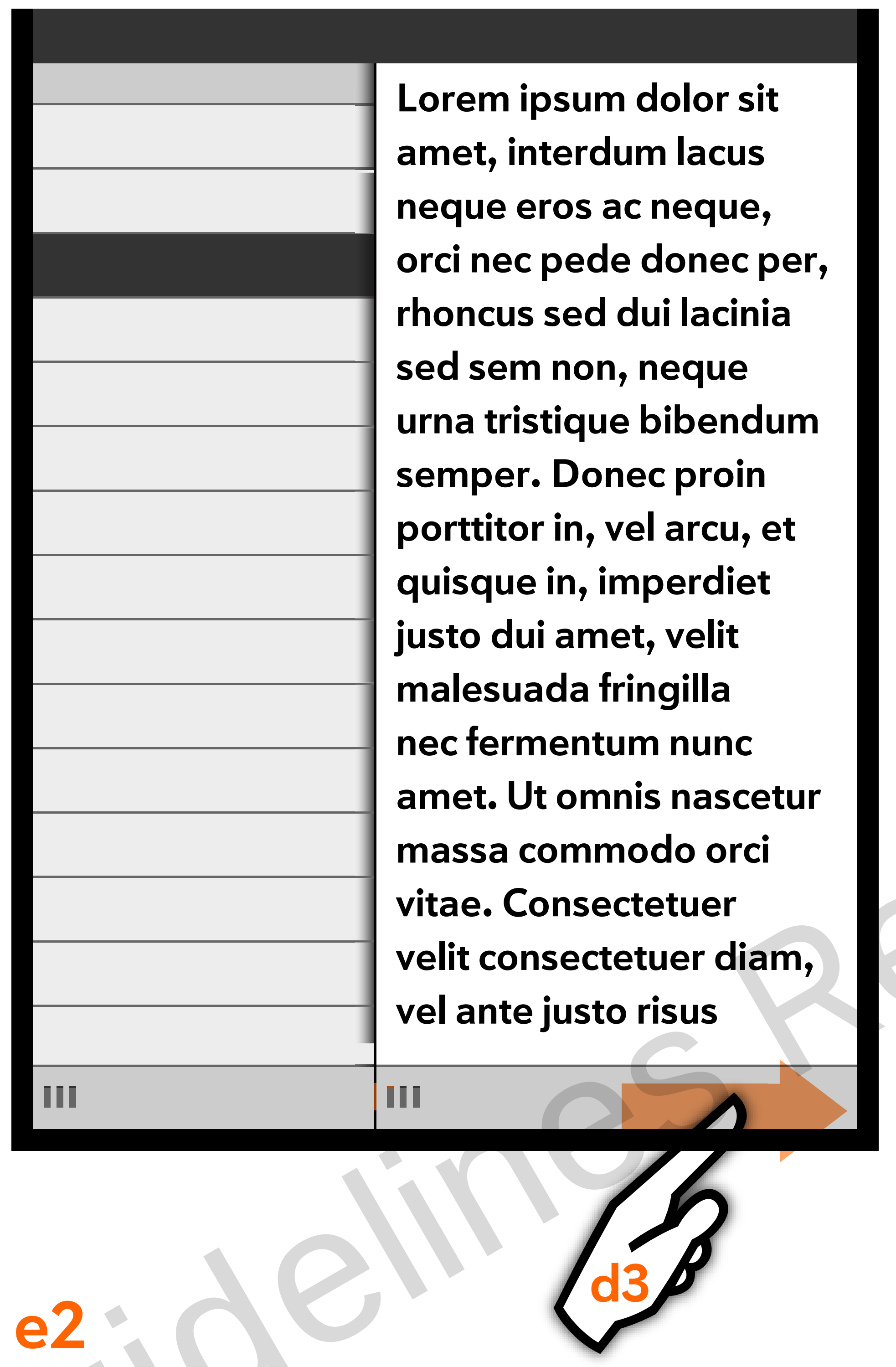
PORTRAIT AND LANDSCAPE LAYOUTS

SLIDING PANES

d1

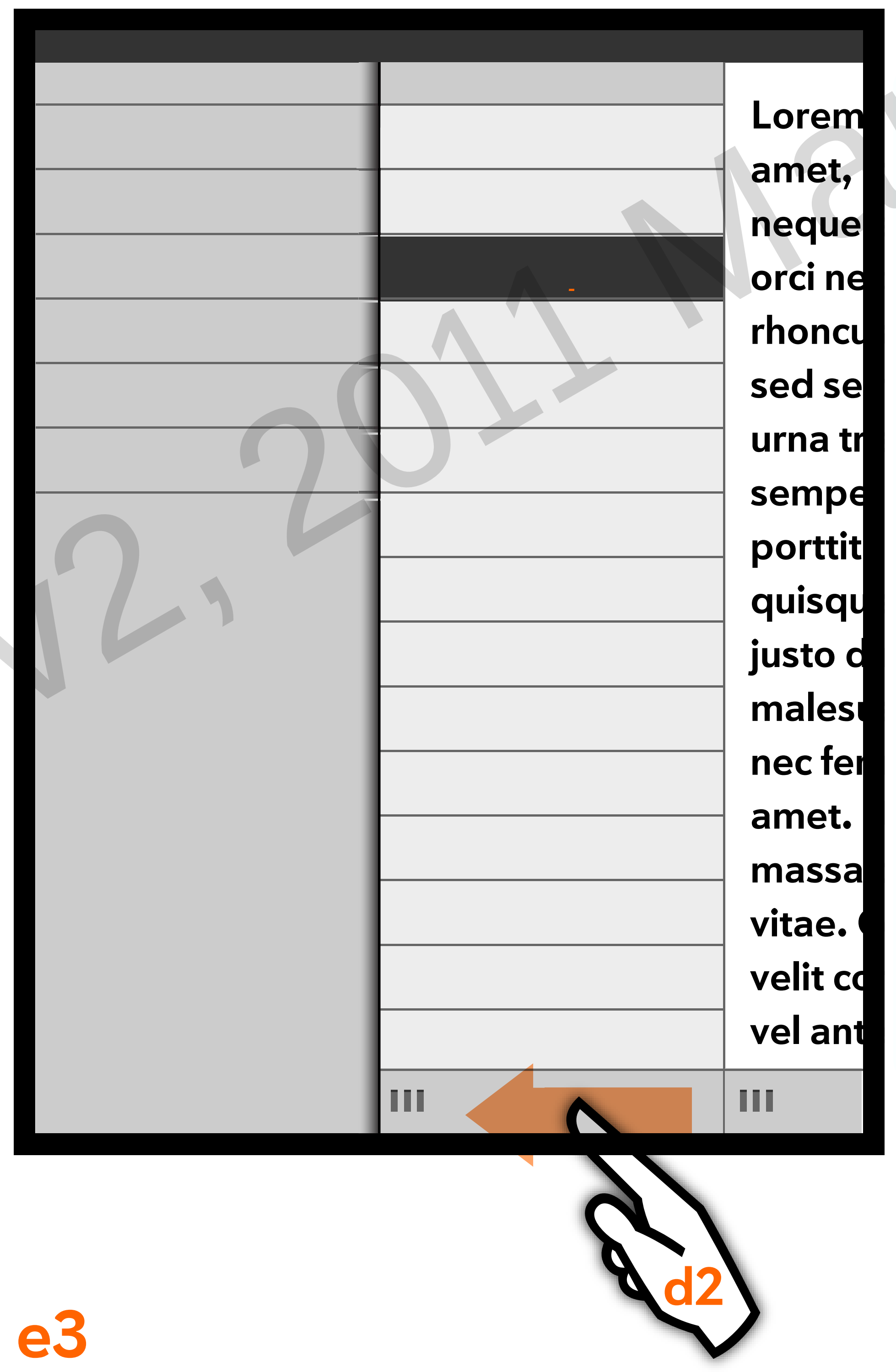


d2

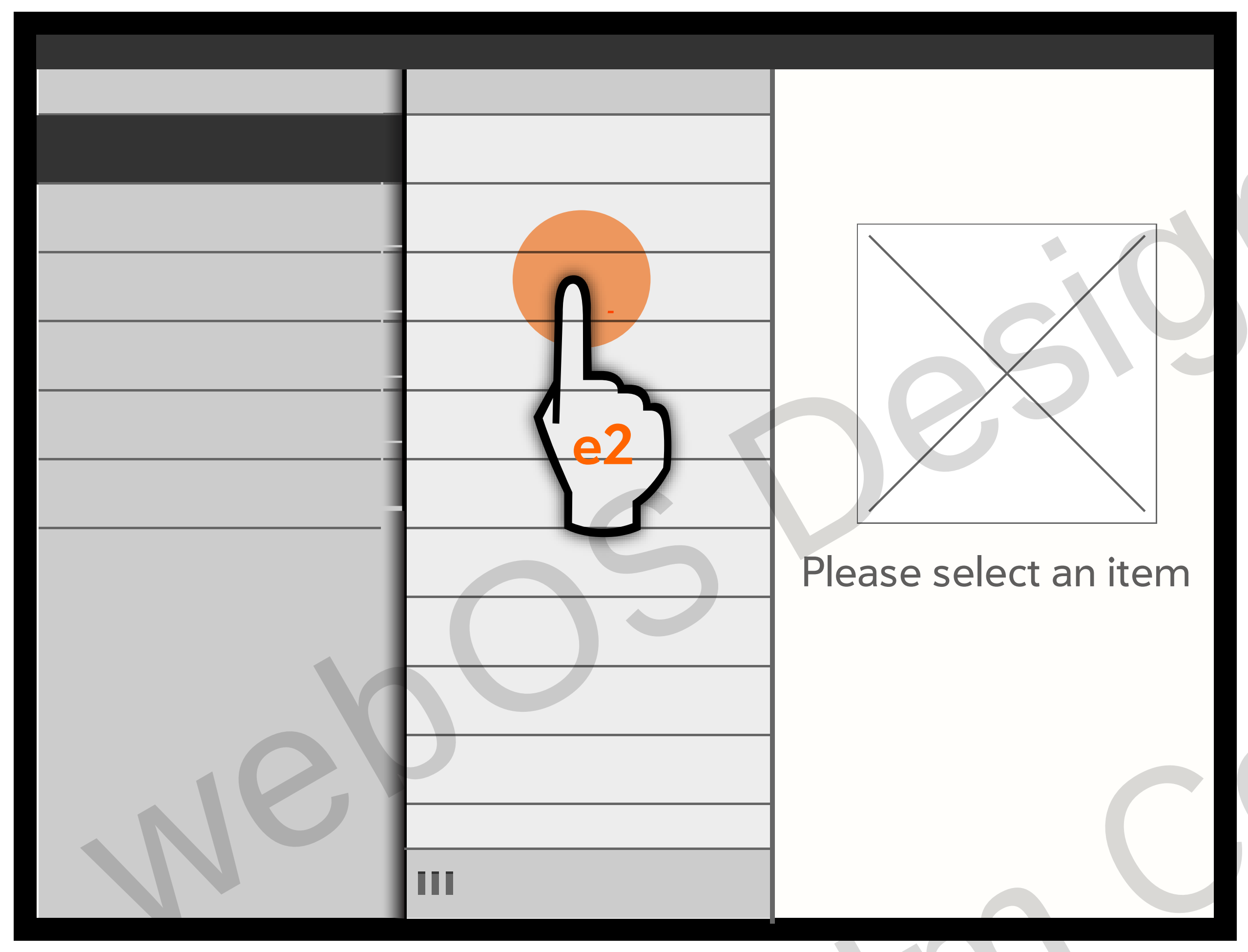


Lorem ipsum dolor sit amet, interdum lacus neque eros ac neque, orci nec pede donec per, rhoncus sed dui lacinia sed sem non, neque urna tristique bibendum semper. Donec proin porttitor in, vel arcu, et quisque in, imperdiet justo dui amet, velit malesuada fringilla nec fermentum nunc amet. Ut omnis nascetur massa commodo orci vitae. Consectetuer velit consectetuer diam, vel ante justo risus
d3

