HP TOUCHPAD™

webOS™ Design Guidelines

Guidelines version

2011 JUNE 20

OVERVIEW

These guidelines help HP webOS app designers and developers create apps for the HP webOS TouchPad. If you are a current webOS developer, start thinking about how you might adapt your current webOS app design or create a new app for the TouchPad. These are general design recommendations and suggested starting points for your design.

Following these patterns helps your app achieve consistency and familiarity for your app on the webOS platform. Please use this document in conjunction with the wireframe stencils in the document HP TouchPad Wireframe Stencil. The stencils enable you to create wireframes for your app design and provide further detail about webOS.

IN THIS DOCUMENT

This document has five key sections that focus on the following:

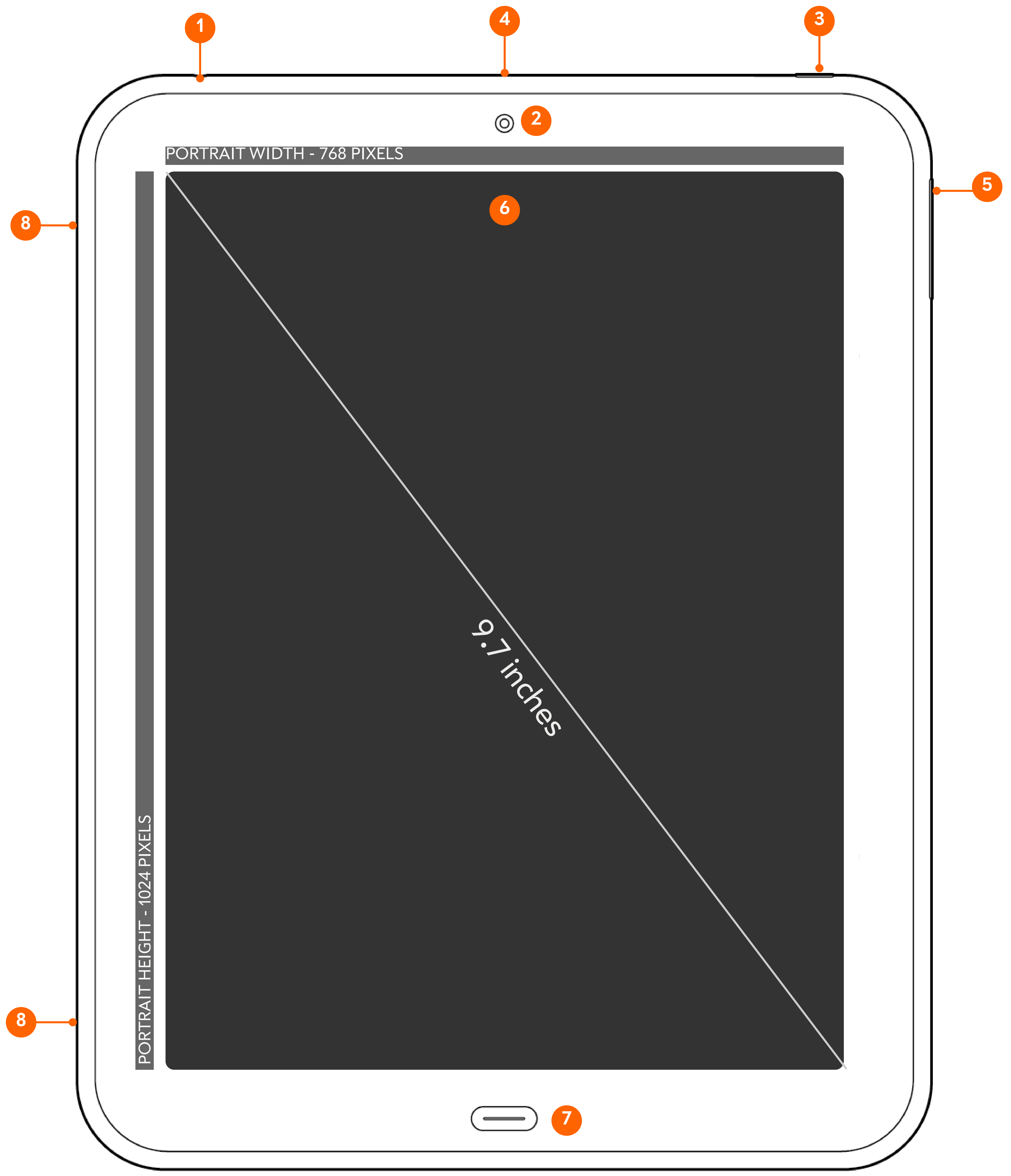
- “Hardware Features” describes the key HP TouchPad features that deserve consideration when creating your app.
- “System Features”highlights the webOS framework elements that you can integrate into your app.
- “App Framework” describes the basic building blocks that can be part of your app.
- “Designing Your App” looks at the different ways to lay out your app, including using panes and other framework components.
- “Navigation Patterns” explains the common navigational patterns to use in your app.

CONTENTS

HARDWARE FEATURES

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

KEY HARDWARE FEATURES



DISPLAY CHARACTERISTICS

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

HARDWARE COMPONENTS

The HP TouchPad has the following hardware components:

1. 3.5 millimeter (mm) headset jack
2. 1.3 megapixel front-facing camera
3. Power button
4. Microphone
5. Volume controls
6. Capacitive touchscreen
7. Center button
8. Stereo speakers

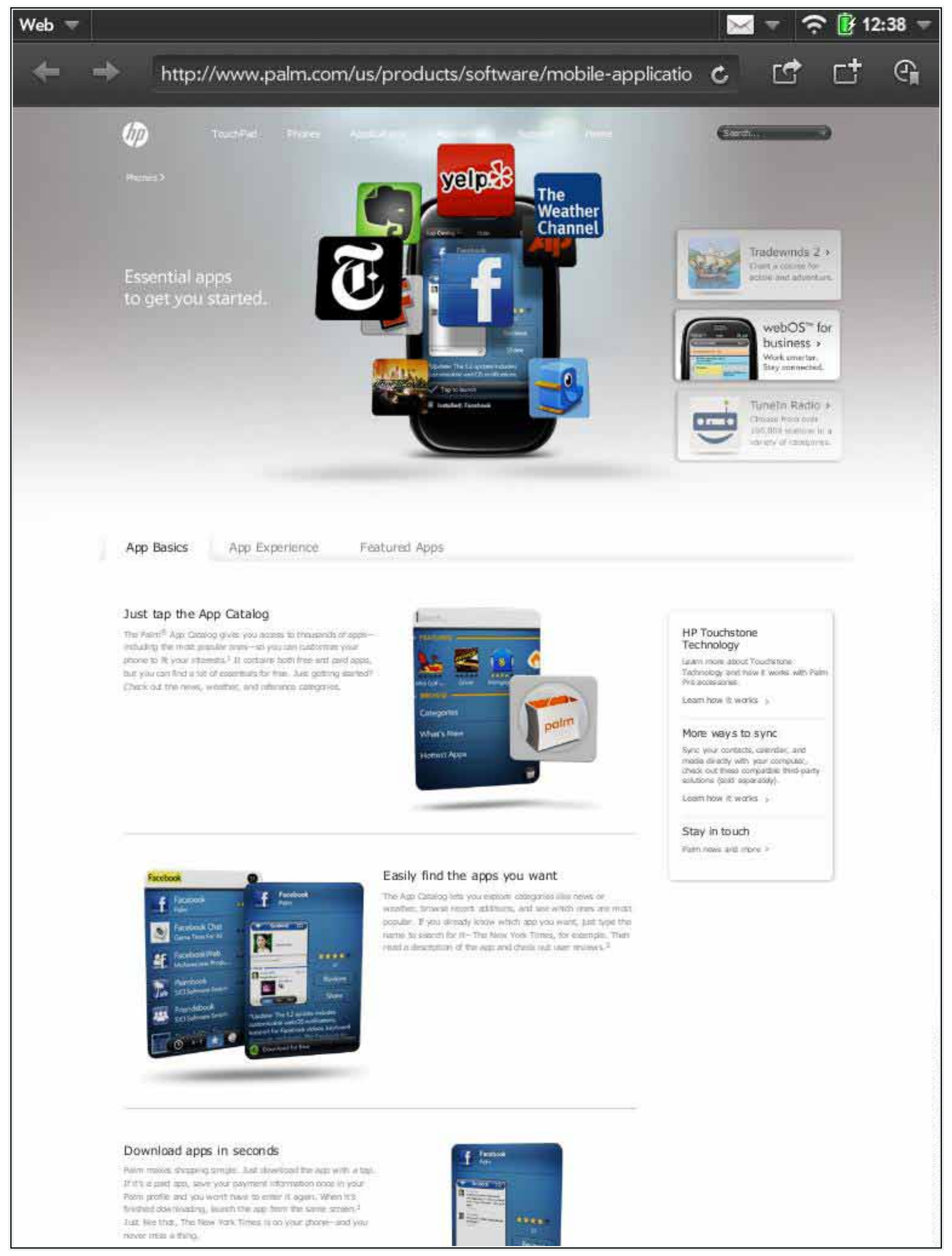
GESTURES

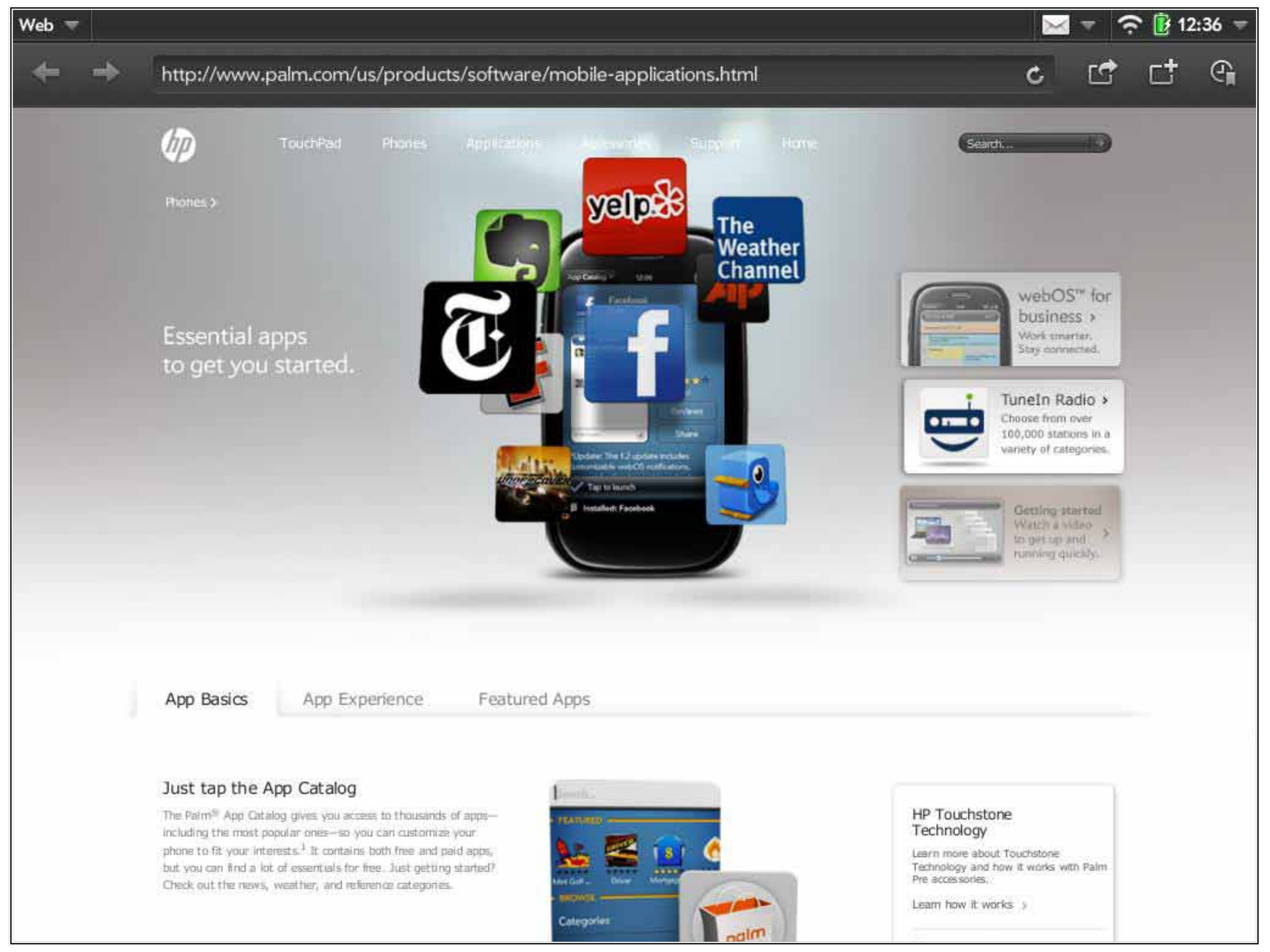
Swiping up from the bottom of the screen in any device orientation takes the user to the device's card view. (See *Card View and Multitasking* section.) If the user is already in the card view, the swipe toggles the *App Launcher* on and off. The swipe from the bottom has the same action as pressing on the device's Center button.

The capacitive touchscreen on the device is also able to detect various touch events, supporting the direct manipulation of the user interface through a variety of gestures. See the *Gesture Framework* section for more details.

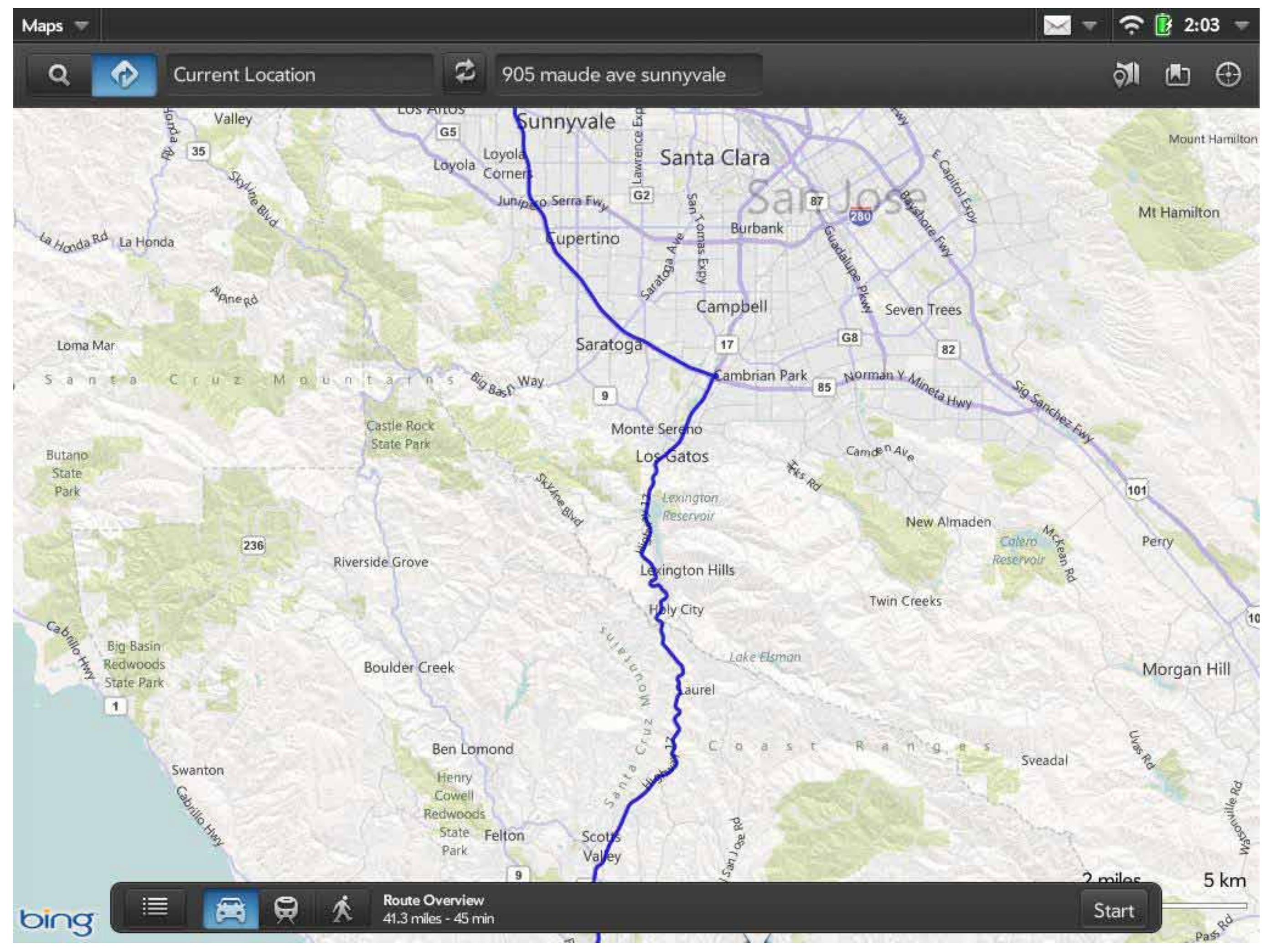
SENSORS

BROWSER IN PORTRAIT AND LANDSCAPE ORIENTATIONS


Web
12:38
http://www.palm.com/us/products/software/mobile-applicatio
Essential apps to get you started.
App Basics
App Experience
Featured Apps
Just tap the App Catalog
Easily find the apps you want
Download apps in seconds


Web
12:36
http://www.palm.com/us/products/software/mobile-applications.html
Essential apps to get you started.
App Basics
App Experience
Featured Apps
Just tap the App Catalog
HP Touchstone Technology

MAPS APP



ORIENTATION

The HP TouchPad includes sensors that detect the device's orientation. The on-device accelerometer can detect the device's orientation to enable apps to rotate the display between portrait (vertical) versus landscape (horizontal) orientations. When a user moves, rotates, or tilts the device, the sensor reports the change, so your app can respond to it.