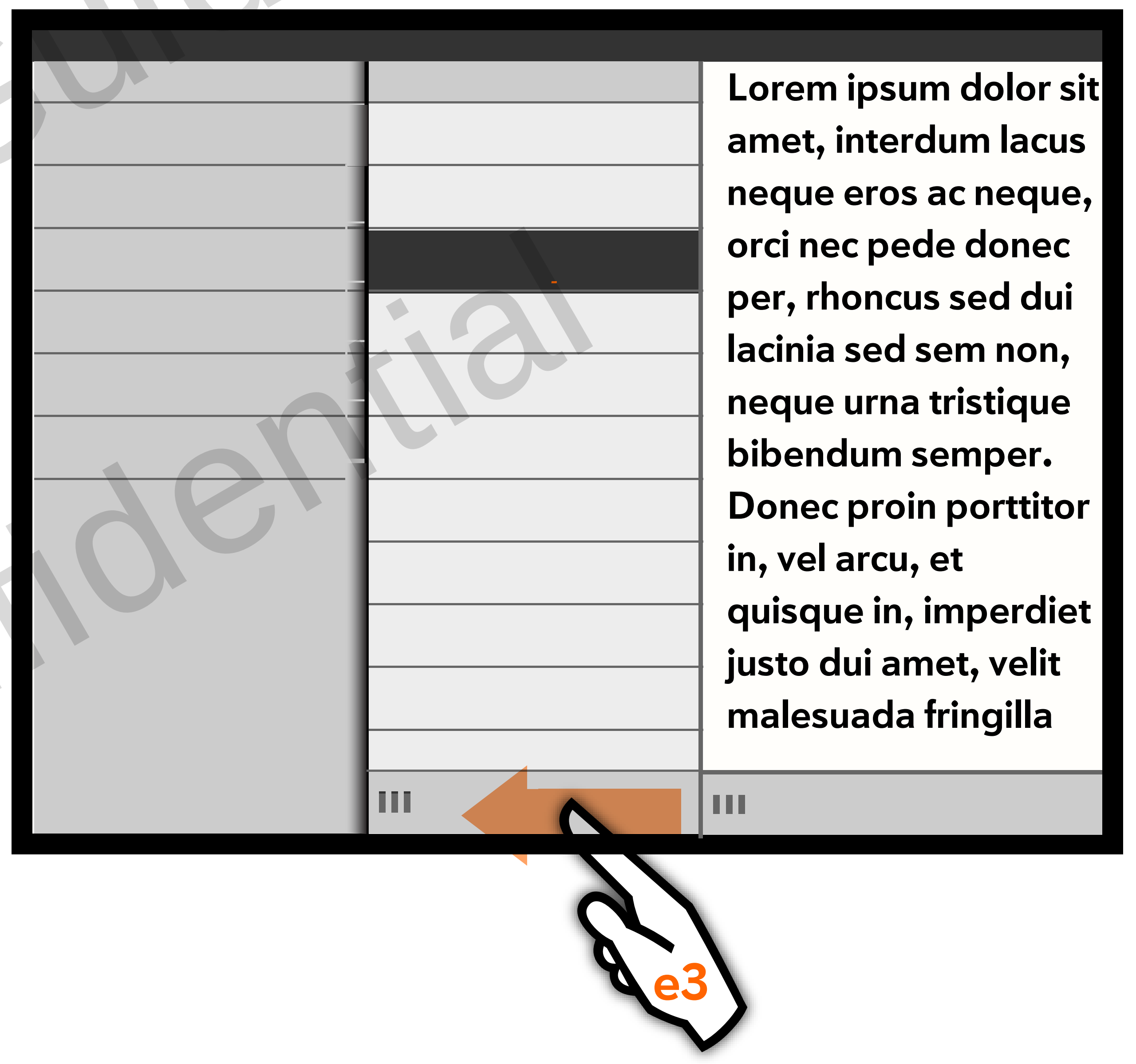
d3



e1

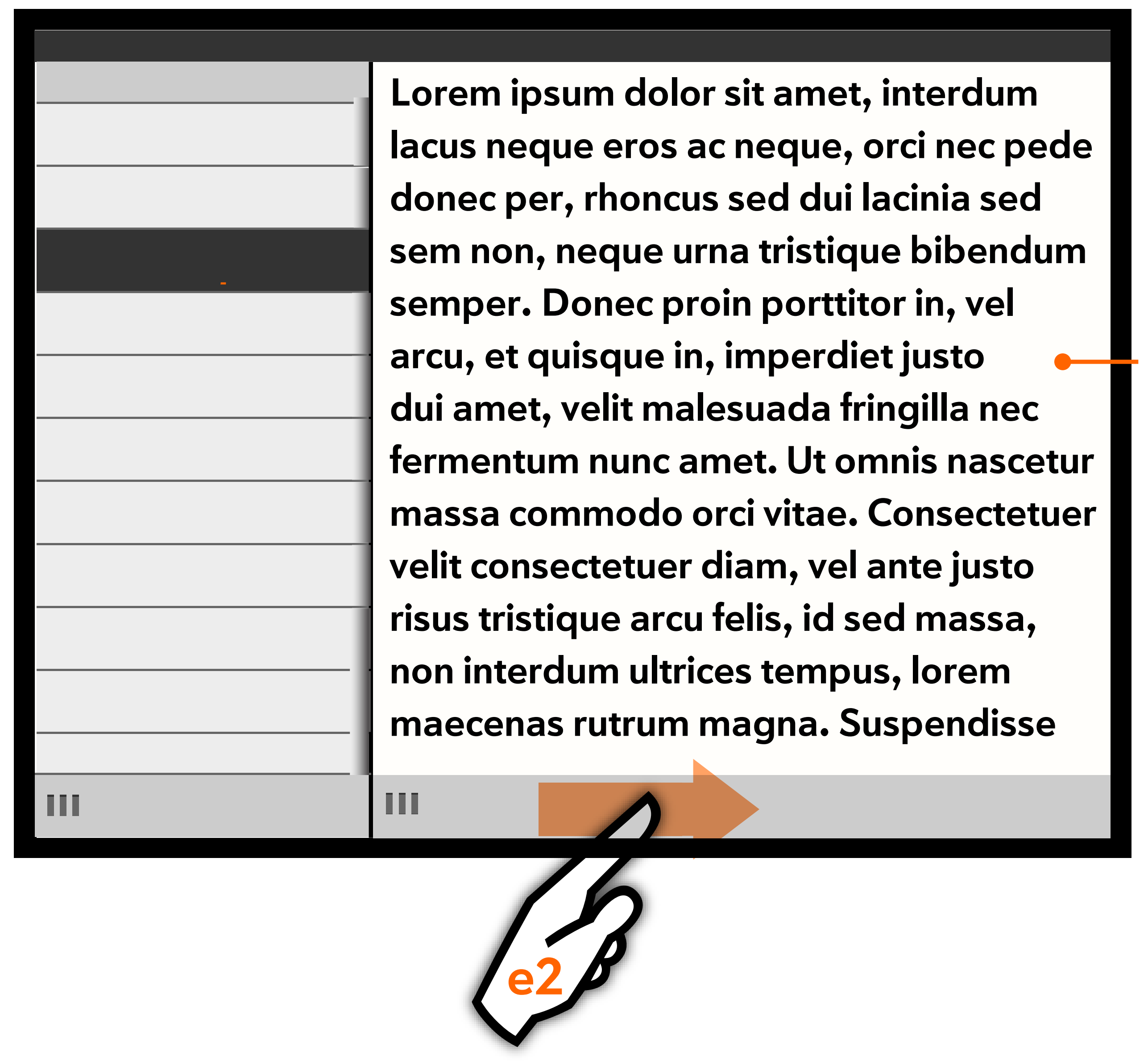


e2



Lorem ipsum dolor sit amet, interdum lacus neque eros ac neque, orci nec pede donec per, rhoncus sed dui lacinia sed sem non, neque urna tristique bibendum semper. Donec proin porttitor in, vel arcu, et quisque in, imperdiet justo dui amet, velit malesuada fringilla
e3

e3



Lorem ipsum dolor sit amet, interdum lacus neque eros ac neque, orci nec pede donec per, rhoncus sed dui lacinia sed sem non, neque urna tristique bibendum semper. Donec proin porttitor in, vel arcu, et quisque in, imperdiet justo dui amet, velit malesuada fringilla nec fermentum nunc amet. Ut omnis nascetur massa