The device accelerometer also provides detailed orientation information that is accurate and fast enough to support gaming experiences. Gaming apps like flight simulator or racing games, can use the accelerometer data to provide accurate gaming experiences.

LIGHTING

The light sensor and software gracefully and automatically dims and brightens the display as ambient light changes. The TouchPad adapts to indoor or outdoor light seamlessly.

LOCATION

The device location service can determine the device's current location using either the GPS sensor or Wi-Fi, depending on the device model. With the location information, your app can provide either one-shot or real-time position tracking, similar to apps like Google Maps or Loopt.

NOTE: Users must have enabled location services on the device during first-time use or through the device Location preferences.

INPUT AND OUTPUT

CAMERA

Apps can capture data from the device's front-facing 1.3 megapixel camera to save both still images and videos. An app can specify the location in which the captured data is saved.

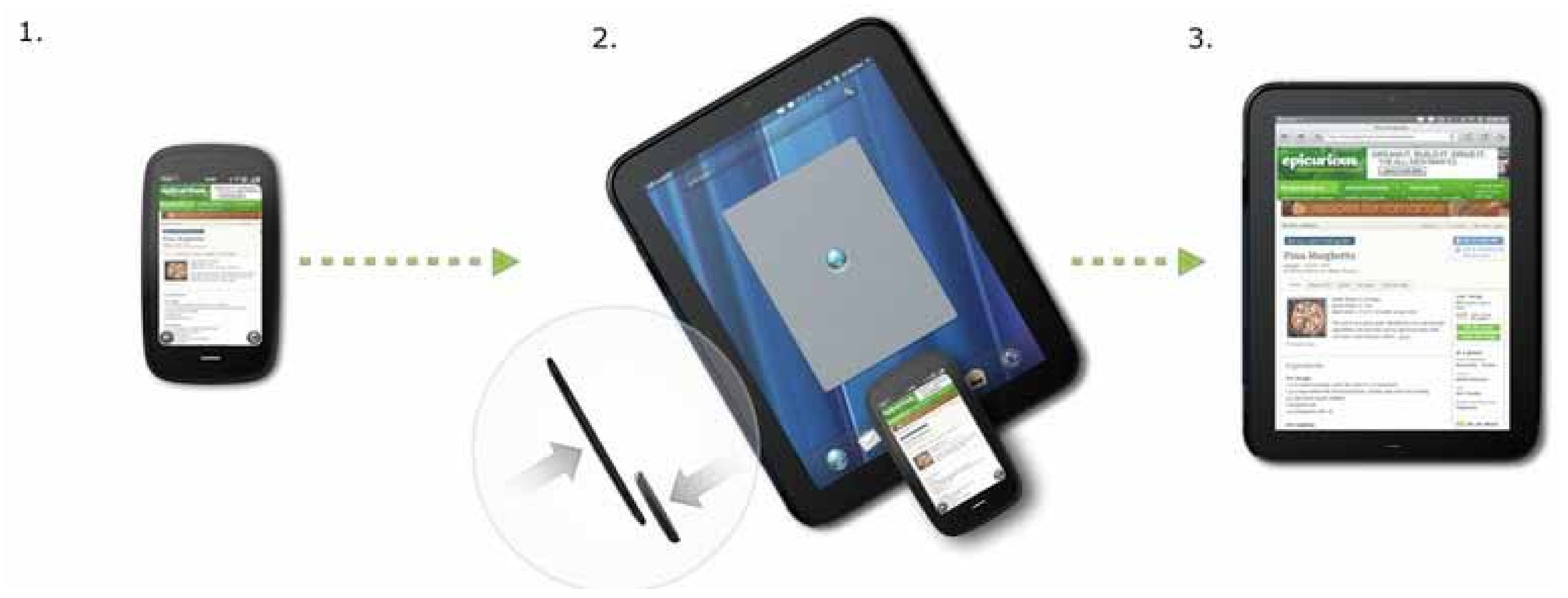
HP BEATS AUDIO™

HP Beats Audio technology, which is built-in to HP laptop computers, is also supported on the TouchPad. HP Beats Audio produces high-quality digital sound output. App developers can take advantage of HP Beats Audio when users are listening to music, streaming video, or playing games. If it sounds better, then the user experience is better.

TOUCH TO SHARE

Touch to Share for tablet and other HP smart devices enables the TouchPad to pair with webOS smartphone devices to transfer text messages, phone calls, and URLs. After pairing, text messages or calls received on the paired webOS smartphone can be displayed on the TouchPad for a totally seamless experience across devices.

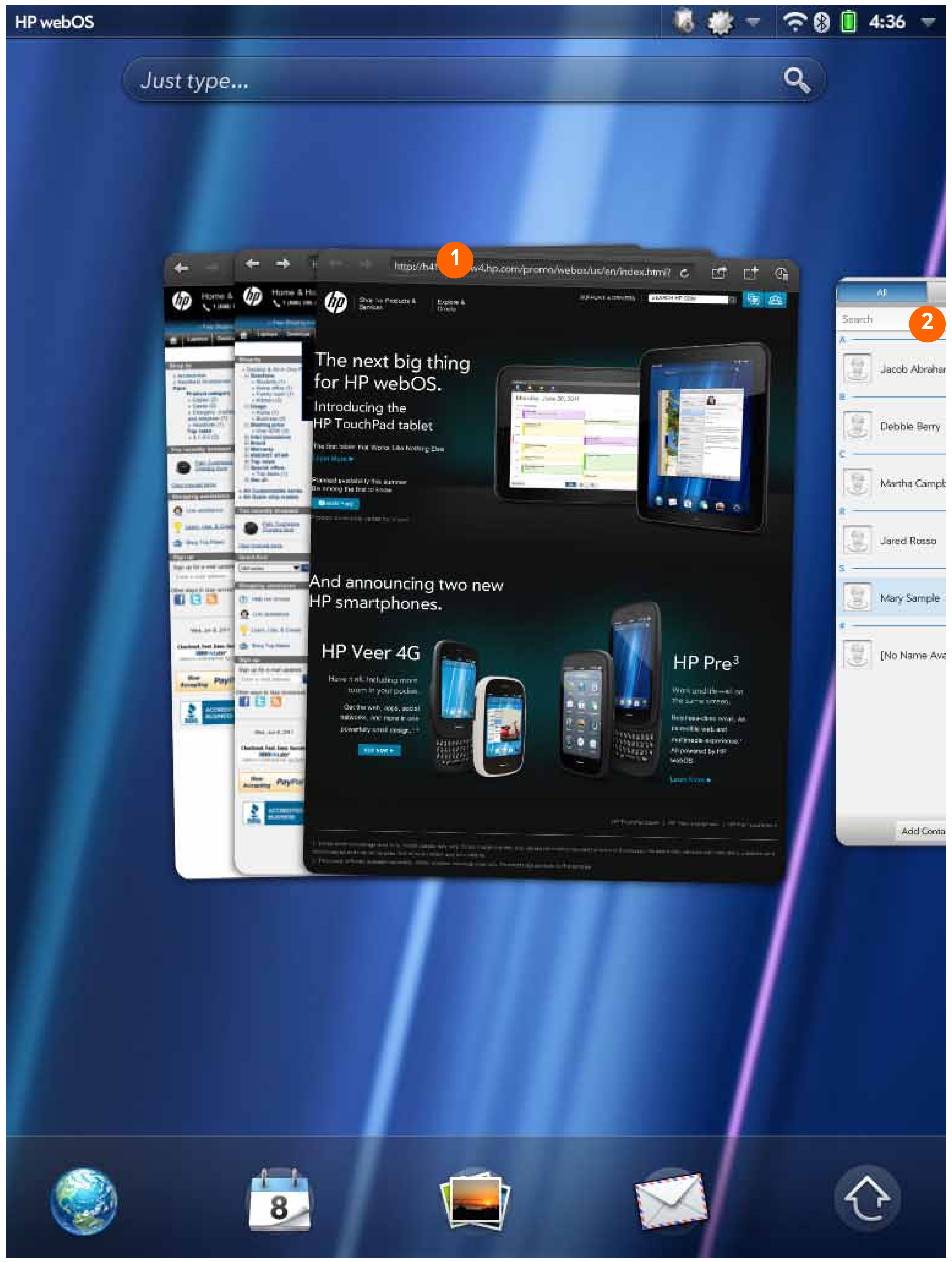
Touch to Share can also be used to transfer URLs between paired devices by simply touching them together. Touch to Share currently only supports the transfer of URLs for the TouchPad, but the API is being extended to support the transfer of other data types.



SYSTEM FEATURES

This section describes the key system features of webOS that your app can utilize and integrate with. These features allow you to enhance your apps by leveraging the power of webOS.

CARD VIEW AND MULTITASKING



1 Card Stack

2 App 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 app 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.

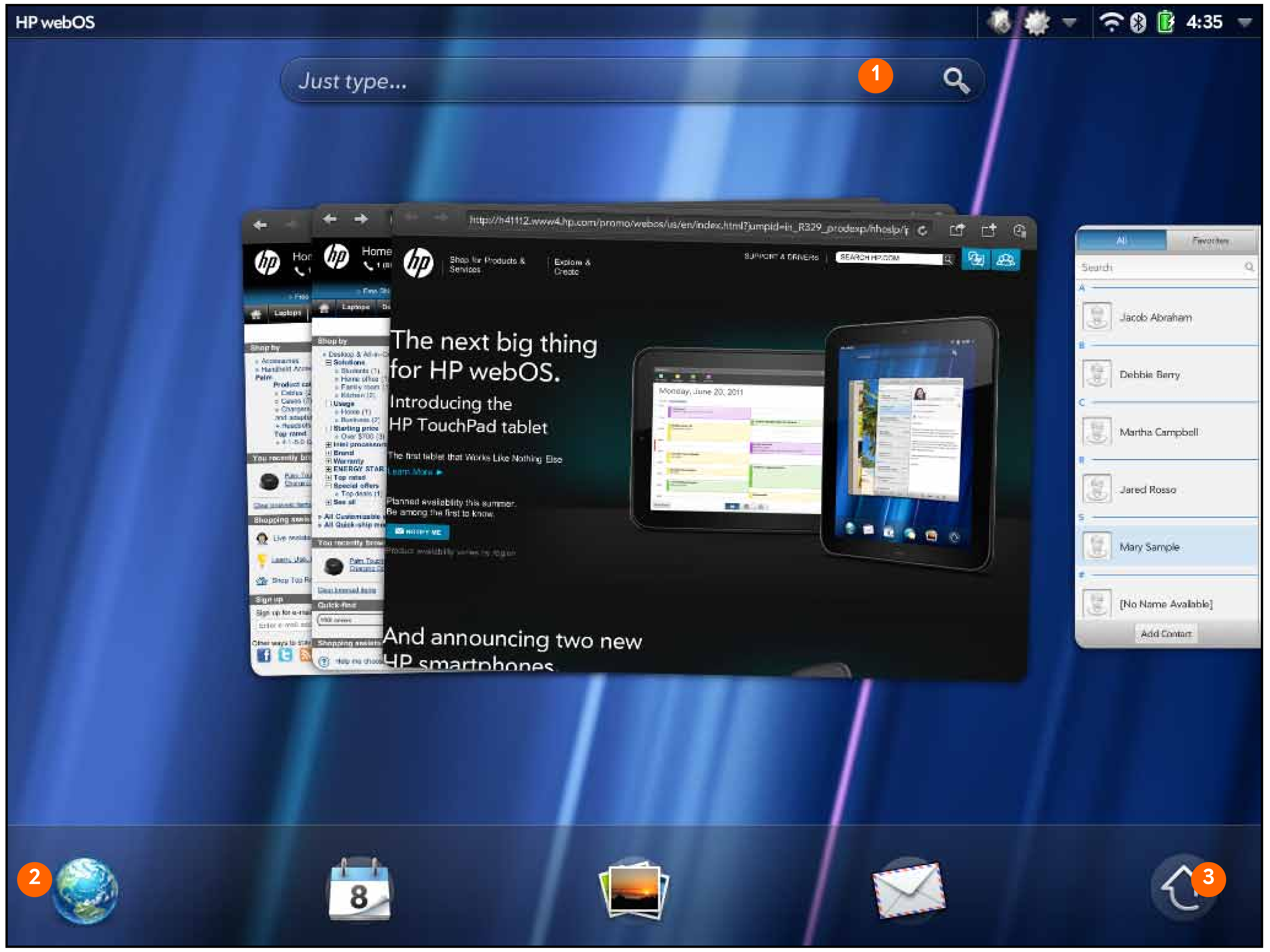
Cards appear as scaled down versions of the full sized apps. Cards will continue to update in card view to reflect the latest state of the activity in the app. If a user navigates away from a card and returns to it later, *it should remain in the latest state* so that the user can continue where they left off.

CARDS ARE ACTIVITIES, NOT APPS

Cards are not the same as apps. An app can create several card instances in the card view. Each of these cards can represent a concurrent activity within the app. For example, the email app creates a new card for each instance of an email composer. All the cards created by an app are associated with each other automatically. Associated cards are shown grouped in the card view in **card stacks.**

When planning your app structure and workflow, consider carefully whether to open a new view within a new card or whether to keep the interaction within a single card. Creating multiple cards can complicate the user experience of your app. However, if there is good reason to open a new card and it purposefully supports the user's workflow, then do so. There should be a genuine user benefit in running the different activities in your app concurrently.

CARD VIEW AND MULTITASKING



1 Just Type Search Field

2 Quick Launch bar

3 App Launcher icon

USER ARRANGEMENT

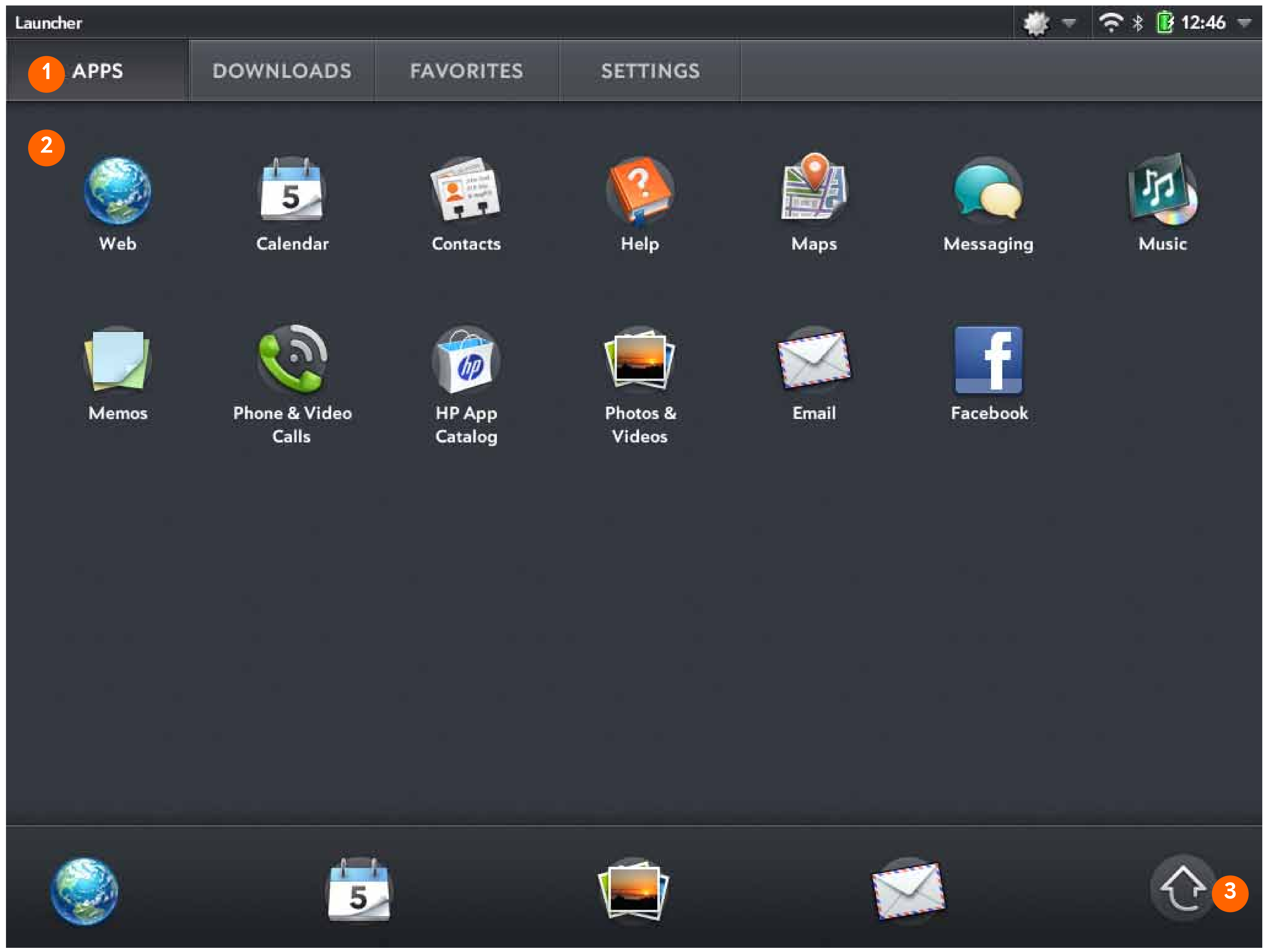
Ultimately, the user has control over how the cards in the card view are organized. By pressing and holding on a card, the user can drag and drop the cards to change the order in which the appear. Users can also re-arrange cards within a stack, or pull a card from a stack and show it individually. Conversely,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

Users can also dismiss cards by swiping them up off the screen. If any data was being entered or edited in the card - e.g., in a form or an email, it should *automatically be saved* when the card is dismissed.

APP LAUNCHER AND JUST TYPE

The card view is also used as a springboard to other system features. Just Type (see *Integrate Your App With Just Type*) is accessed by tapping on the search field at the top of the display and apps can be launched by tapping on the app shortcuts in the Quick Launch bar. The user can also access the *App Launcher* (see next page) by tapping the App Launcher icon in the Quick Launch bar.

APP LAUNCHER



1. View tabs
2. App icon grid
3. Launcher icon

The App Launcher lets users launch all apps on the device. Each app is represented in the App Launcher by an icon within a grid view. Tapping the app icon launches the app inside a new card.

The App Launcher contains four tabs; each with a grid of apps:

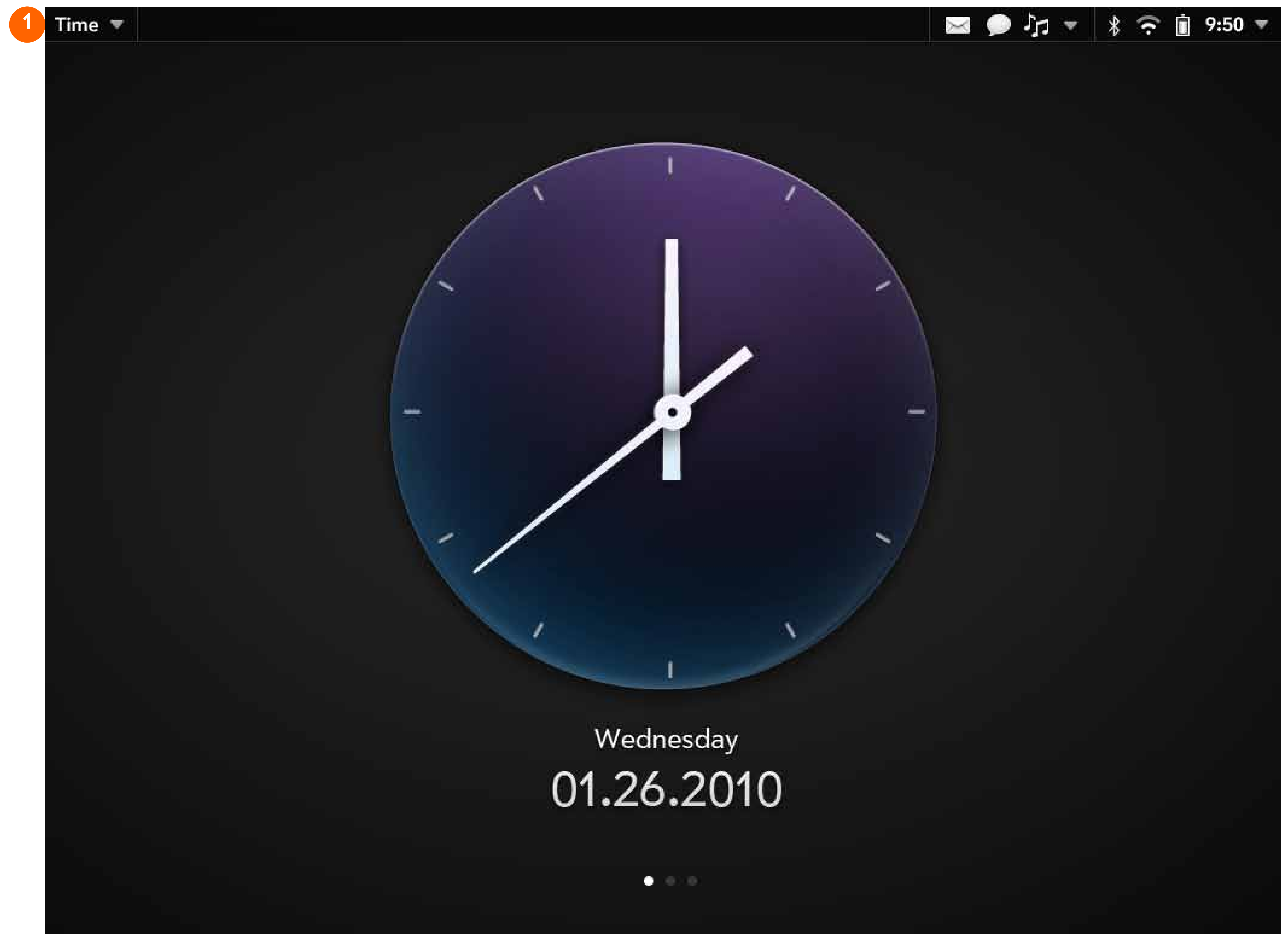
- APPS displays the native built-in apps that come with the device, such as Contacts, Memos, and Calendar.
- DOWNLOADS displays apps downloaded from App Catalog.
- FAVORITES shows apps selected as favorites by the user.
- SETTINGS is the System apps, such as VPN, Wi-Fi, Bluetooth®, and Accounts.

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

MANAGING APPS

The App Launcher also provides a range of functions for managing apps. Users can enter an Edit mode by performing a Press and Hold on any app icon. In this mode, users can re-order icons in the grid, delete apps, and move apps between tabs by dragging them from one to another.

EXHIBITION MODE



1 App 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 App Launcher. Apps can be written to run in Exhibition mode, either as an extension to your existing app, or as stand-alone Exhibition mode apps, allowing you to create all-new app experiences.

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

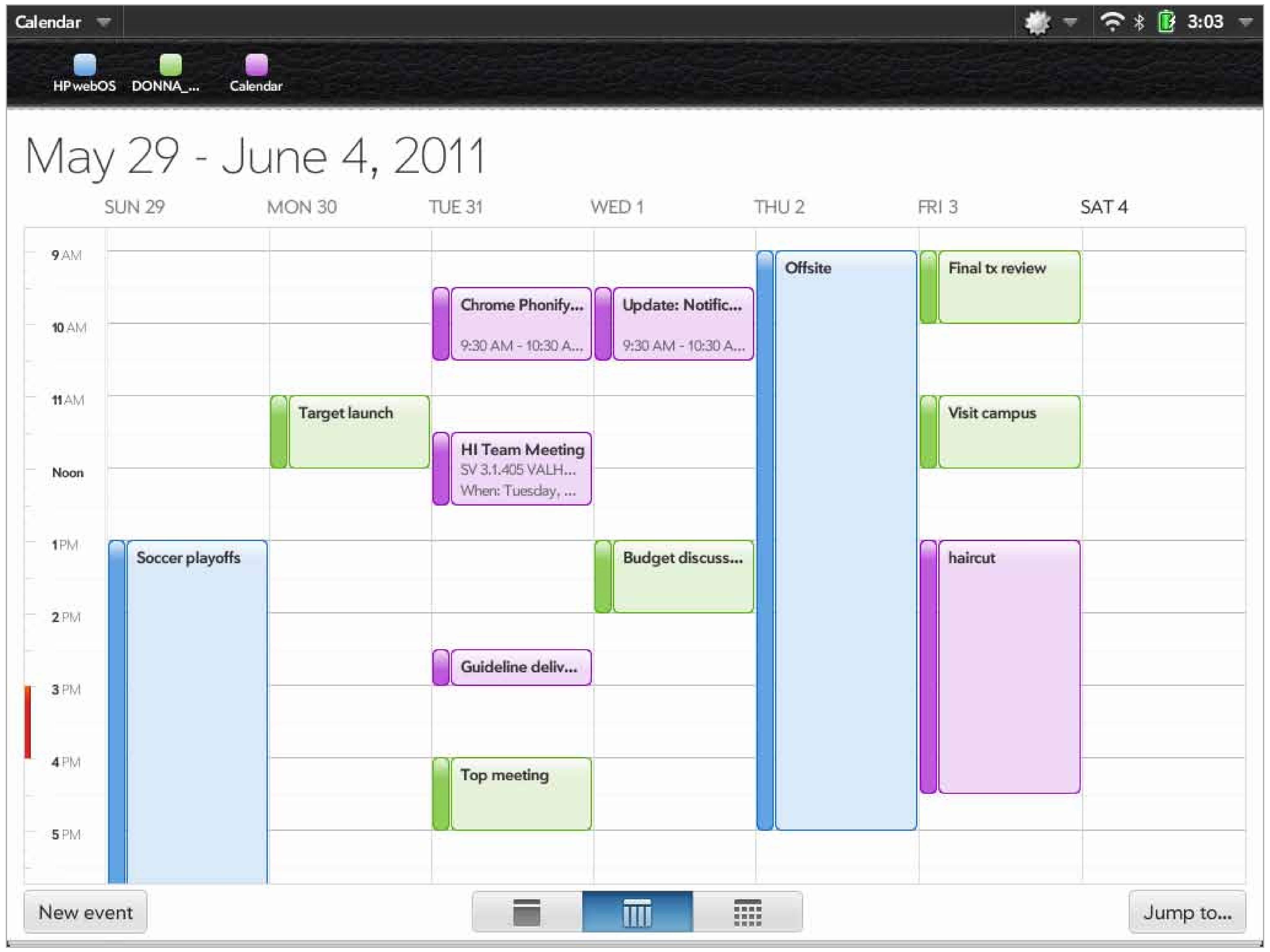
FULLY FUNCTIONAL

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

SYNERGY™



INTEGRATE DATA INTO DEVICE APPS

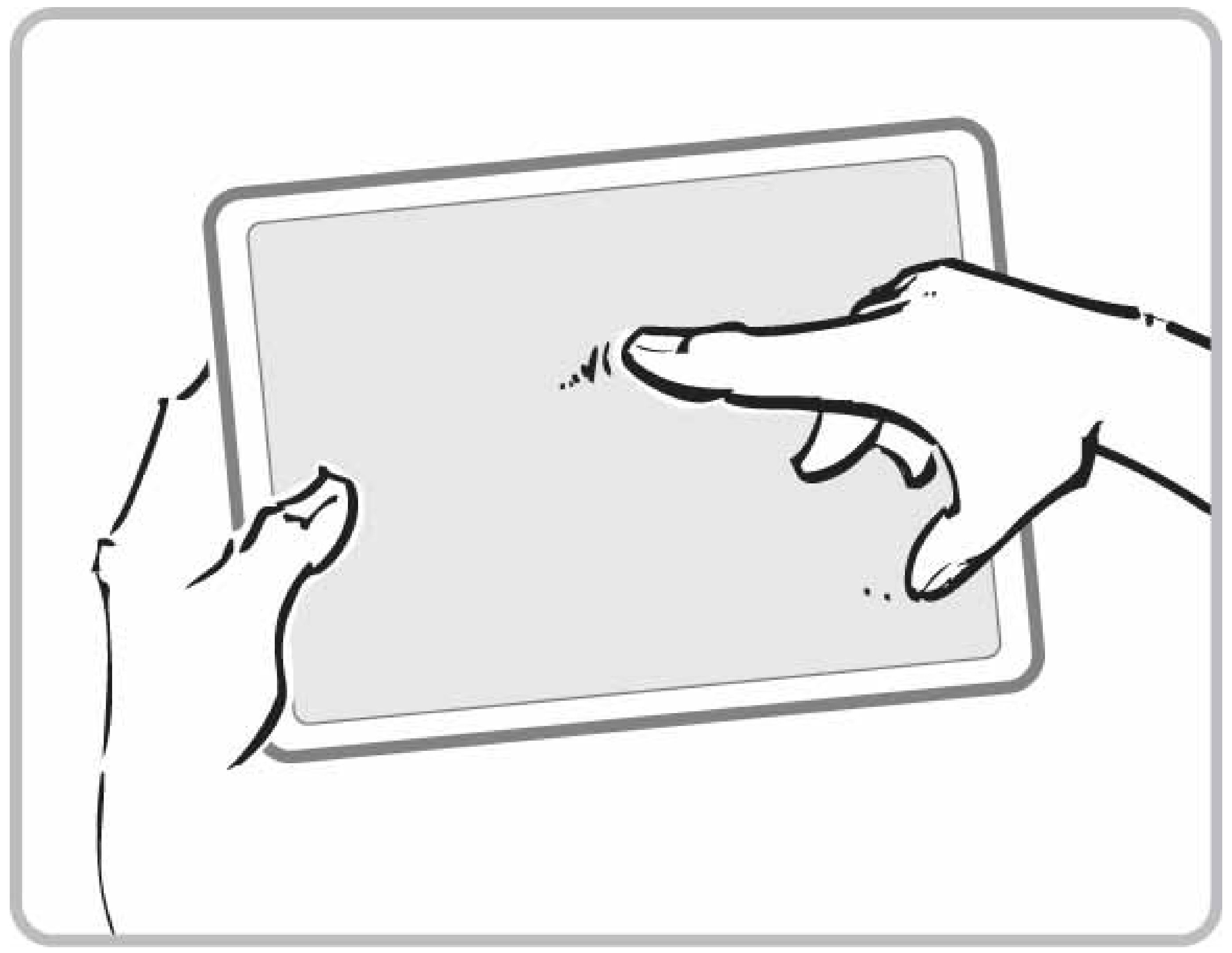
With HP webOS, you can create a Synergy connector that allows apps to sync data directly into device apps, including the Contacts, Calendar and Messaging apps.

HP Synergy is a powerful method that allows your app to integrate and be discoverable directly from the device experience. In Calendar, messages and appointments can be synced from a user's service and shown color coded alongside other Calendar data.

In Contacts, contact data from a user's service can be synced and integrated into the address book. If contacts with the same name are found in the address book, the contacts' data from the service is integrated into the same contact card.

In these ways, HP Synergy seamlessly integrates data from multiple sources into a **single, coherent** experience for the user.

GESTURE FRAMEWORK



GESTURE FRAMEWORK

The capacitive display on the HP TouchPad is a powerful sensor for capturing interactions made with the user's fingers. HP webOS provides support for the direct manipulation of UI elements on the screen using the user's fingers through its gesture framework.

TOUCH EVENTS AND GESTURE SUPPORT

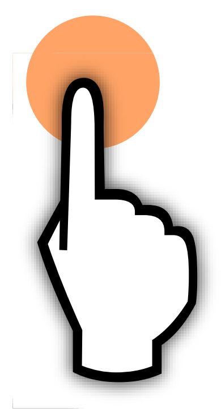
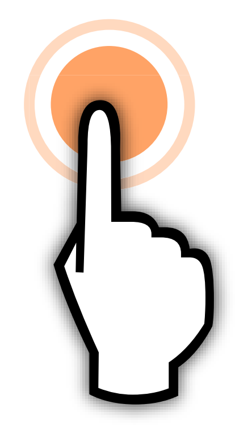
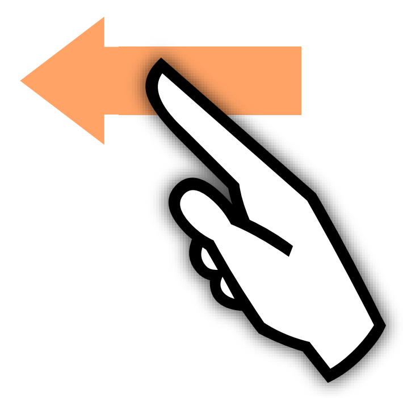
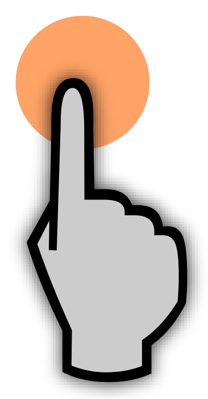
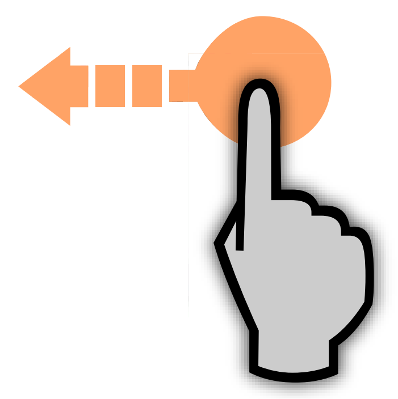
Any UI element on the screen can detect the following touch events:

- A finger(s) is placed on the element
- A finger(s) is dragged along the element
- A finger(s) is removed from the element

For any given event, the system can detect all the fingers currently on the screen, which of these are on the current UI element, and which fingers were involved in the touch event. For example, if a finger was placed on a UI element and a second finger was then placed on the same element, the TouchPad detects that two fingers were now placed on the UI element.