commodo orci vitae. Consectetuer velit consectetuer diam, vel ante justo risus tristique arcu felis, id sed massa, non interdum ultrices tempus, lorem maecenas rutrum magna. Suspendisse
e2

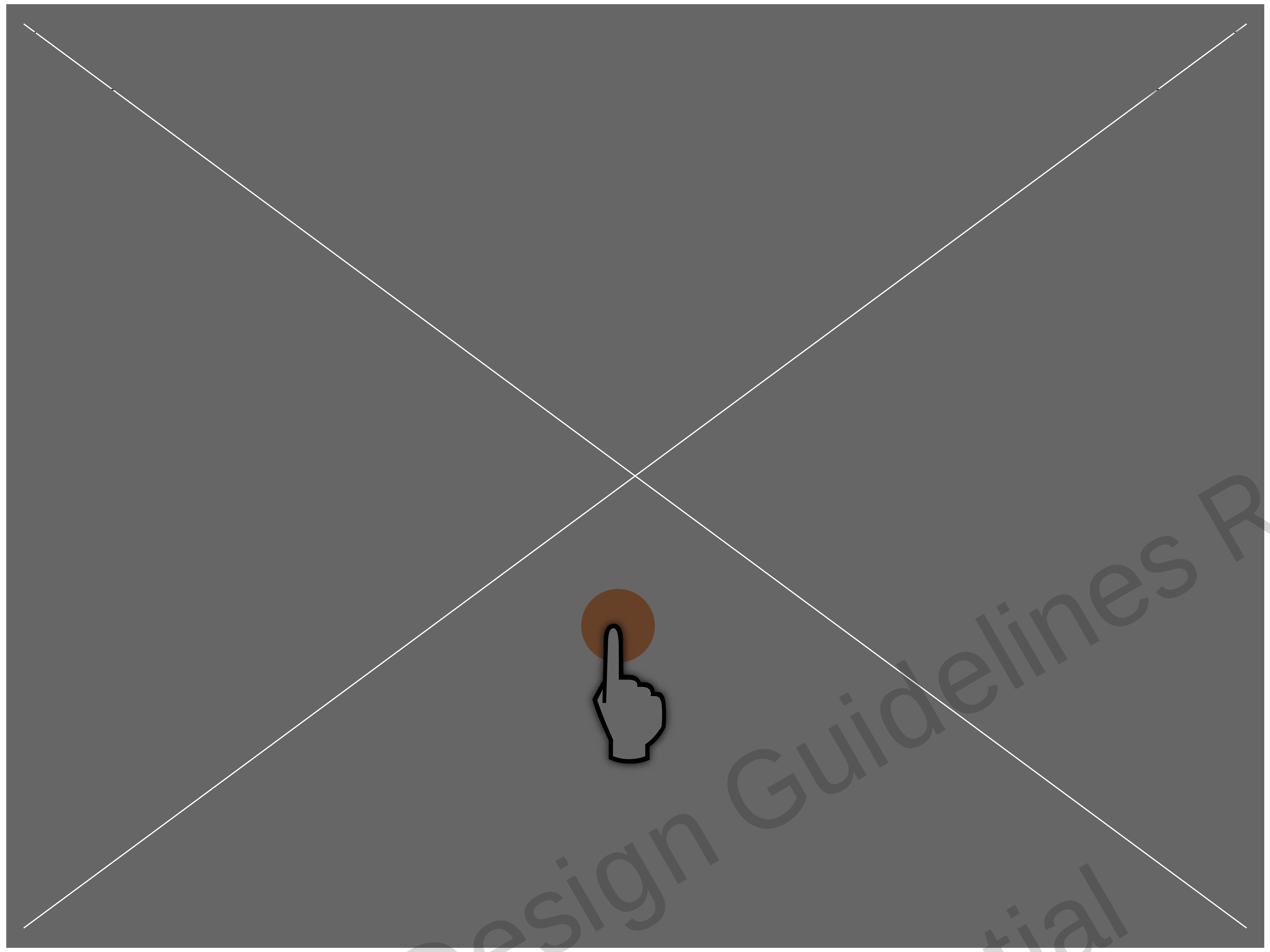
Content in the details pane re-flows to fit the available width provided the pane is at least 320 pixels wide

DESCRIPTION

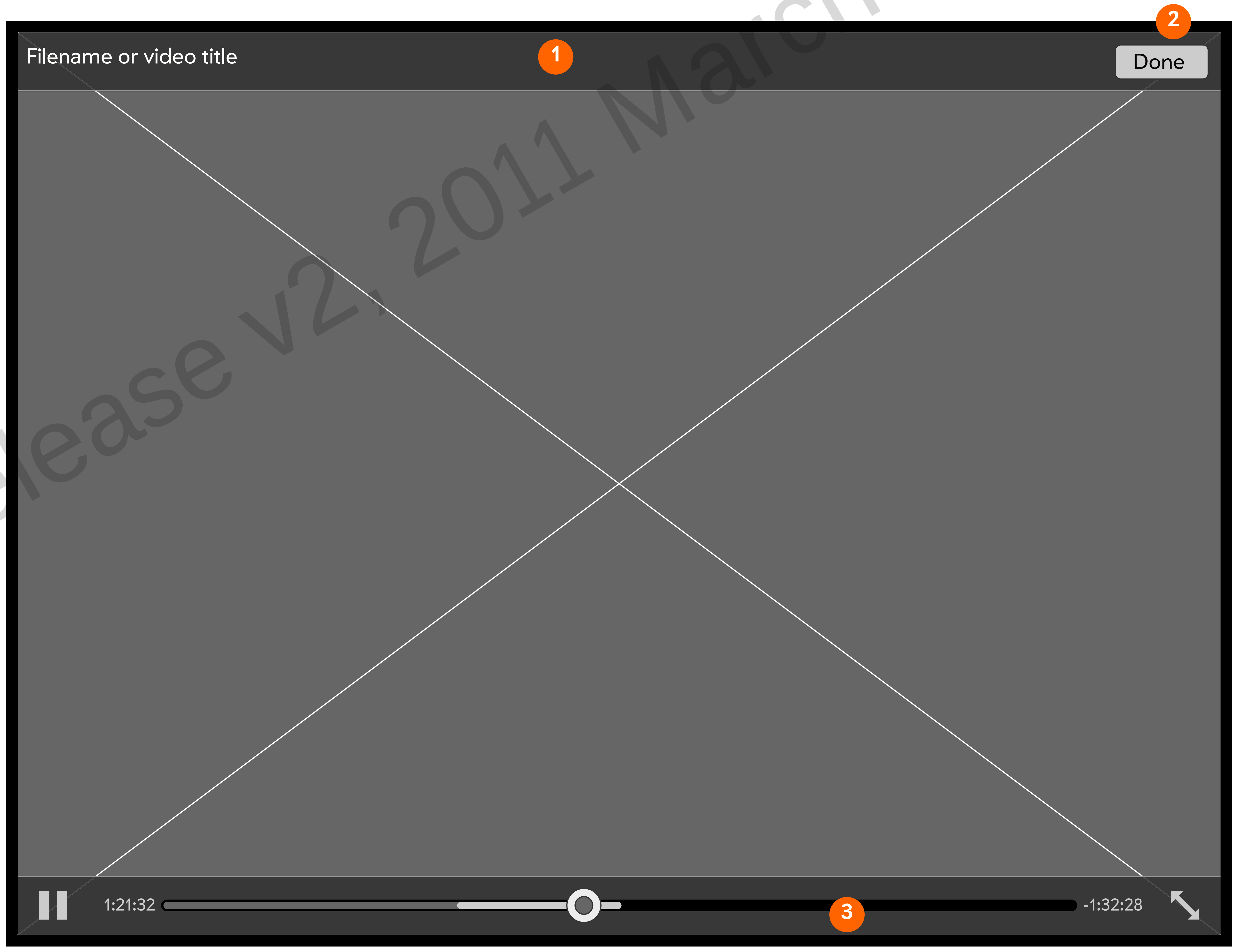
- Panes within the Sliding Panes layout are generally fixed in width, except for the detail pane which adjusts to fit the available space
- The content in the free pane should re-flow to fill the available space when the orientation changes, provided the width of the pane is at least 320 pixels wide