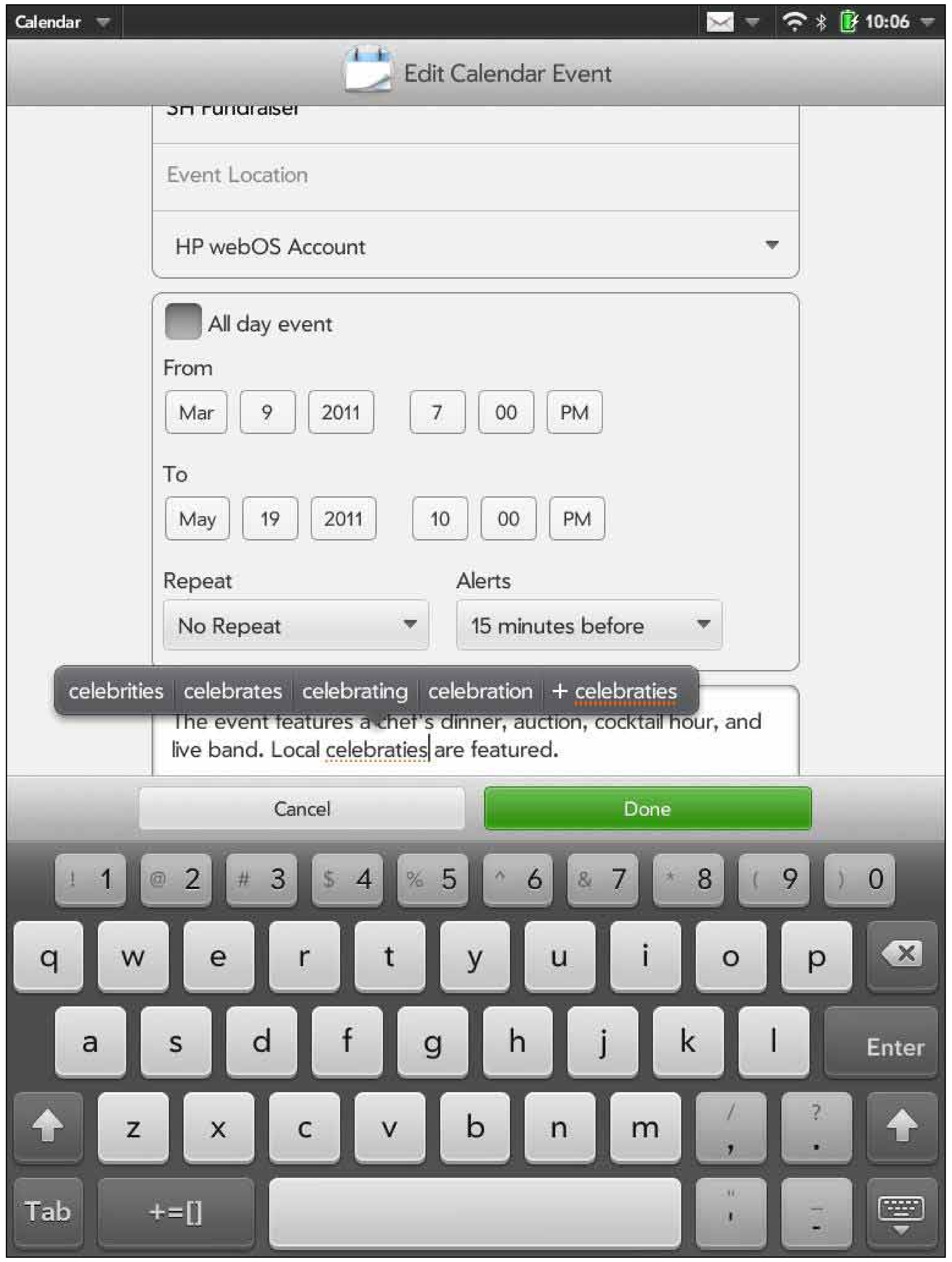
The touch events support provides a rich enough feature set to enable virtually any type of touch-based interaction, including multi-touch gestures, such as pinch-zoom, two finger drag, and rotation. The table on the next page, however, shows the standard gestures used within webOS apps, and the typical actions associated with the gestures. Use these as a reference when designing touch-based interactions in your app but remember that the gesture framework provides support for a broader range of gestures.

STANDARD GESTURES

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 or out; select a word within a text field.
FLICK		Scroll left, right, up, and down.
PRESS & HOLD		Enter edit or 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 in.
PINCH OUT		Zoom content out.

TEXT INPUT

VIRTUAL KEYBOARD



SH Fundraiser

Event Location

HP webOS Account

All day event

From

Mar | 9 | 2011 | 7 | 00 | PM

To

May | 19 | 2011 | 10 | 00 | PM

Repeat: No Repeat

Alerts: 15 minutes before

celebrities | celebrates | celebrating | celebration | + celebraties

The event features a chef's dinner, auction, cocktail hour, and live band. Local celebraties are featured.

The TouchPad does not come with a physical keyboard. Text input is handled instead using a virtual keyboard that is displayed when a user taps in a text field. The keyboard disappears if the user taps off the text field, except when text is being input in an *Interactive Pop-Up* (described later in the guidelines).

- When invoked, the virtual keyboard rises from the bottom of the display and covers the content beneath. The user can still interact with content, for example, to scroll content or tap on buttons.
- The text field that is focused should always remain visible when the virtual keyboard is displayed. If necessary, the view should scroll automatically so that the text field is still in view.
- The virtual keyboard supports suggestions and spell correction. Tapping a word that is not recognized in the dictionary provides the user with a suggested corrections (see example).
- The user can set the size of the keyboard to extra-small, small, medium and large configurations.
- The keyboard supports different language layouts, which the user can set using the TouchPad's Regional Settings.

CONFIGURING THE KEYBOARD

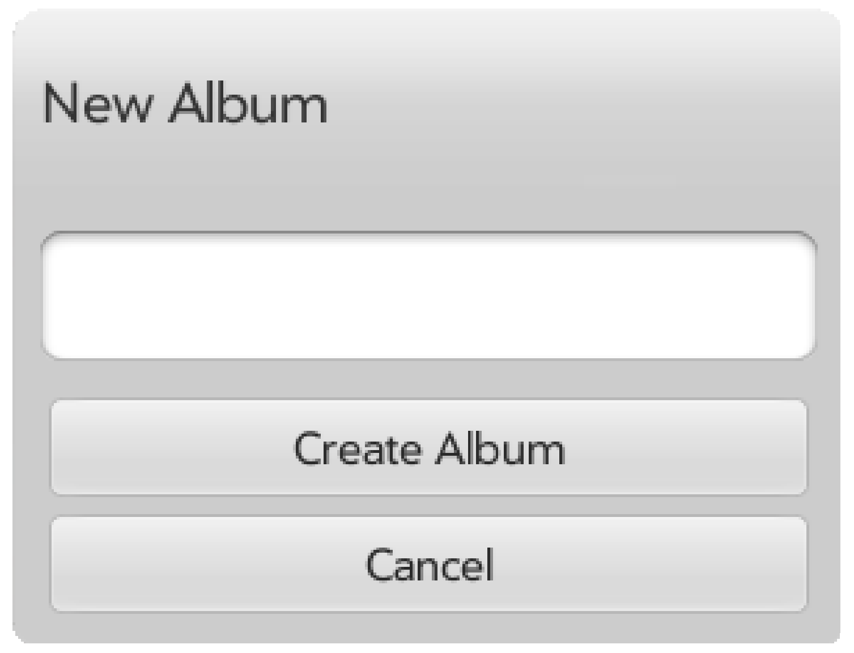
A number of standard keyboard configurations are available. Each keyboard configuration is optimized for a particular type of text input. The app can specify which keyboard layout to display depending on context:

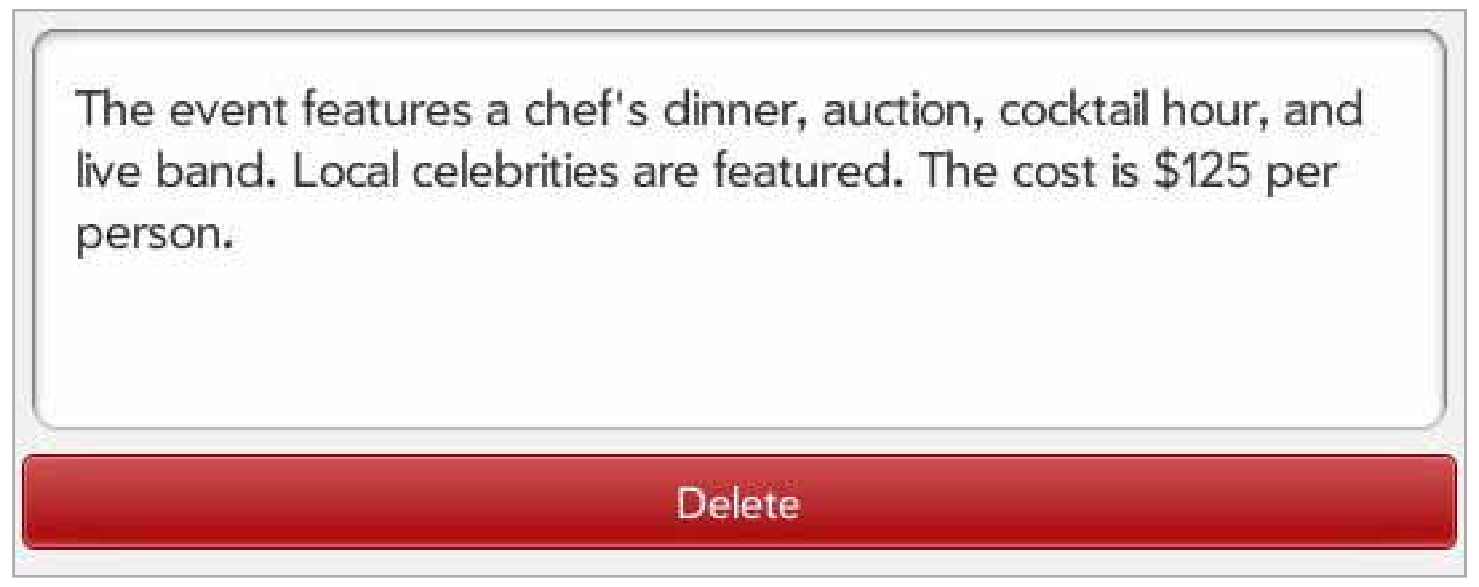
  - Default (alphanumeric)
  - Email—use when entering email addresses
  - Form navigation—use when filling forms
  - URL—use when entering URLs
- An app can also set the Shift key to be on or off when the keyboard is launched, or to switch spell correction on and off when the keyboard is displayed.
- An app can also customize the Enter key to contain a different text string.

TEXT FIELDS

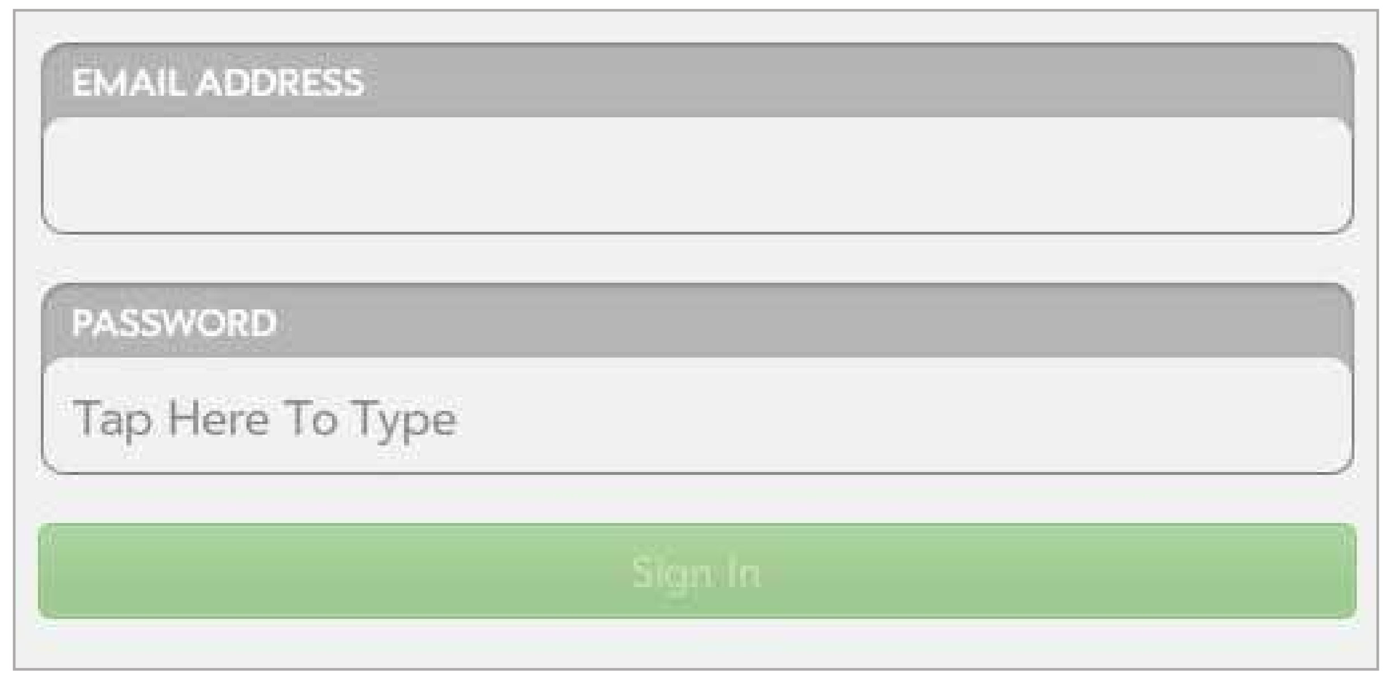
TEXT FIELDS

Single and Multiline Text Fields




The event features a chef's dinner, auction, cocktail hour, and live band. Local celebrities are featured. The cost is $125 per person.

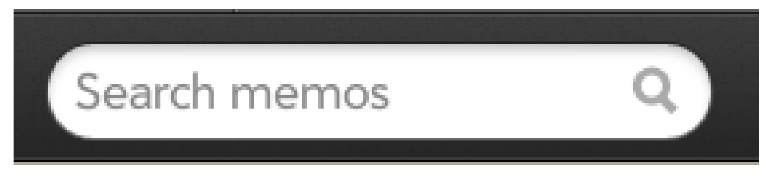
Text Fields with Labels

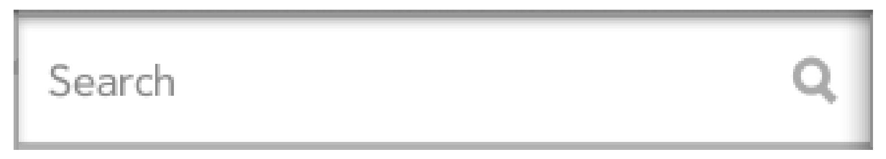


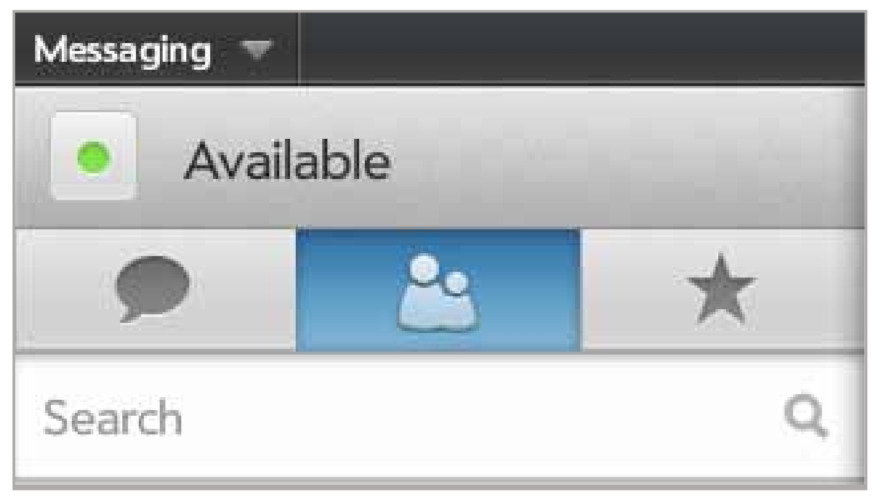
DESCRIPTION

- The webOS framework provides many ready-made text field components.
- Text fields can be single or multi-line and can appear alone or within List and Group Boxes.
- Text fields can contain embedded controls and labels.
- For further information, please see the HP *TouchPad Wireframe Stencils* document.

SEARCH FIELDS



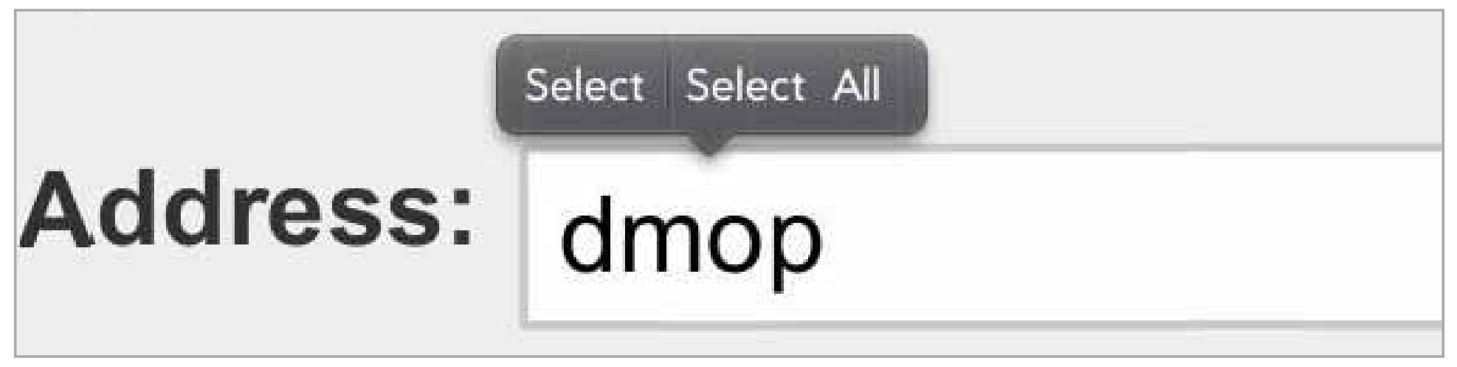


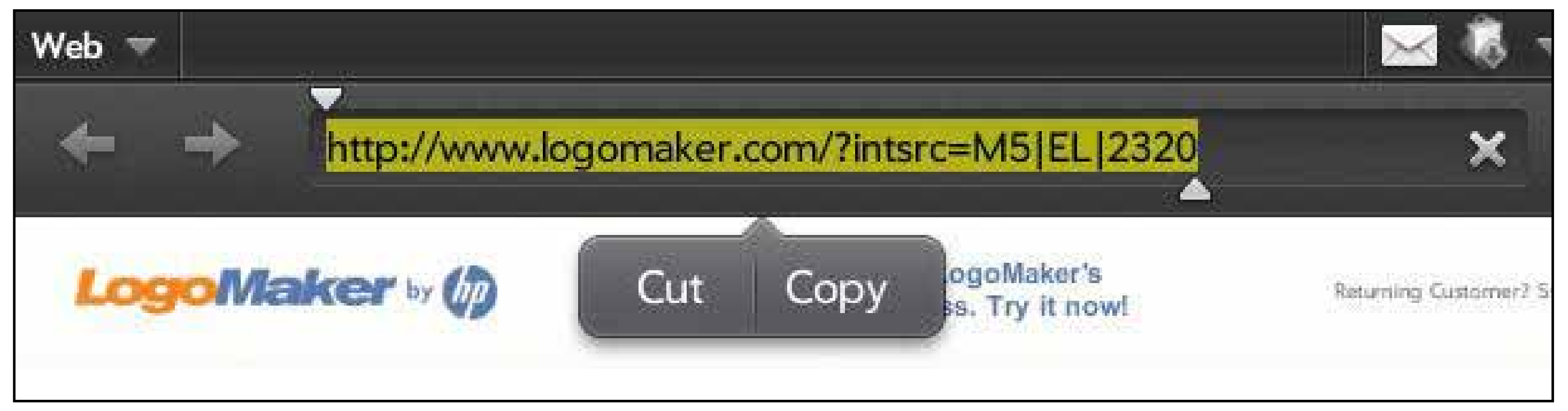


DESCRIPTION

- A search field is a variant of a text input field that contain a search icon and hint text.
- Two style of search field exist. Use a search pill when the search field is placed inside a toolbar and a full width search field when the search field appears within a list.

CUT, COPY & PASTE

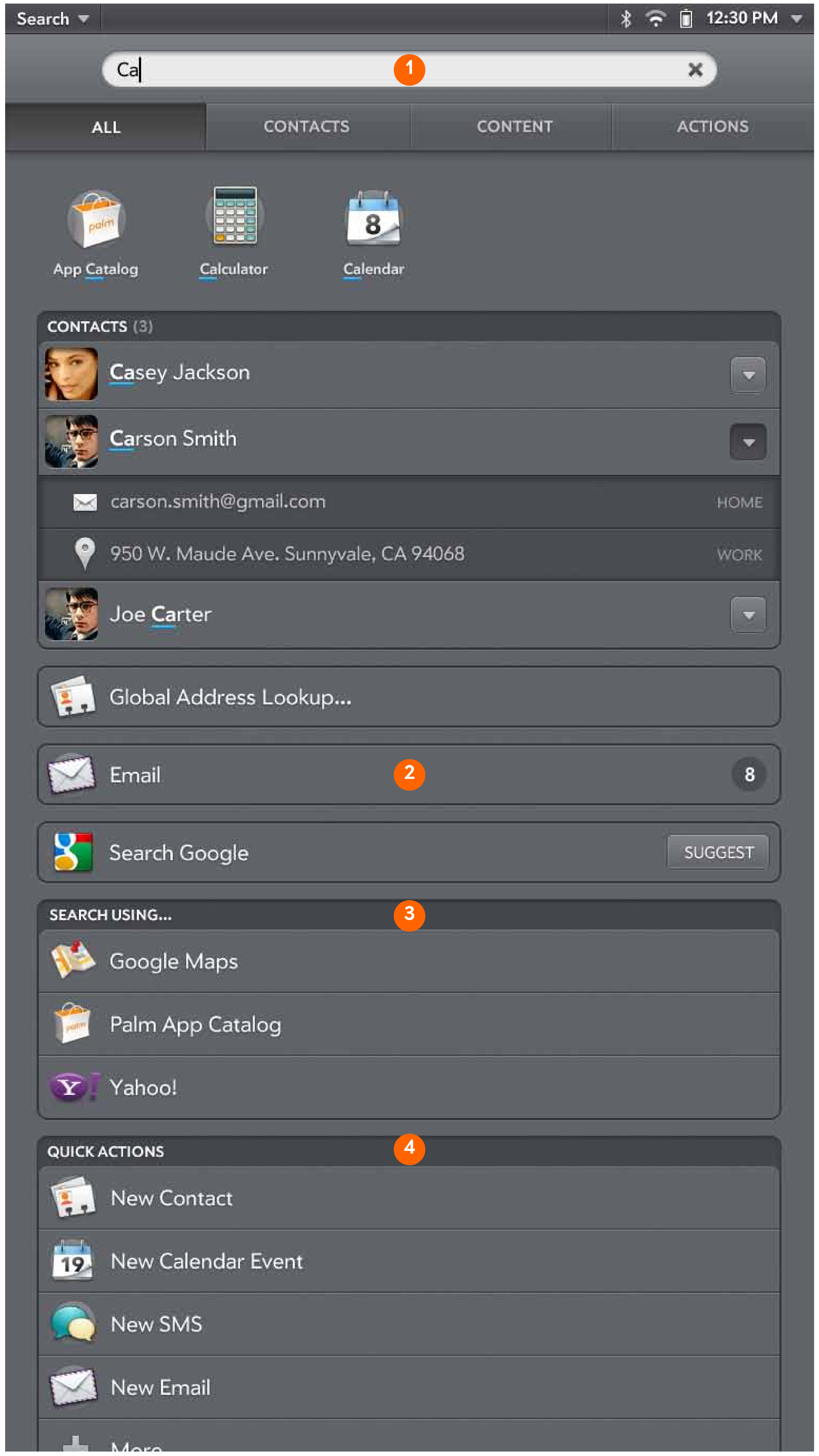




DESCRIPTION

- Cut, Copy & 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.
- After a text selection has been made, the user can cut or copy the selection.

INTEGRATE YOUR APP WITH JUST TYPE



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

A POWERFUL UNIVERSAL SEARCH AND MORE

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

SEARCH DATA WITHIN YOUR APP

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

SEARCH DATA ONLINE

Just Type can also trigger an online search. By integrating with Just Type, your app is listed among the Online Content searches. Tapping an Online Content search launches your app and passes along the search term.

TRIGGER QUICK ACTIONS

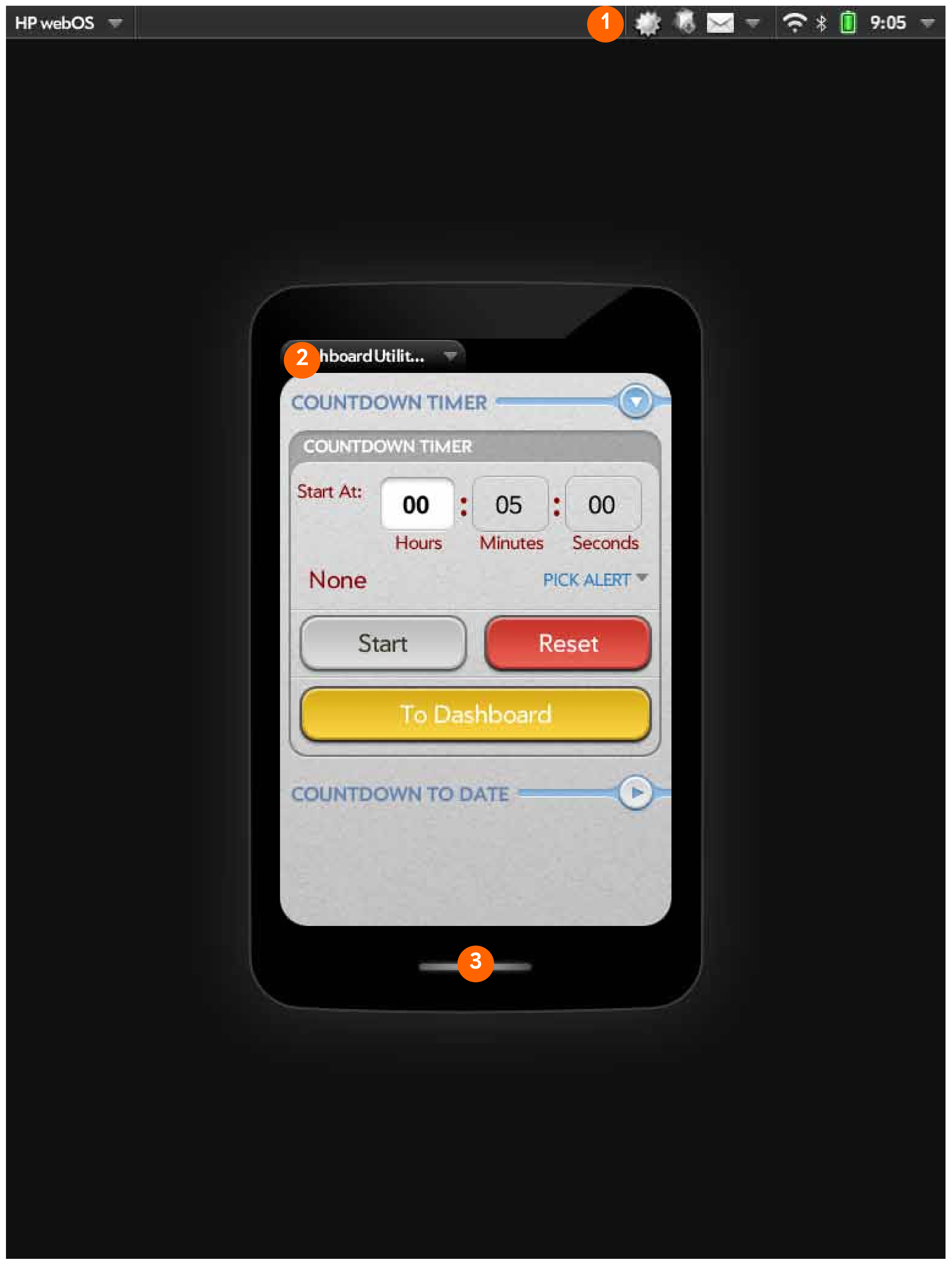
Just Type also displays a list of Quick Actions, such as creating new content, updating your status, etc. The user can perform all of these without the need to launch an app. Your apps can define one or more Quick Actions that link to functionality in your app and make the actions available to users 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.
App 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 app 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 app 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.
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 app. For further information, please refer to the Service API documentation in the SDK.

COMPATIBILITY MODE



1. Status bar
2. App menu
3. Back gesture area

RUNNING MOJO™ APPS ON THE TOUCHPAD

The Enyo framework is the preferred method for creating new webOS applications, but legacy Mojo applications can run on the TouchPad in Compatibility mode.

When the user launches the application, it displays within a window in either portrait or landscape orientation depending on the default orientation for the application.

When running an app in Emulation mode:

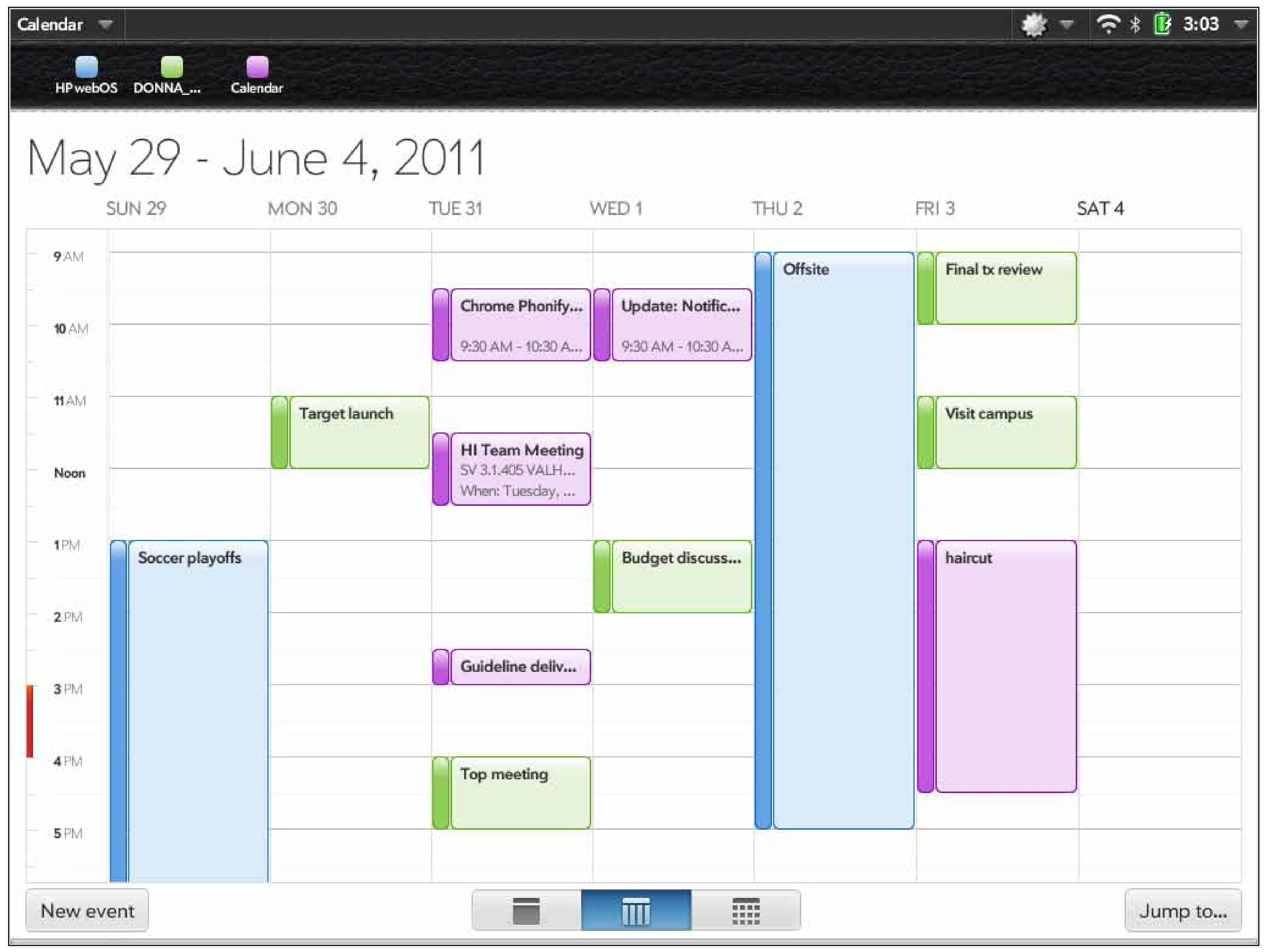
- The App menu appears in the upper left and functions the same as on the device.
- Notifications and alerts appear in the status bar.
- To navigate, users can move between views by swiping back on the Back gesture area in the Compatibility mode window. Swiping from right to left displays the previous view.