FULL SCREEN MODE

FULL SCREEN MODE



FULL SCREEN MODE - CHROME INVOKED



DESCRIPTION

- An application can be set to enter full screen mode if there is a need to provide an more immersive experience that maximizes space for content
- In this mode, all application chrome - i.e., the status bar and all header and footers - are dismissed and the content area fills the entire screen
- If the user taps on the display in full screen mode, the application has the option to display semi-transparent, transient header and footer areas to display navigational information and controls. The header and footer areas should fade out after a time-out
- If a notification is received in full screen mode, it is not displayed as the status bar is not present. If an alert is received, however, it is still displayed above the content area
- The application should always provide a means of exiting full screen mode

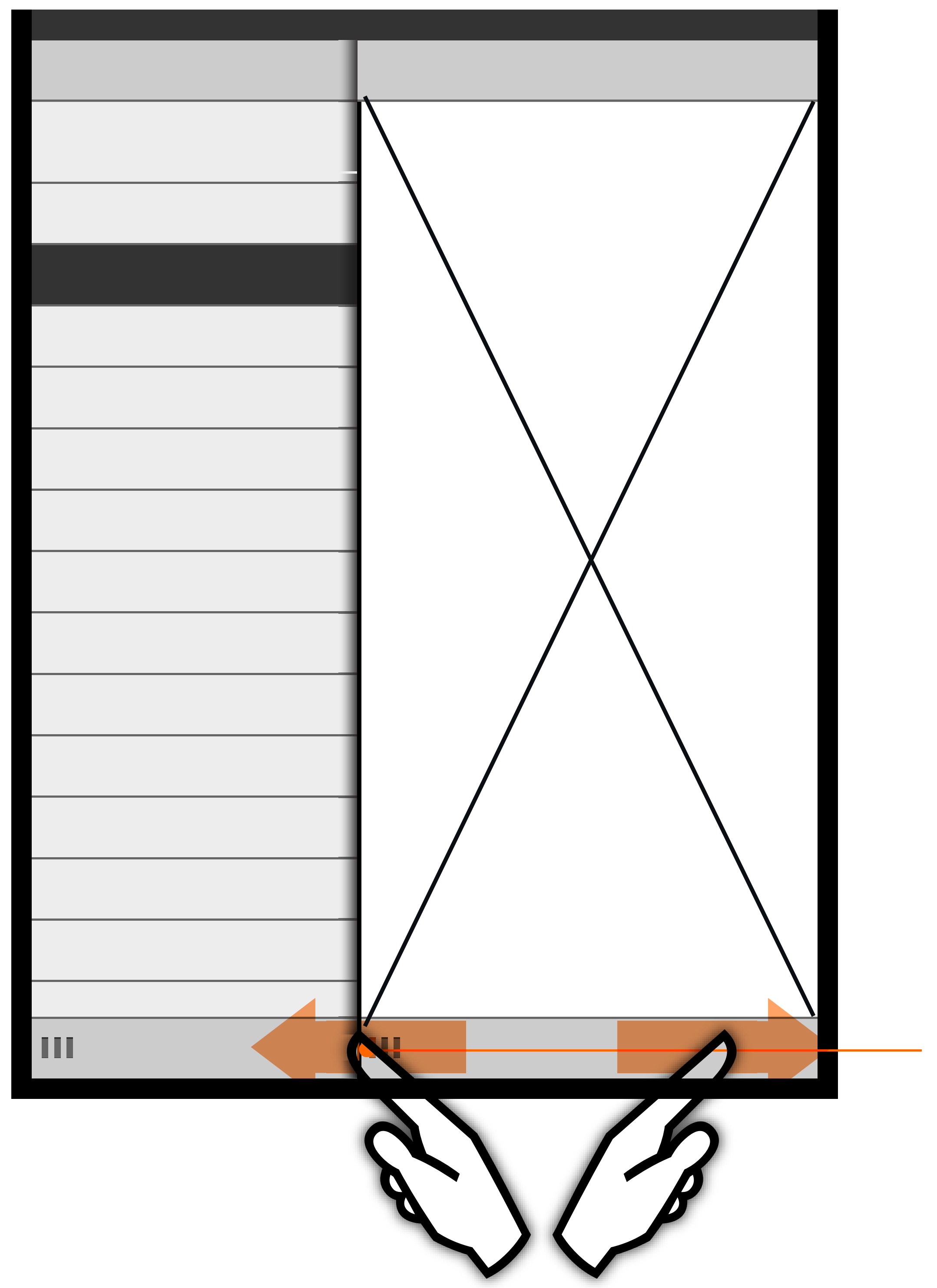
1. Transient header showing navigational information and functions
2. Always provide a means to exit full screen mode
3. Floating footer providing application controls

NAVIGATION PATTERNS

This section describes common methods for organizing the navigational structure of applications on webOS. Follow these patterns to ensure a consistent experience for your application.

NAVIGATION PATTERNS: HIERARCHICAL NAVIGATION

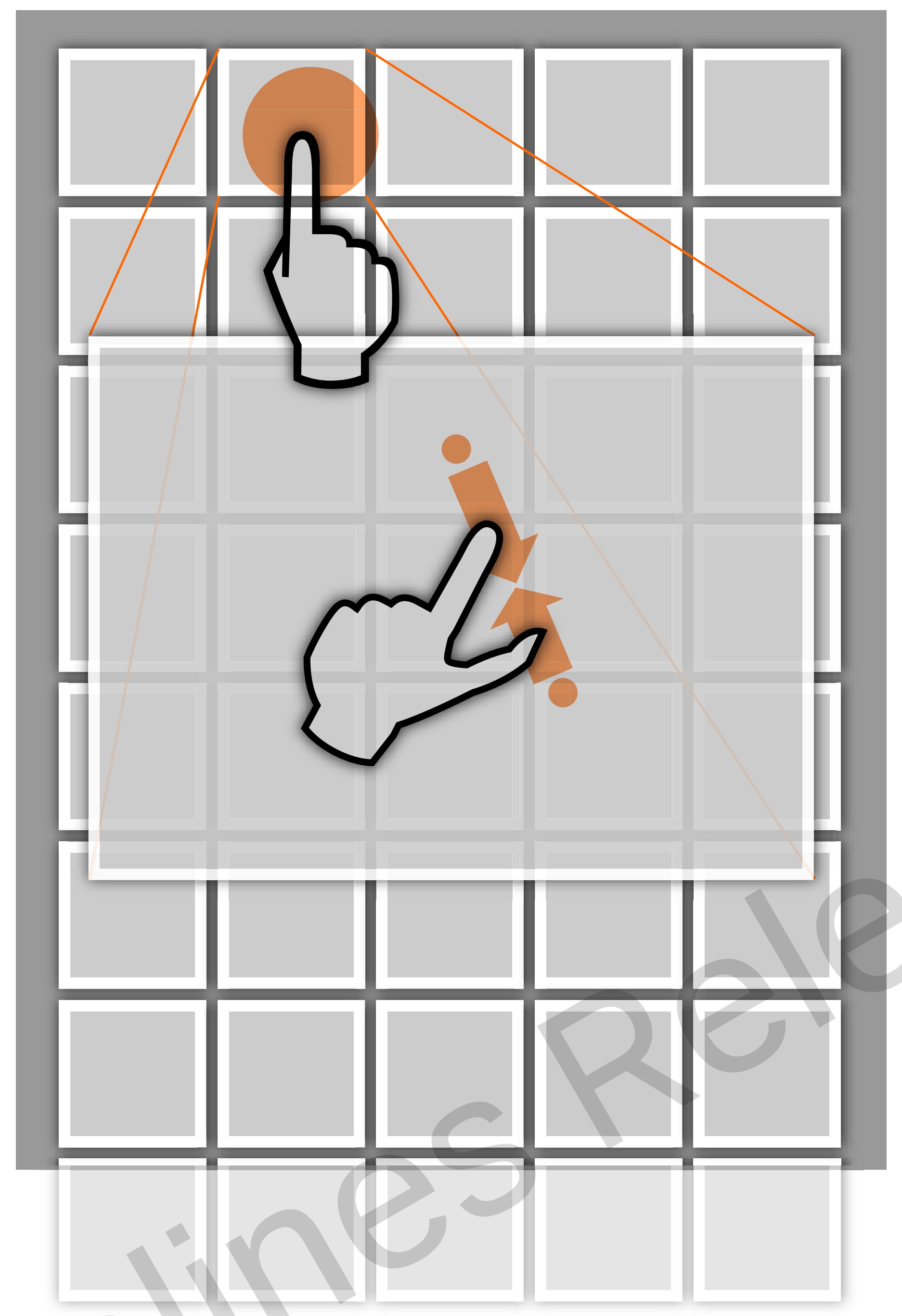
SLIDING PANES



DESCRIPTION

- Sliding Panes are an effective way to organize content hierarchically
- Panes can scale to multiple levels of information hierarchy and the user can navigate between levels intuitively by swiping panes to the left and right
- See the Sliding Panes layout section for further information

ZOOM IN / OUT



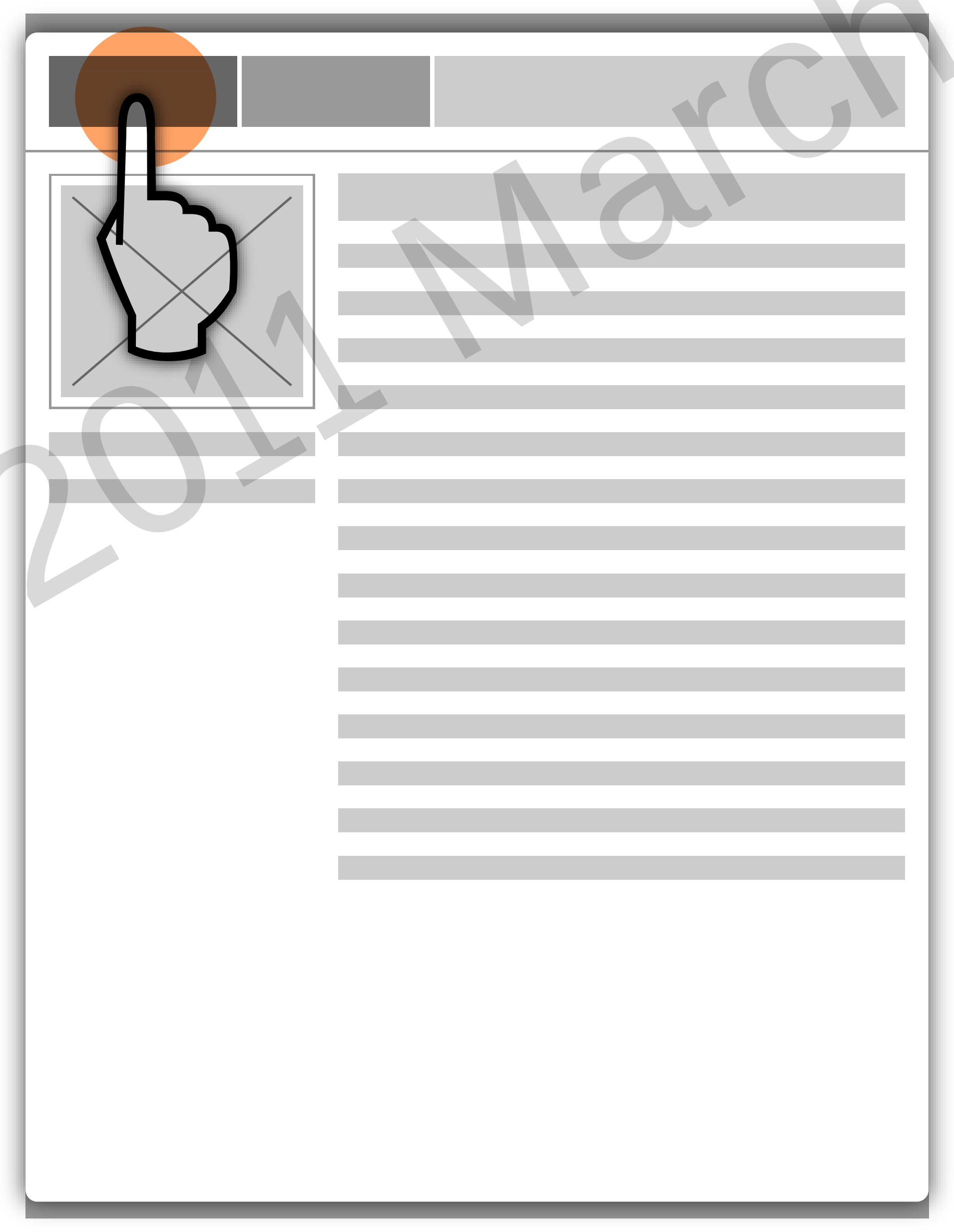
DESCRIPTION

- Tap on an item arranged in a grid or free form layout to zoom into it.
- Zooming can open the object in a new view or pop up a content window above the object grid.
- When drilling down to a new view, use the Back stepping pattern. When popping up a content window, the user can return to the grid by tapping outside the pop-up window

APP EXAMPLES

- Gallery: Photo album > Grid of photos > Photo
- Memos

BACK BUTTON

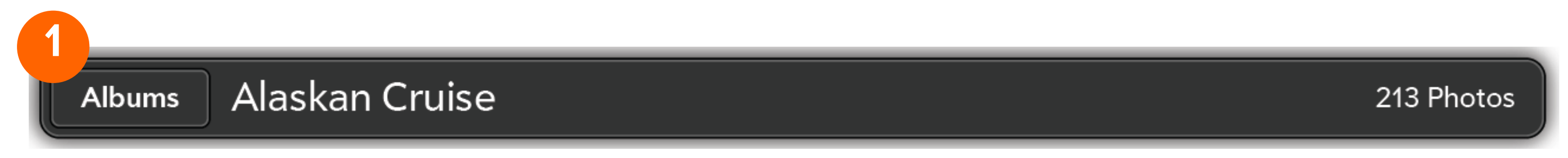


DESCRIPTION

- When drilling down a new view, always provide the user with a means to back step to the level from which they came
- The view should contain a button within the header of the view. The button should be labelled with the title of the previous view
- Tapping on a page transitions to a new full width page containing a button labeled with the previous page or category.