APP FRAMEWORK

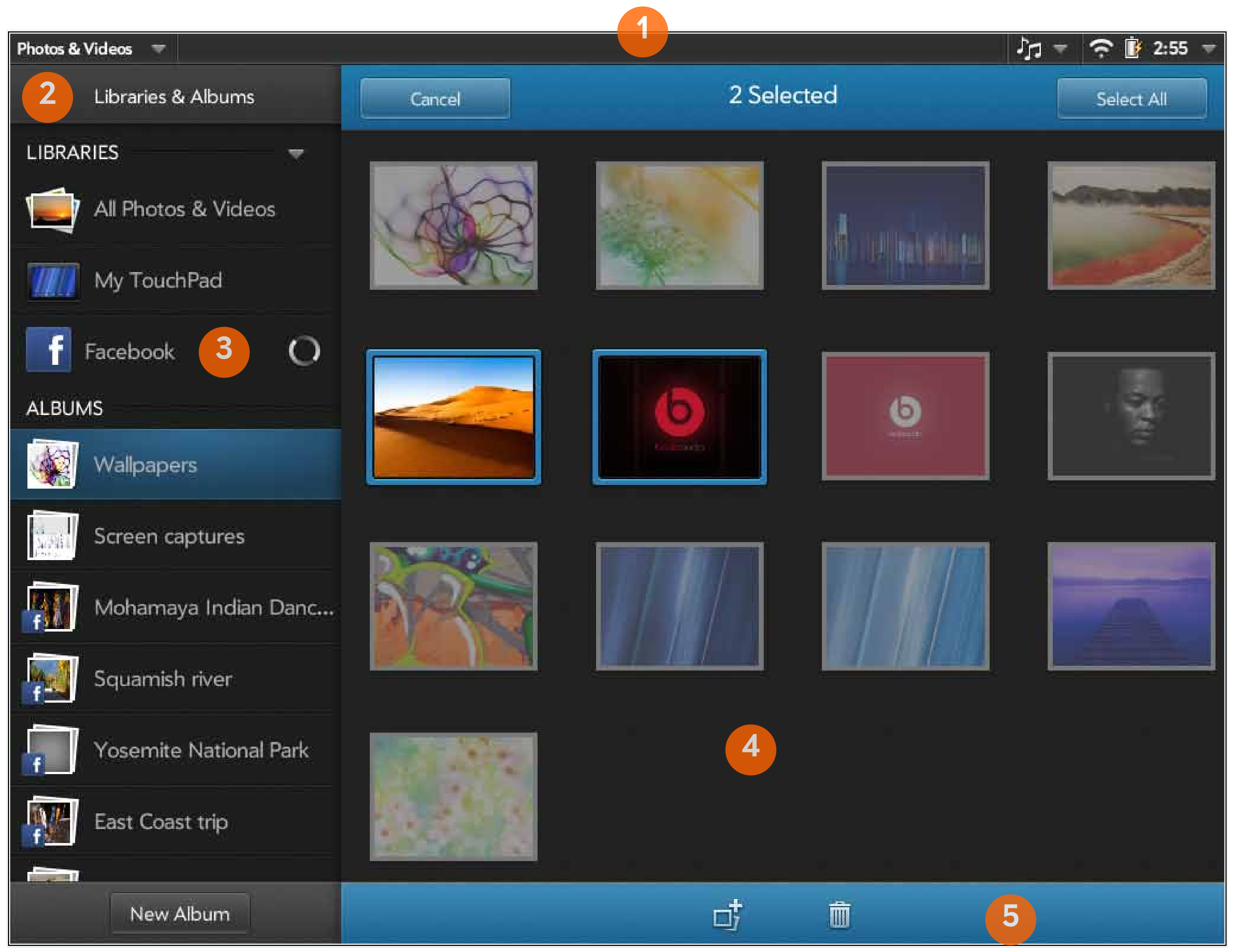
This section will describe the key building blocks of the Enyo application framework and how to use them to structure your app's user interface (UI). For a complete set of UI components, please refer to the *HP TouchPad Wireframe Stencils*.

VIEWS AND PANES

SINGLE PANE LAYOUT



SPLIT PANE LAYOUT



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

The webOS framework provides a great deal of flexibility in creating view layouts for your app. An app can consist of one or more **views**, each of which can contain one or more **panes** that contain app content. A view can either have just a single pane, a split pane, or multiple panes - please see the *Designing Your App* section for more details of the different layouts.

App content can be freely assembled either using common UI components provided by the webOS framework, or using custom components and app-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 within the view.

Each pane in the app can also contain a Header and Footer area to contain application controls and navigational elements. If the view has multiple panes, they should all either consistently contain Headers or else none should contain Headers for a consistent visual appearance.

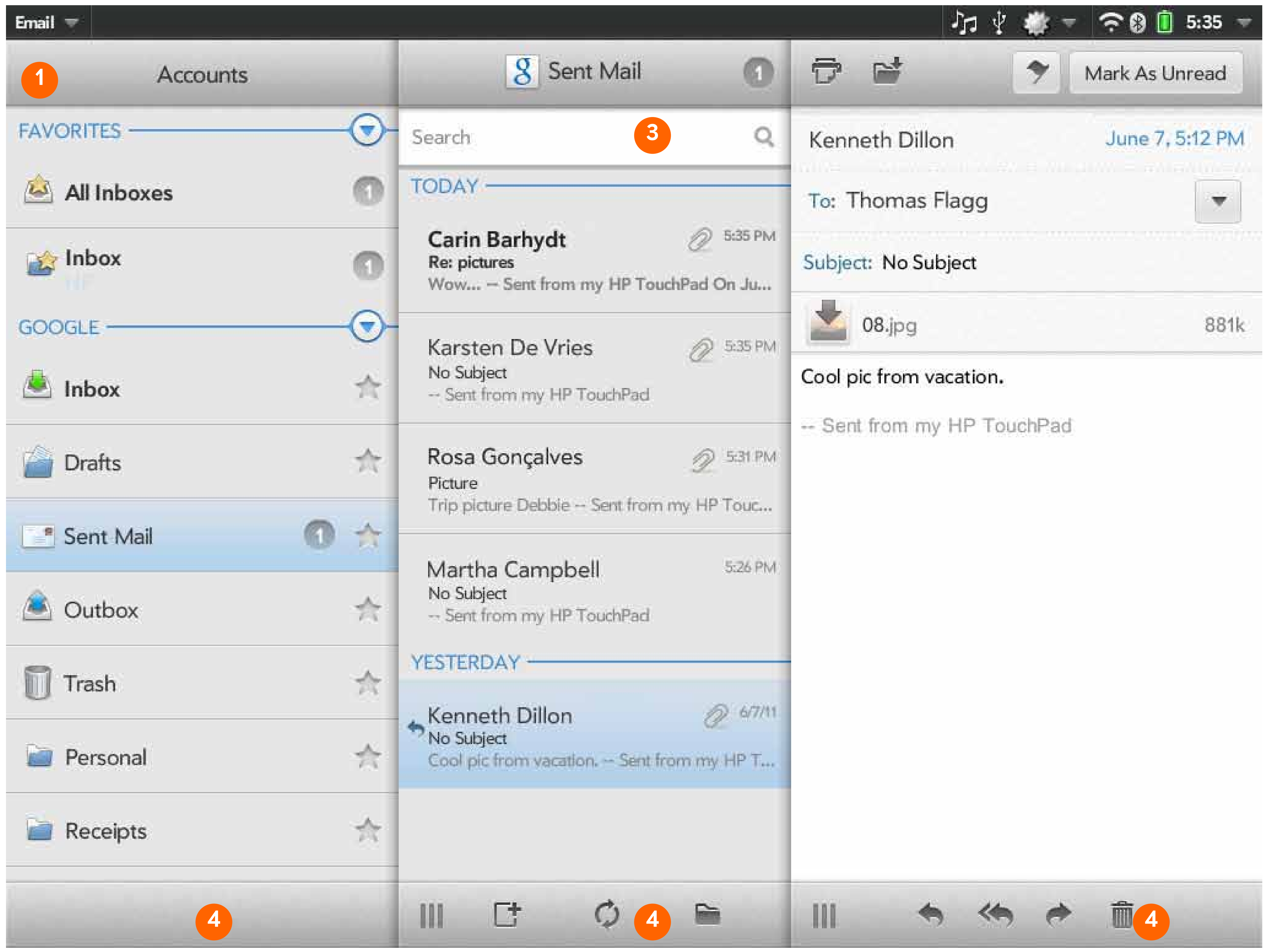
STATUS BAR

The status bar is generally visible on most views as it provides an area for key system indicators, such as battery and network strength, as well app 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.

HEADERS AND FOOTERS

A Header area can contain several stacked Headers. Headers can either be fixed or scrolling. Fixed headers remain anchored to the top of the pane whereas scrolling headers scroll with the app content. Headers can contain a toolbar or other contents, such as a search field. Footers can be used to contain a toolbar that contains icon buttons. Please refer to the *Designing Your App* section for details of what to use Headers and Footers for and the *HP Wireframe Stencils* document for examples.

EMAIL HEADER AND FOOTER AREAS



Carin Barhydt 5:35 PM
Re: pictures
Wow... -- Sent from my HP TouchPad On Ju...

Karsten De Vries 5:35 PM
No Subject
-- Sent from my HP TouchPad

Rosa Gonçalves 5:31 PM
Picture
Trip picture Debbie -- Sent from my HP Touc...

Martha Campbell 5:26 PM
No Subject
-- Sent from my HP TouchPad

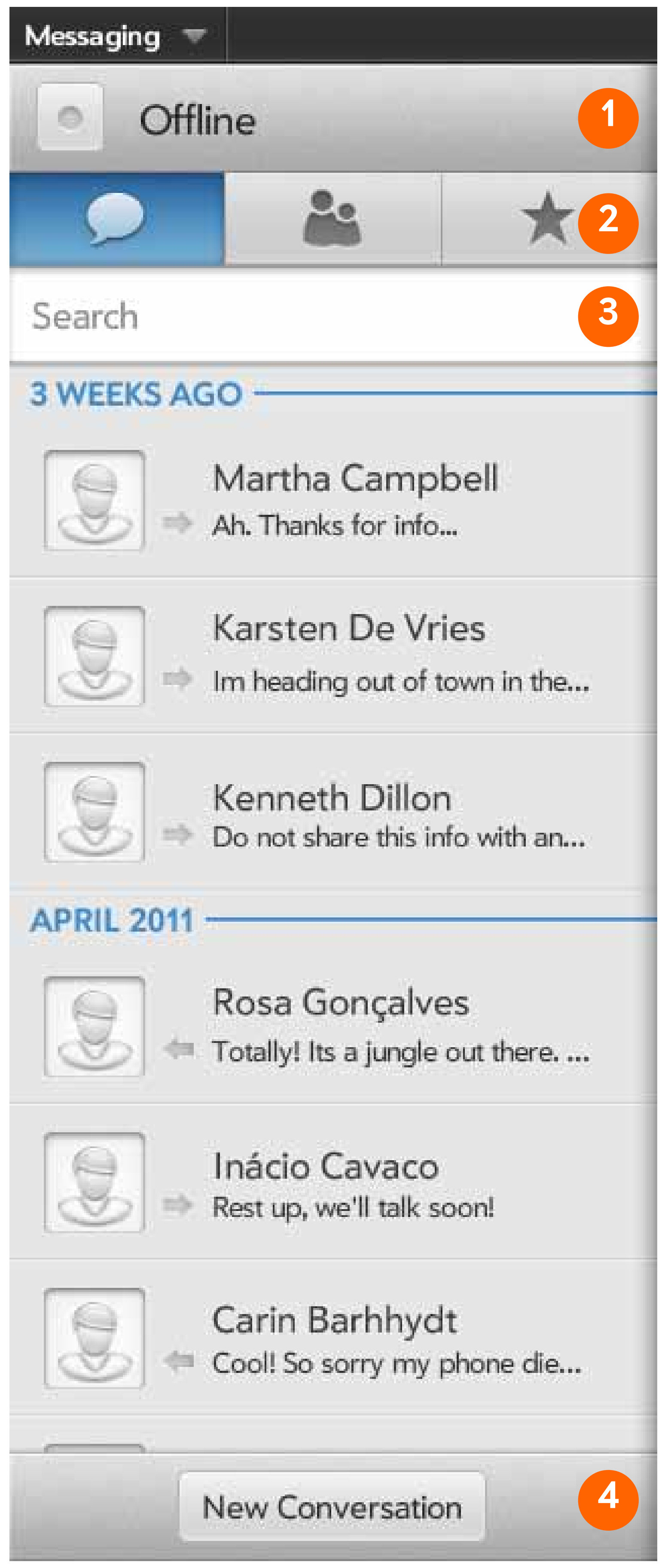
Kenneth Dillon 6/7/11
No Subject
Cool pic from vacation. -- Sent from my HP T...

Kenneth Dillon June 7, 5:12 PM
To: Thomas Flagg
Subject: No Subject
08.jpg 881k

Cool pic from vacation.

-- Sent from my HP TouchPad

MESSAGING HEADER AND FOOTER AREAS



Martha Campbell
Ah. Thanks for info...

Karsten De Vries
Im heading out of town in the...

Kenneth Dillon
Do not share this info with an...

Rosa Gonçalves
Totally! Its a jungle out there. ...

Inácio Cavaco
Rest up, we'll talk soon!

Carin Barhydt
Cool! So sorry my phone die...

1 Header toolbar with title
2 Tabs
3 Search field
4 Footer toolbar

DESCRIPTION

- Each pane contains a toolbar in the Header and Footer area.
- The header toolbar in the left pane contains the pane's title.
- The header toolbar in the middle pane contains the pane's title and a badge to indicate the number of unread emails. Beneath the toolbar there is also a search field in the header area.
- The header in the right pane contains an icon button to print the email as well as other buttons to flag and mark the email.
- The toolbar in the footer area contain icon buttons for various actions, such as creating new emails, replying to, forwarding, deleting and moving emails. The middle and right footers also contain grab handles for dragging the panes - see the *Layout: Sliding Panes* section for details.

DESCRIPTION

- The Header area in the messaging application contains three stacked elements - a toolbar contain the pane title, a set of tabs to navigate content, and a search field to filter the list contents. All three header elements are fixed and so do not scroll with the list.
- The footer toolbar contains a push button to initiate a new conversation.

LISTS

One of the most common ways of displaying content within a pane is to structure the content using a list. A list is a multi-line row that displays a set of available items, of which one or many can be selected.

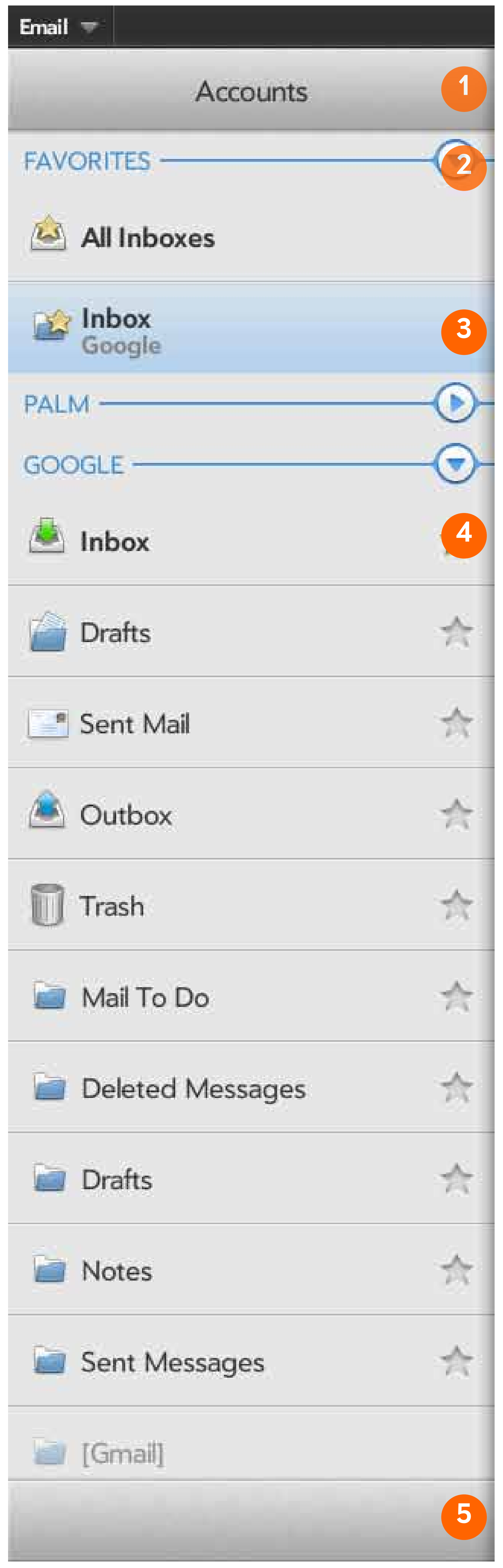
A list can be used to add other items to the list or to delete items from the list (see the next page). Lists can also:
- Contain a variety of dividers for organizing the list in different sections (see *HP Wireframe Stencils* for details).
- Contain expandable and collapsible dividers.

A list row can contain a single, tappable item or multiple items, each which process a tap differently. In a simple list, the entire row is tappable. Tapping the item either selects the item (revealing further details in a new pane or view), or triggers an action, such as playing a song.

In more complex designs, each row can contain several different elements. For example, a list item could contain a thumbnail image that can be tapped to open a larger image inside a pop-up, or a button to trigger to show further details of an item. See the Photos & Videos app.

Please refer to the HP Wireframe Stencils for more details on different list and list divider examples.