APP EXAMPLES

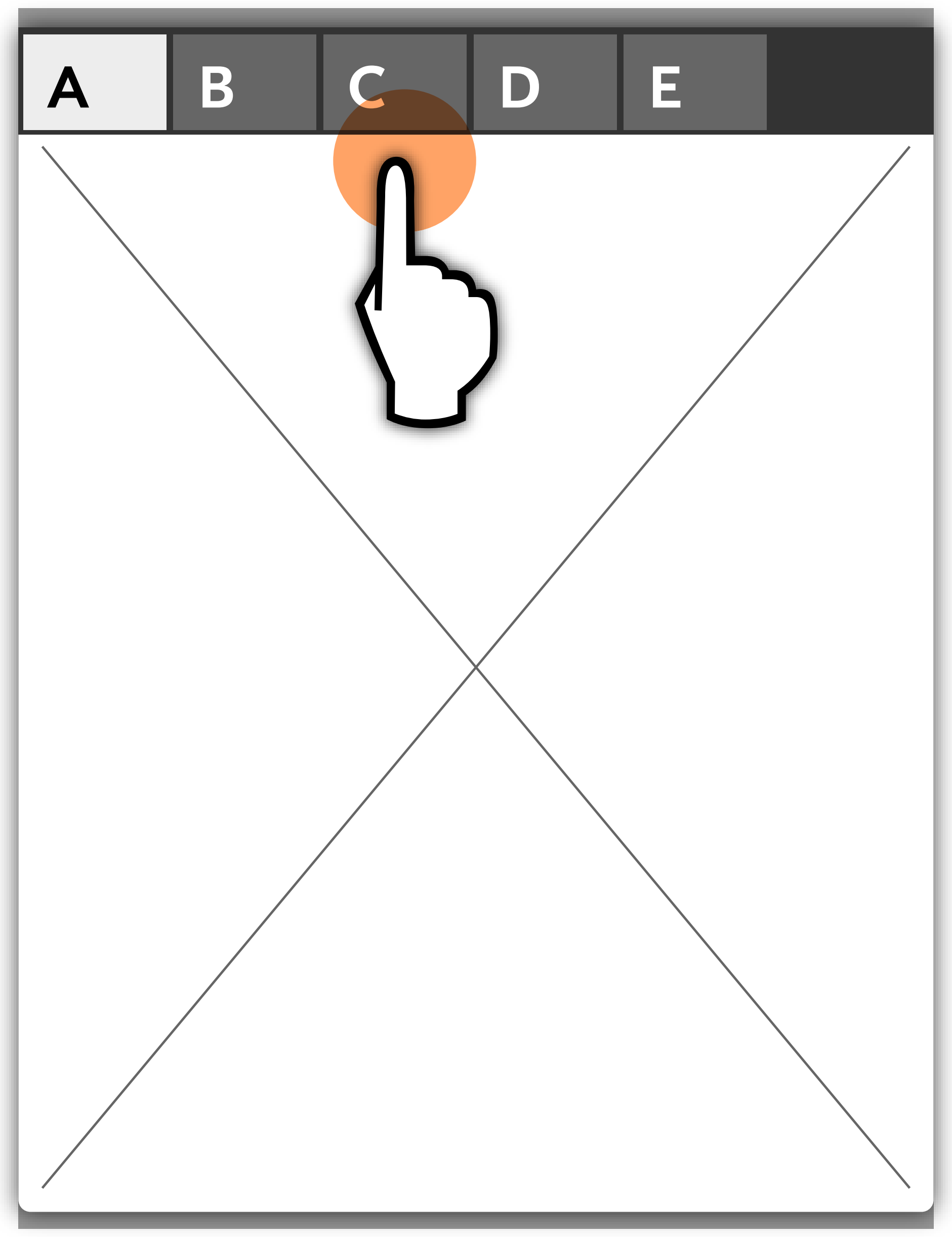
- Gallery: Photo album > Grid of photos > Photo



1 Back button labeled with title of previous view/category

NAVIGATION PATTERNS: NAVIGATING AT THE SAME LEVEL

TABS FOR NAVIGATING SECTIONS



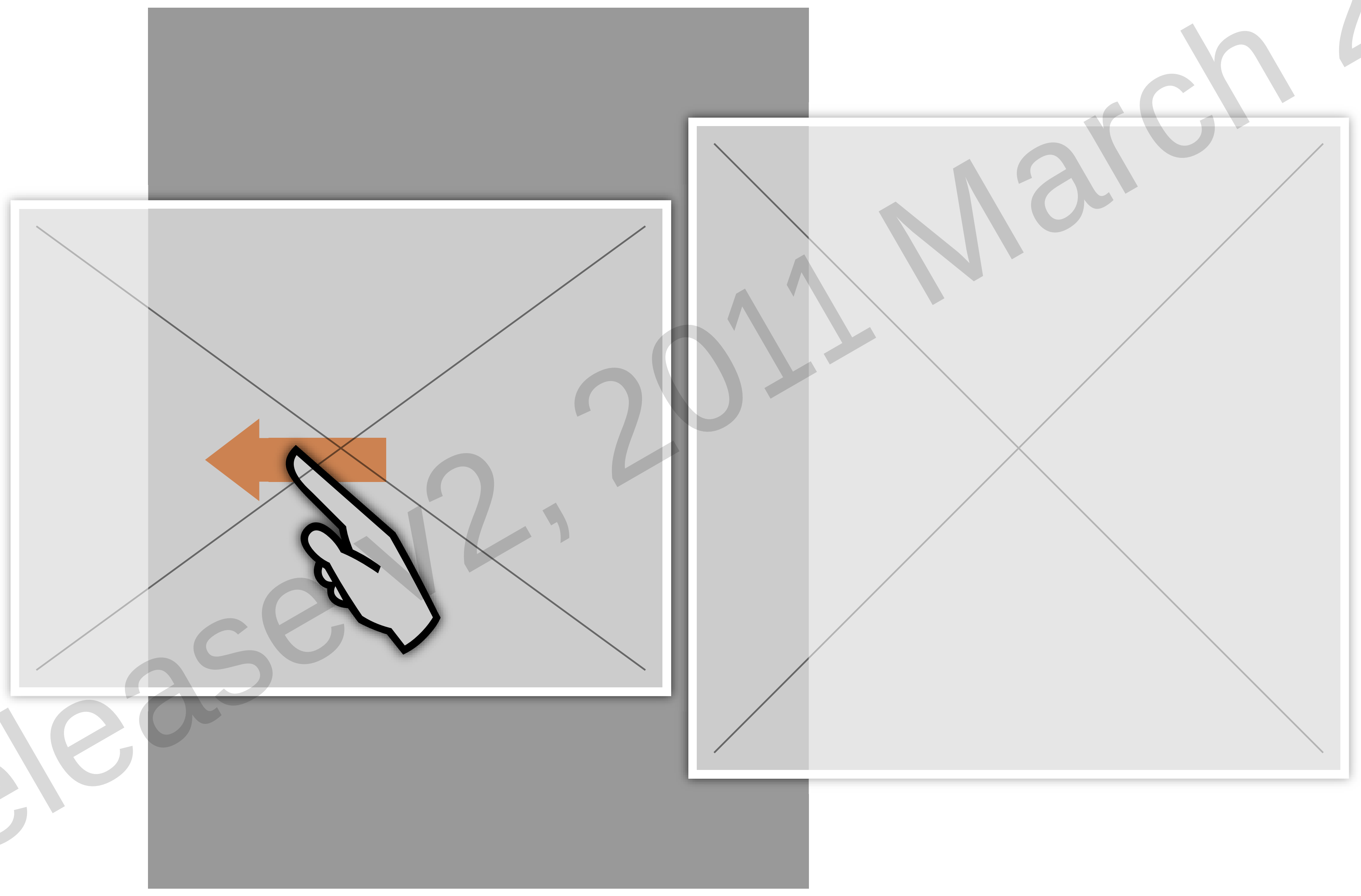
DESCRIPTION

- Tabs provide the user with quick access to different sections of content within an application
- Tabs are generally displayed within the header of a pane; actions associated with the content are generally displayed in the content area or within a tool bar in the footer of the pane

APP EXAMPLES

- Launcher

PAN CANVAS



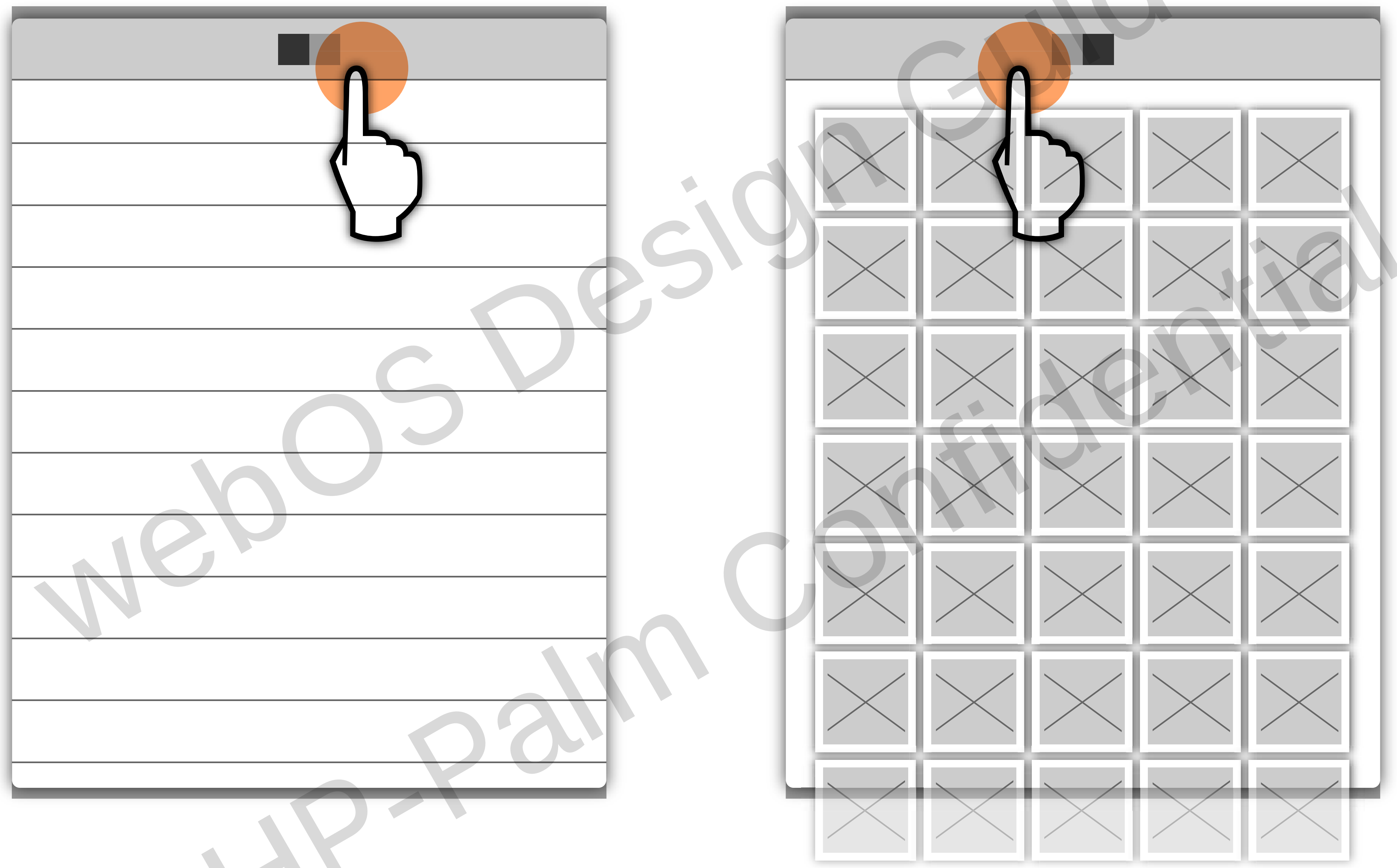
DESCRIPTION

- Flick a full screen object sideways to reveal the next or previous object

APP EXAMPLES

- Photos: Moving from photo #1 to #2

VIEW-SWITCHING



DESCRIPTION

- Button groups can be used within a header to switch between different views
- View-switching is different from using tabs in that generally the same content is being viewed when switching views, but the presentation of the content is changing