LIST EXAMPLE

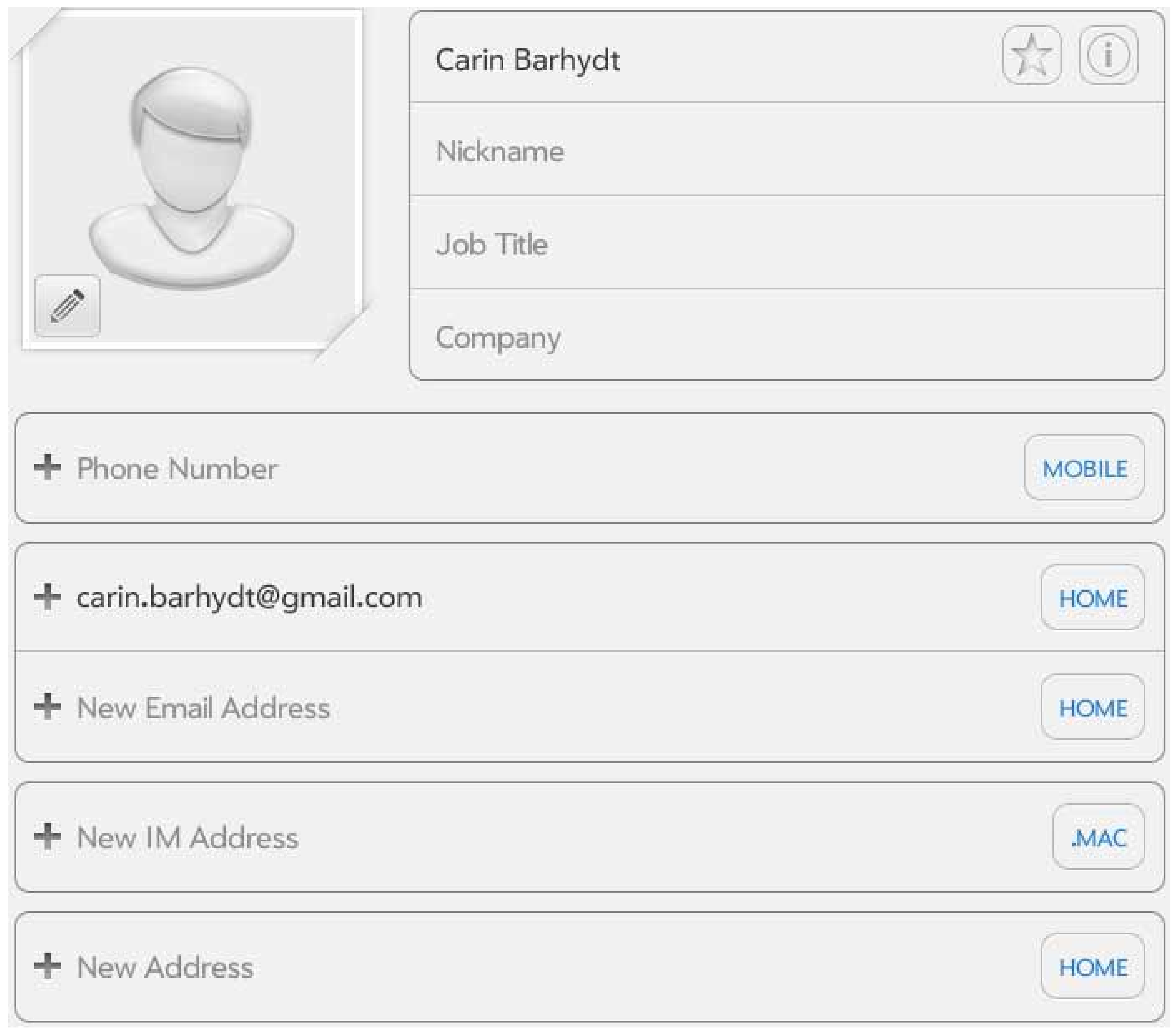


1	Pane header
2	Collapsible divider
3	Simple list item
4	Multi action list item
5	Pane footer

DESCRIPTION
- The list uses collapsible dividers to organize content. The user taps on the to collapse (see Palm divider) and expand (see Google divider) content.
- List items can contain a single tappable area; for example, see item 3. Tapping on this item shows the content of the Inbox in the adjacent pane.
- List items can contain multiple tap targets; for example, for item 4, tapping the icon adds the list item to the Favorites category above.

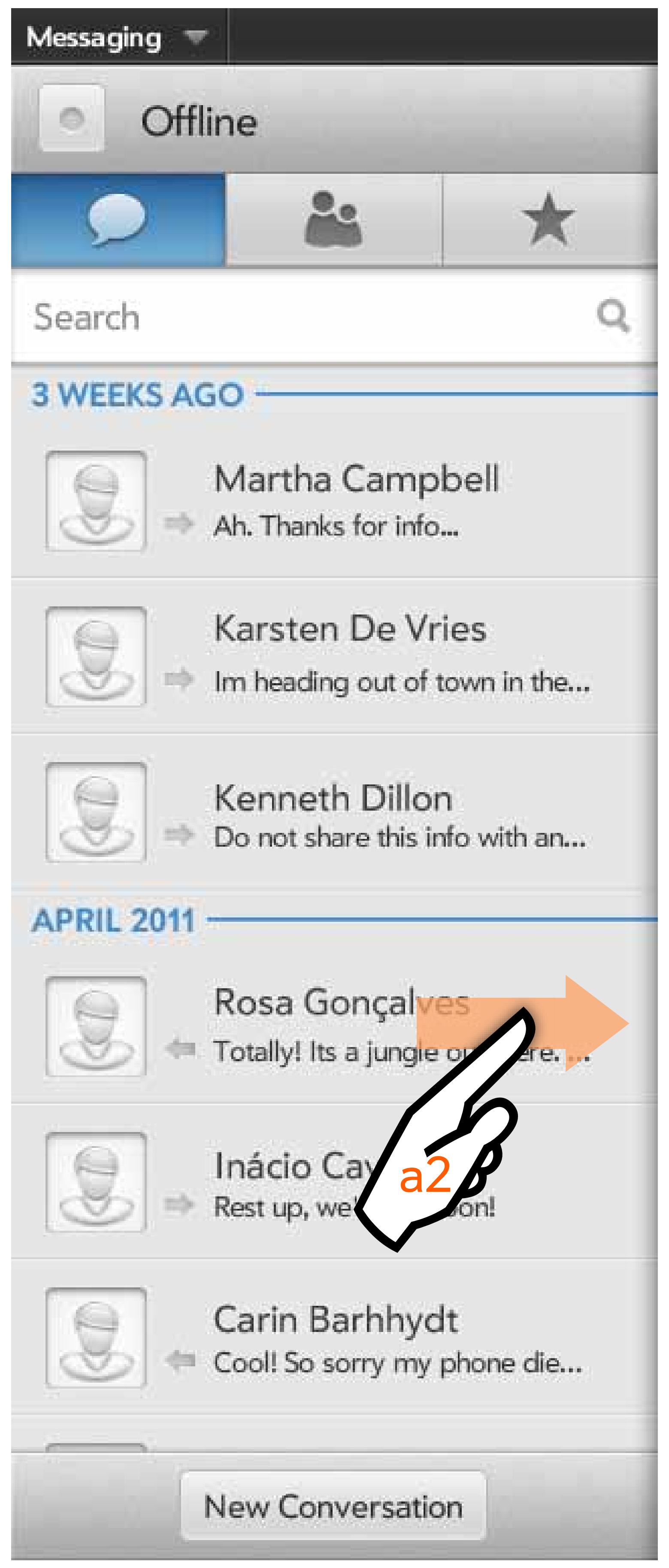
LISTS: ADDING AND DELETING ITEMS

ADDING ITEMS

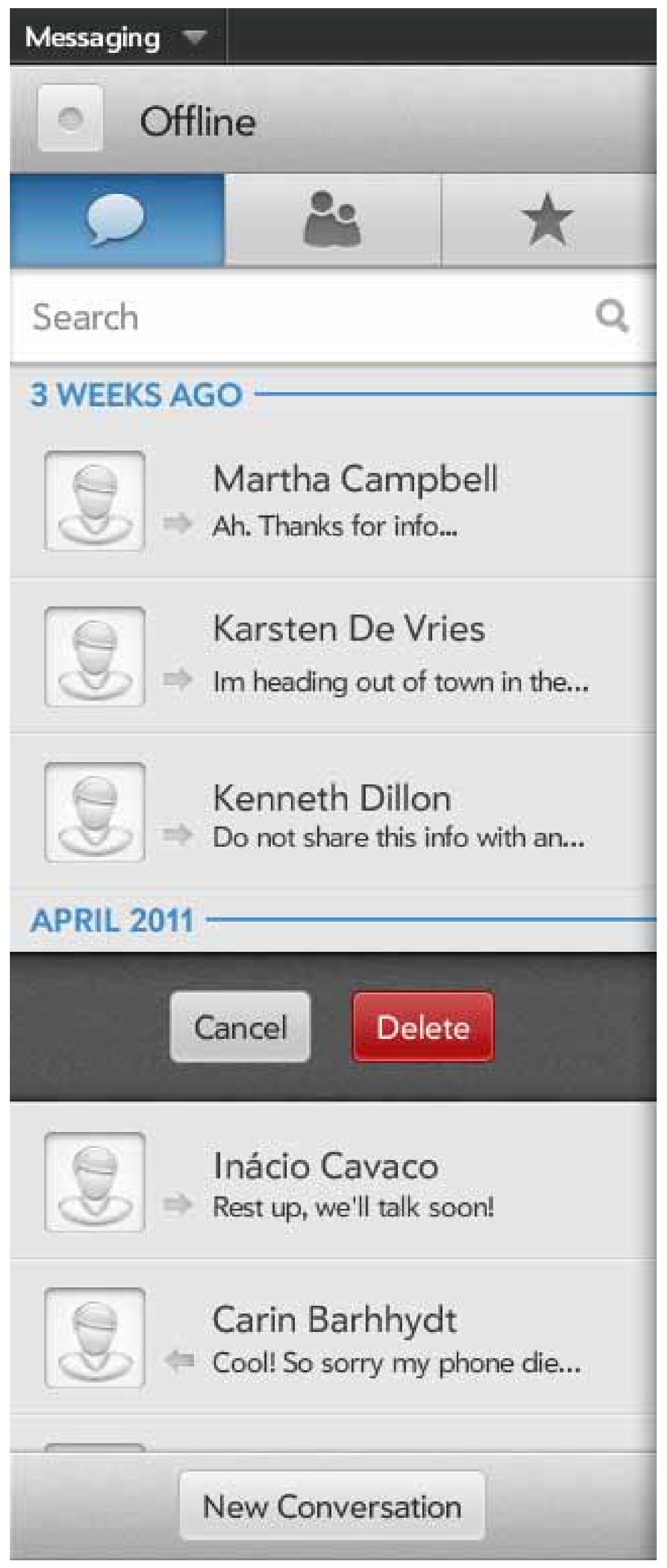


DELETING ITEMS

a1. List of Messages



a2. Delete confirmation



DESCRIPTION

- Lists can contain text input fields to allow the user to edit the list in line and add additional items.
- For example, in the Contacts app, the user can tap on any of the fields marked with a "+" to edit or input new contact details.
- For further examples of in line editing, please refer to the *Buttons and Forms* section of the *HP TouchPad webOS Wireframe Stencils.*

DESCRIPTION

- Lists can also be used to quickly delete items using gestures.
- Swiping a list item away to either the left or right reveals confirmation buttons beneath the list item. The user can confirm to delete the item or cancel the operation.

BUTTONS: PUSH BUTTONS

WHAT THEY ARE

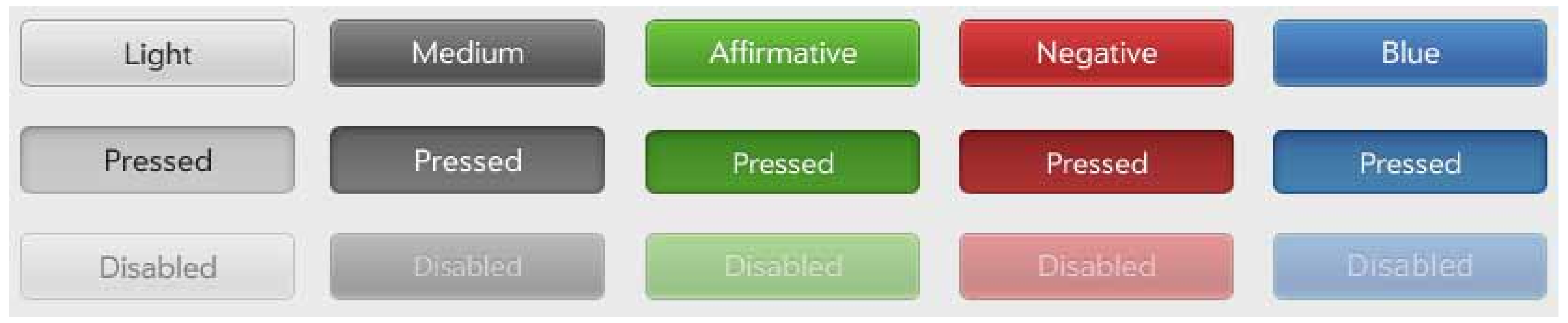
Push buttons are used to enable the user to perform an action. They are called push buttons because the user needs to tap or 'push' them to trigger the action. When the button is pressed, it takes on a 'Pressed' state and returns to its normal state when the finger is lifted.

Push buttons can also have a 'Disabled' state which is shown when the button's action cannot be completed. The 'Disabled' state is also shown when an activity indicator is shown in the button - see the section on *Providing Feedback*.

There are a variety of different standard colors for buttons:

- **Light grey** - this is the default color for buttons and should be used in most circumstances.
- **Grey** - is used to indicate a primary action out of a selection of different choices. The primary action is the action that would generally be selected in most cases. If there is no primary action, then do not use the dark grey color.
- **Green** - is used in situations where the user needs to make a decision between a positive and a negative action - e.g., between 'Done' and 'Cancel', 'Yes'/'No'. The green button should be used to indicate the positive choice. In task flow navigation (see *Task Flow Navigation* section), the 'Done' button should be green.
- **Red** - is used to indicate an action that has destructive consequences. It is typically used for a delete action.
- **Blue** - is used to highlight a button to draw the user's attention to a button. Use blue if you want to make an action prominent in the interface. The blue color should be used sparingly.

PUSH BUTTONS



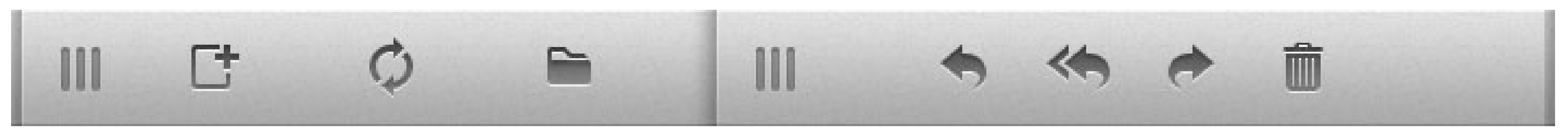
WHEN TO USE

Use push buttons to enable the user to trigger actions - push buttons are commonly used in interactive pop-ups, dialogs, within lists, headers and footers.

BUTTONS: OTHER BUTTONS

HP webOS has a variety of button styles. This page outlines some guidelines for using the most common button styles. For full details of the different button styles, please refer to the *HP TouchPad Wireframe Stencils.*

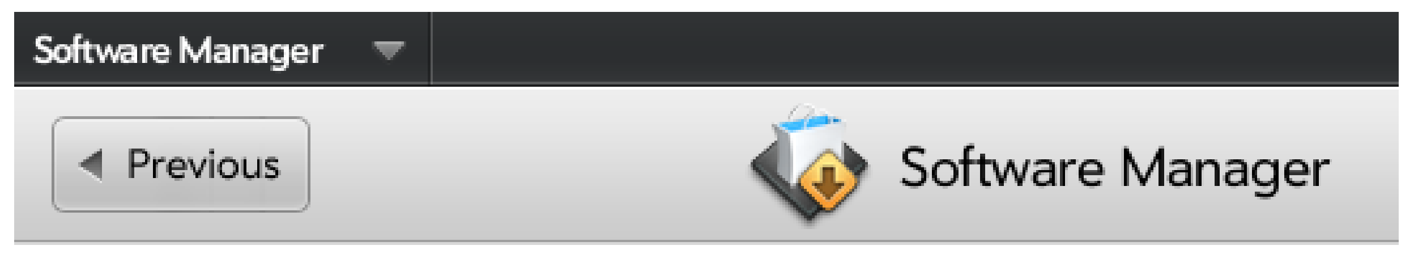
ICON BUTTONS

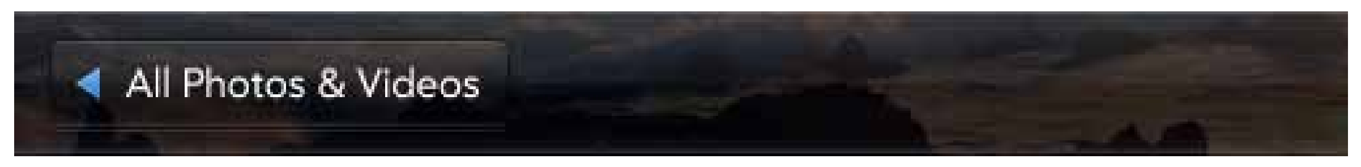


DESCRIPTION

- Icons buttons appear within toolbar components in either the header toolbar or the footer toolbar.
- Like push buttons, icon buttons also have a pressed state, but icon buttons have no label and no button border.