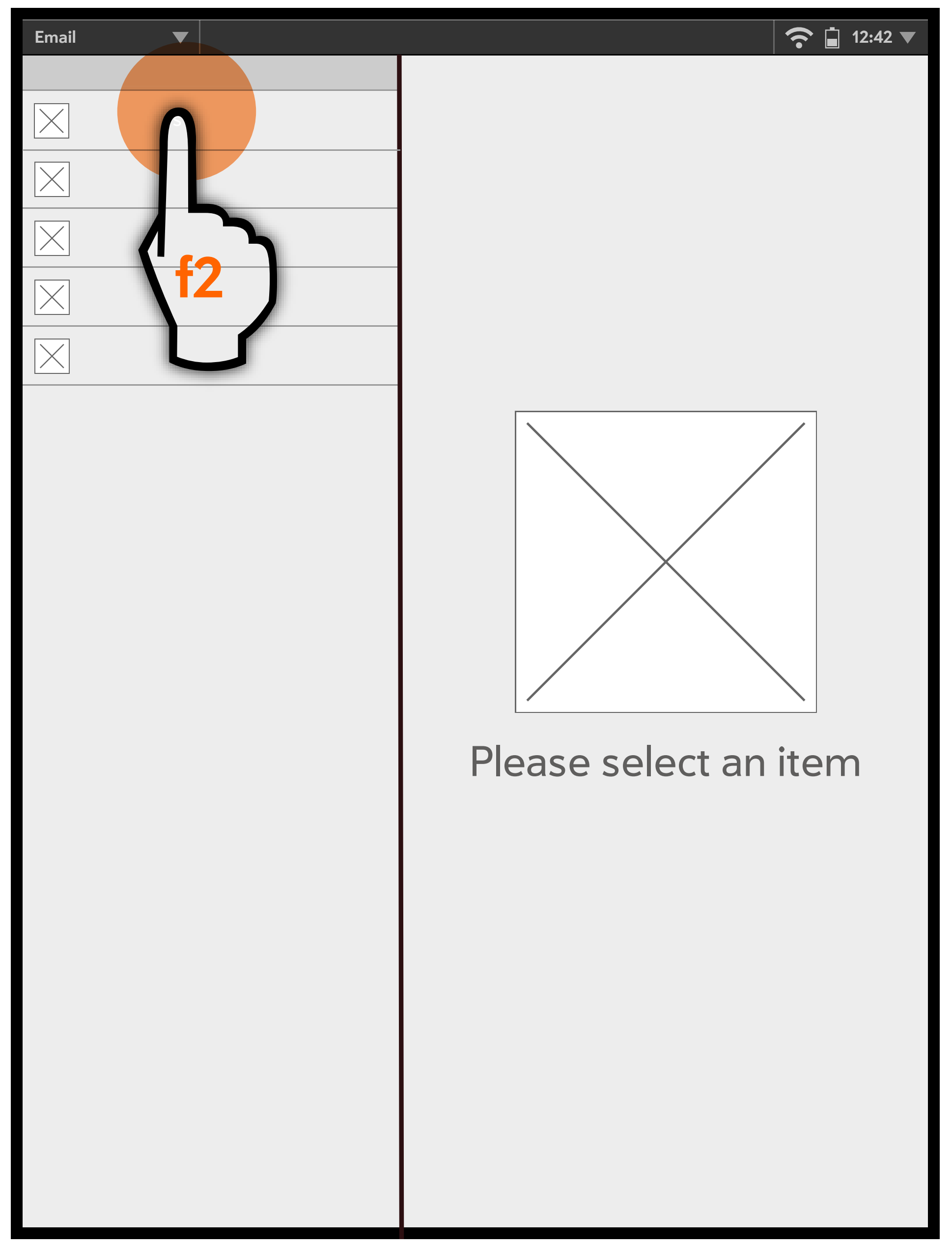
APP EXAMPLES

- Day vs week vs month views
- List of books vs grid of book covers

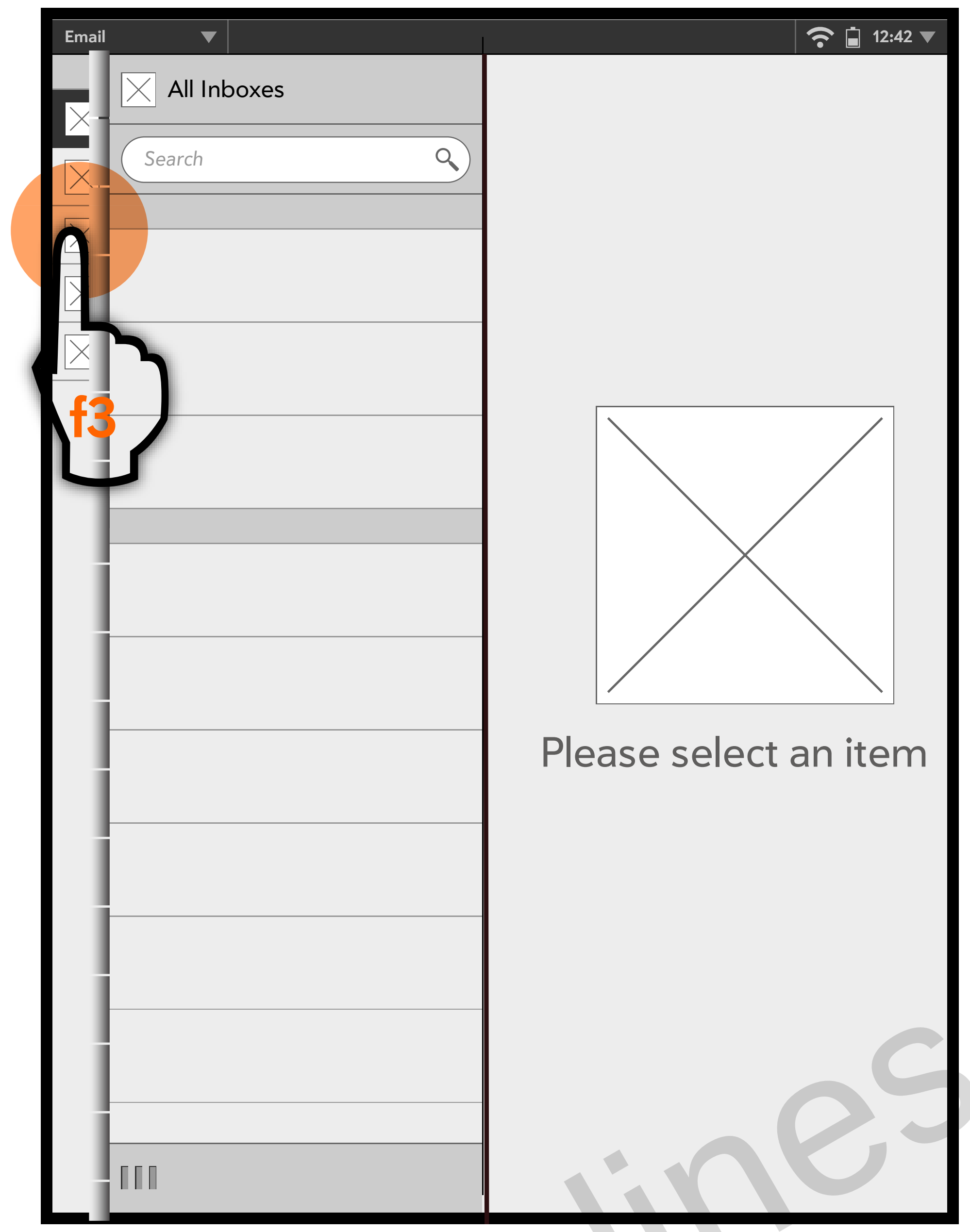
NAVIGATION PATTERNS: COMBINING HIERARCHICAL AND SIDEWAYS NAVIGATION

PANE PEEKING

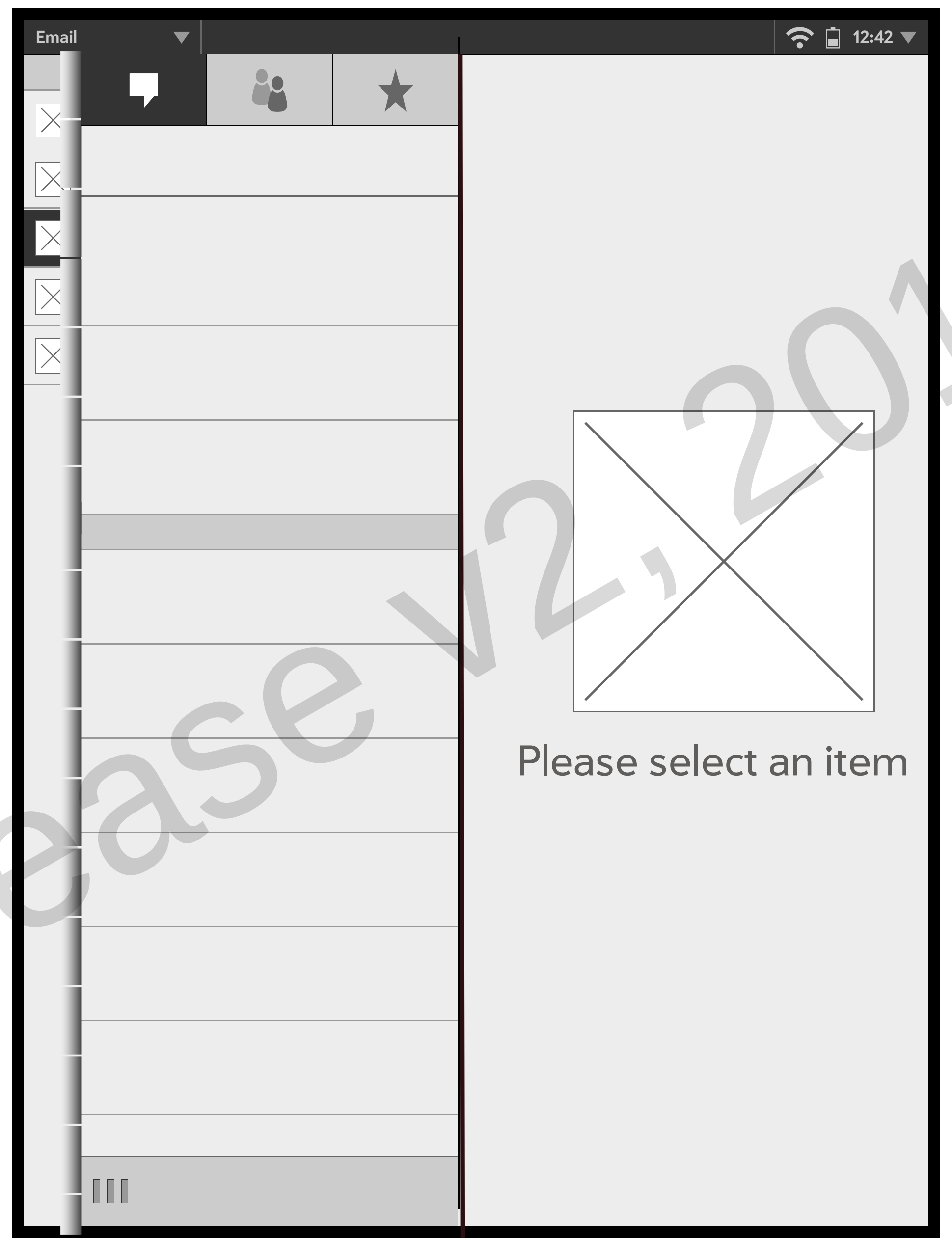
f1



f2



f3

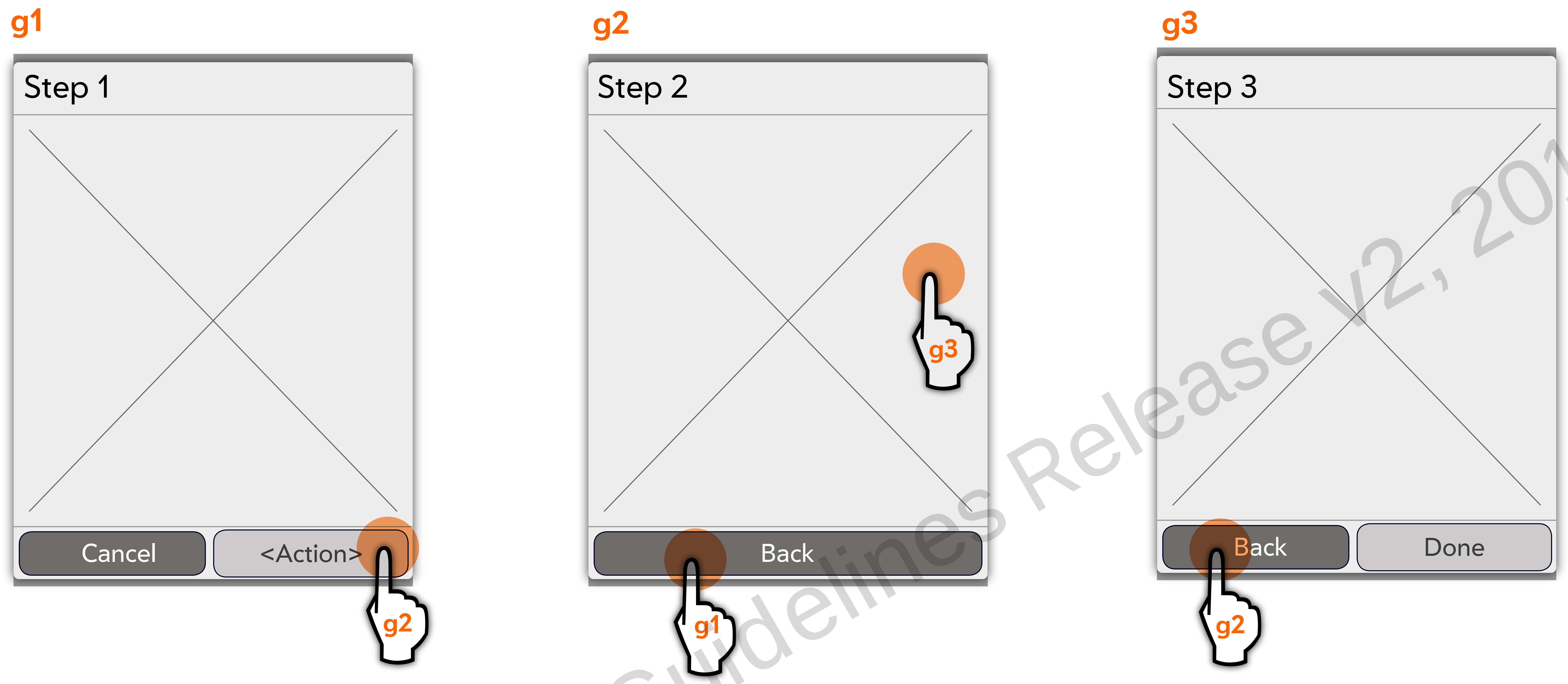


DESCRIPTION

- Pane peeking maintains a portion of the top level in view while drilling down through an information hierarchy
- At any time the user may tap on the top level icons to navigate between distinct sections of the application
- This pattern therefore allows the user to both drill down through the structure and navigate sideways between application sections

NAVIGATION PATTERNS: TASK FLOW NAVIGATION

STEP BY STEP / TASK FLOWS



DESCRIPTION

- When creating a constrained task flow that guides the user through a linear series of steps, use a Task Flow navigation pattern.
- Following a Task Flow is different than drilling through a hierarchy in that the user is not following a series of content categories, but instead following a set of steps in a task
- In this pattern, there are one or more actions to advance or complete the task flow, and a button to back step through the process or to cancel the process.
- An action button may optionally be used to advance the task flow. Alternatively, the user may advance the flow by interacting with the content area.
- A 'Done' button may optionally be used to complete the flow and save changes. Alternatively, the user may complete the flow by interacting with the content area.
- A 'Back' button should be used to back step through the flow.
- A 'Cancel' button should be used to cancel the flow without saving changes. There should be a 'Cancel' button shown at the start of the task flow.
- All buttons should all be placed in the footer area of the Pane or Interactive Pop-up. The 'Back'/'Cancel' buttons should always be placed at the left/ bottom of the footer area.
- The action buttons and 'Done' buttons should be placed on the right/top of the footer area.

THANK YOU!