BACK BUTTON

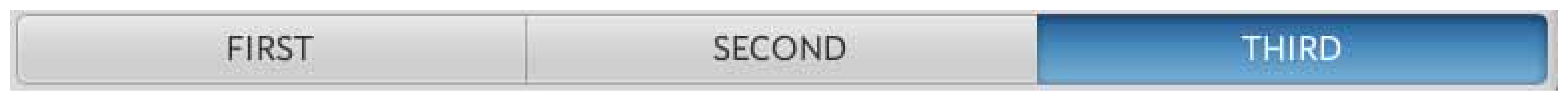




DESCRIPTION

- A back button takes users back to the view from which they came. See *Hierarchical Navigation* section.
- A back button is a button that appears inside a toolbar in either the header or footer area.
- The back button should be aligned to the left of the toolbar.
- There are two styles of back buttons - one for a light-styled toolbar and one for a dark-styled toolbar.
- The back button contains a back arrow to distinguish it from push buttons, but otherwise these buttons behave in the same way.

RADIO BUTTONS



DESCRIPTION

- Radio buttons are used to select an item from a group.
- Tapping on a radio button selects the item from the group of buttons - the selected item is shown in a selected state.
- Only one button from the group can be selected.
- Radio buttons can be shown within a Header - in this case, selecting an item should change the contents that are shown in the pane.

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. A scrim is displayed beneath the interactive pop-up to indicate to the user that the view is non-interactive.

As the name suggests, interactive pop-ups contain interactive content that is displayed inside a pop-up. The content varies freely depending on the app 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.

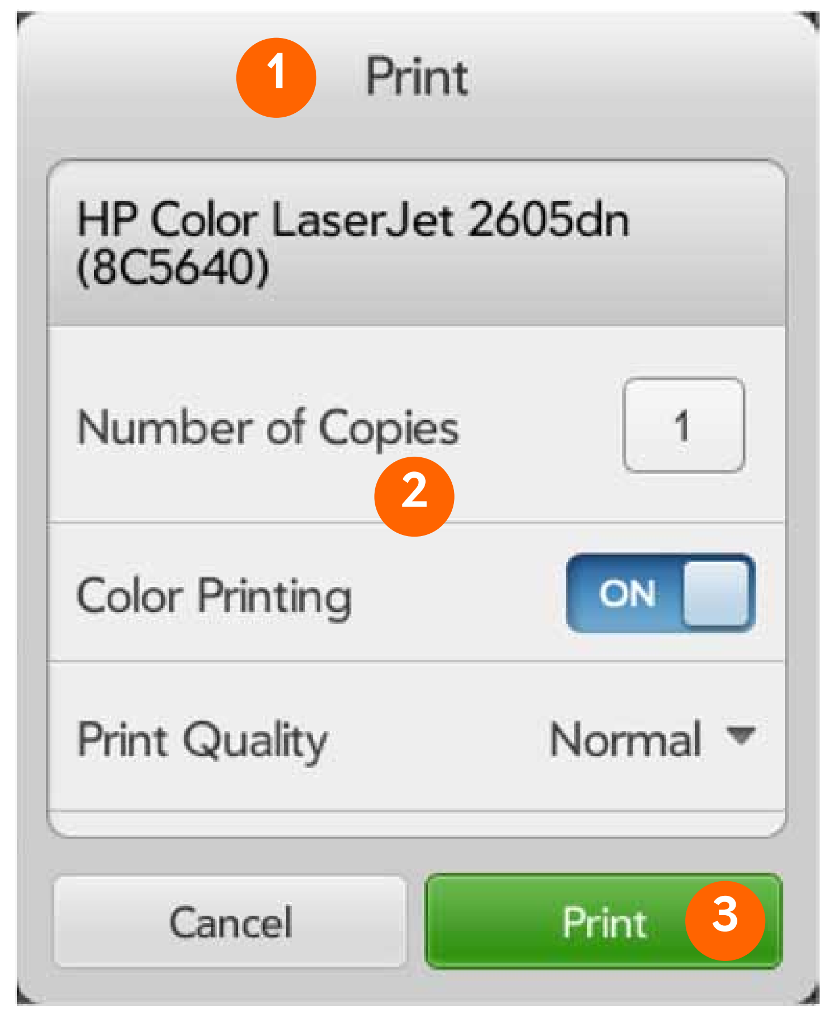
In addition to the content area, pop-ups also contain a button area and an optional header area. Headers may sit in the chrome or scroll with the content. Pop-ups may contain a title or instructions for the user. Pop-ups may or may not be editable.

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, for example, Print) are stacked and navigation buttons (buttons used to confirm, cancel or navigate between views) are shown side-by-side. If buttons are displayed side-by-side, no more than two buttons should be shown in the pop-up.

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. The sub-task should be relevant to the view from which it was launched. On tablet devices that have large screens, navigating away from the view to perform these smaller tasks can be disorientating. Interactive pop-ups allow the user to remain within the context of the view that the interactive pop-ups are in.

Nevertheless, avoid the temptation to contain too much interaction inside a pop-up. Interaction inside a pop-up should be brief and contained and should be used sparingly and only if there not an elegant way to perform the interaction in-line. An interactive pop-up should not be the main way for the user to interact with a view.



HP Color LaserJet 2605dn (8C5640)

Number of Copies: 1

Color Printing: ON

Print Quality: Normal

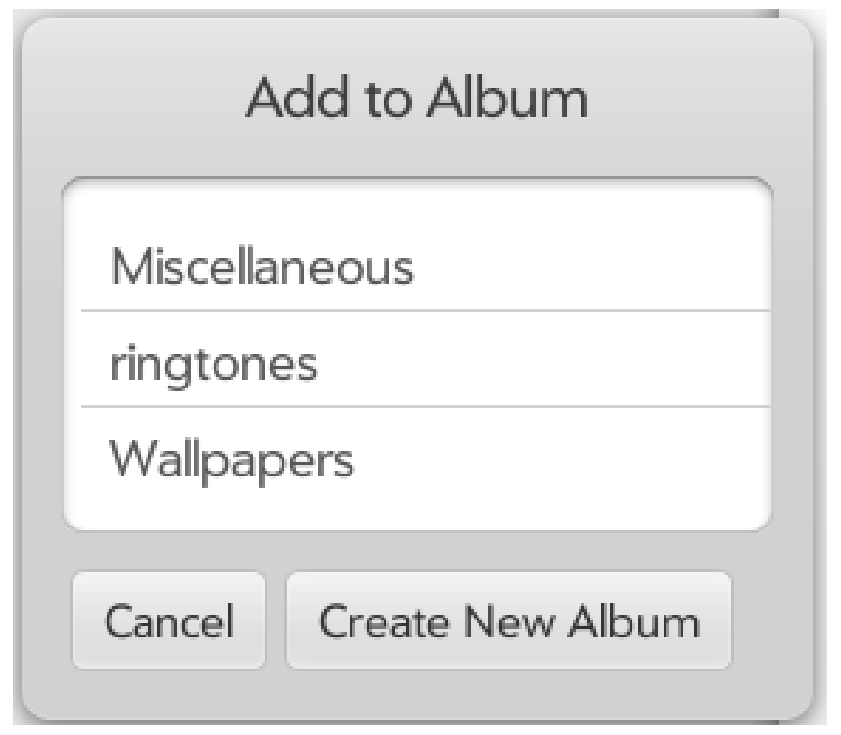
1 Header area - Title or instructions

2 Content area

3 Button area

INTERACTIVE POP-UPS: EXAMPLES

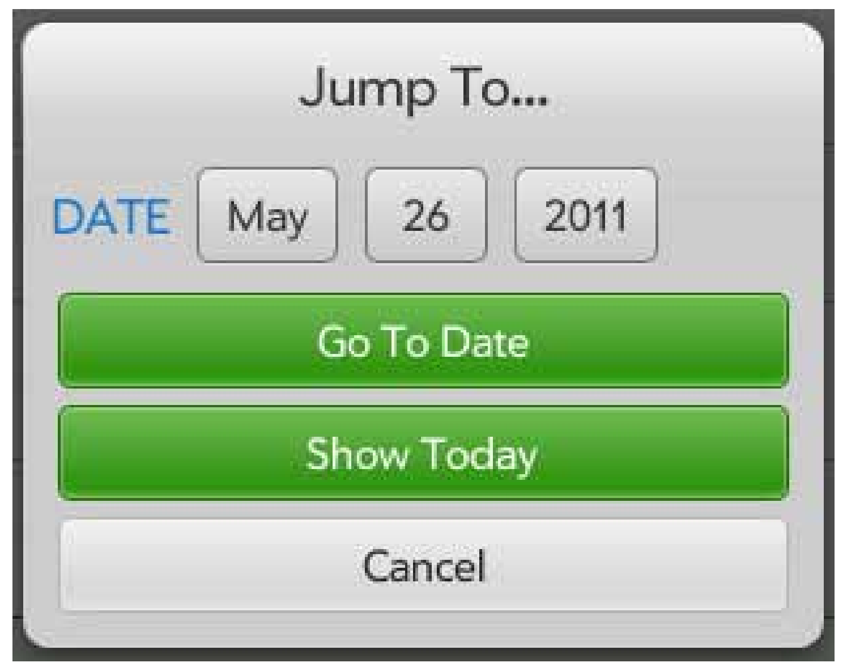
SELECTION POP-UP



DESCRIPTION

- Use this style of pop-up so users can make a single selection from a range of options.
- Use a header to describe the selection list.
- Let user select an item by tapping one of the items in the list, and that closes the pop-up.
- Add a "Cancel" button to dismiss the pop-up without making a selection.

MULTI-ACTION POP-UP

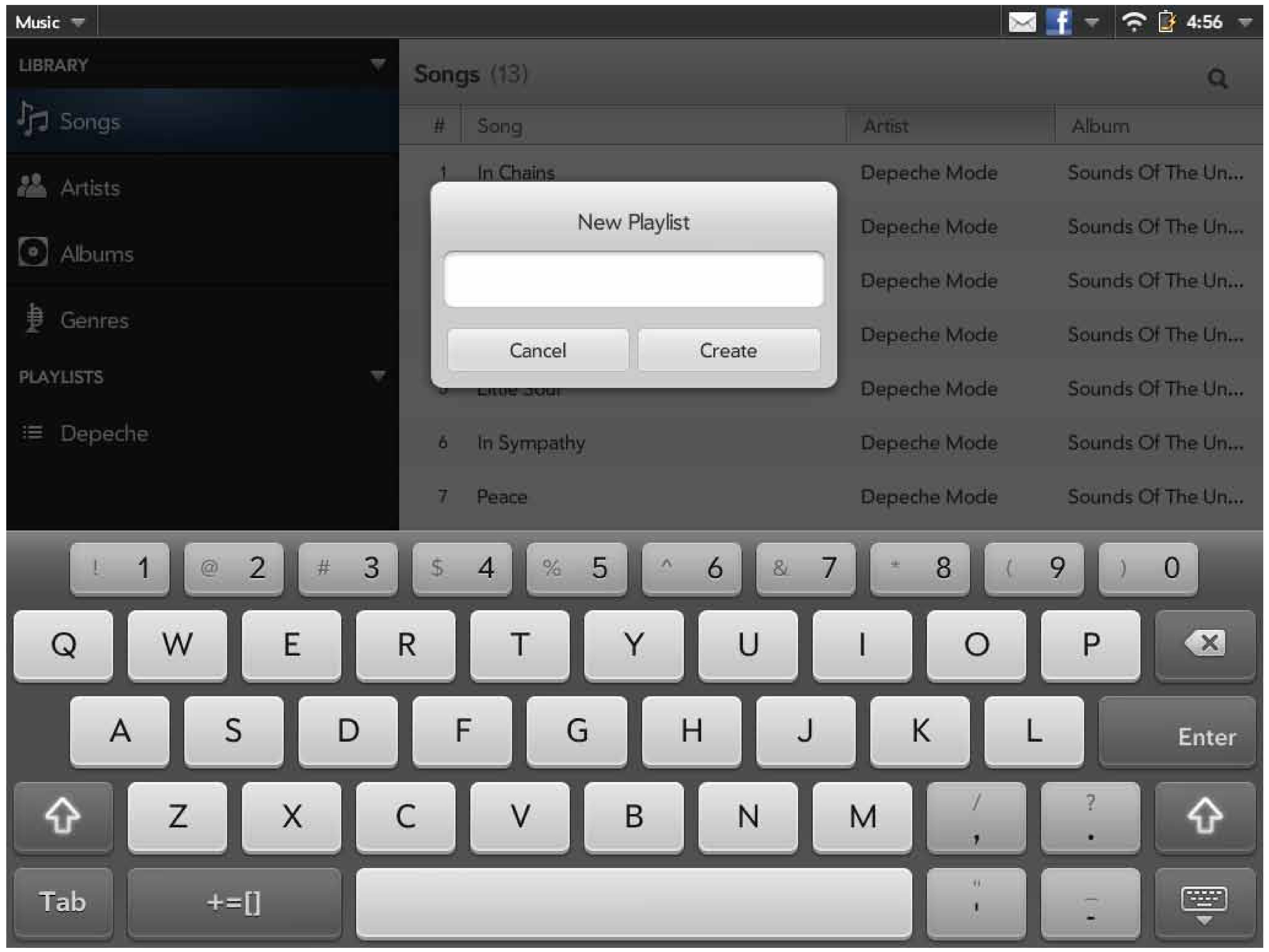


DESCRIPTION

- Use this style of pop-up to give the user a selection of actions.
- Use vertically stacked buttons to present choices to the user.
- Add a "Cancel" button to dismiss the pop up without saving the changes.

INTERACTIVE POP-UPS: TEXT INPUT

TEXT ENTRY POP-UP



One usage for interactive pop-ups is to enable users to input text.

If the user taps on a text field within an interactive pop-up, the virtual keyboard is invoked as usual to enable text input. However, 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. For these reasons, it is generally better for users to enter text in-line rather than in an interactive pop-up.

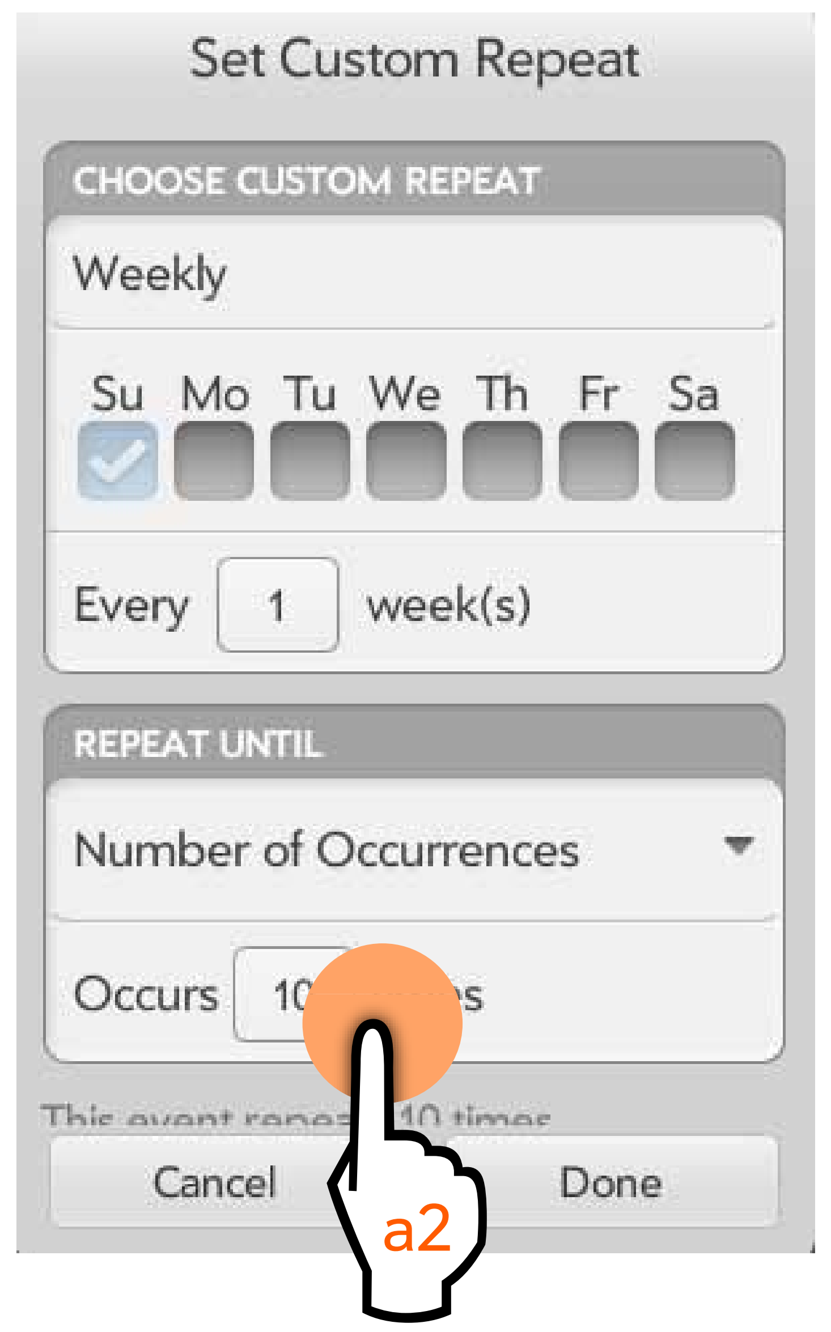
NOTE: The virtual keyboard should not be dismissed until the interactive pop-up is also dismissed.

INTERACTIVE POP-UPS: FORM FILLING

Another common usage for interactive pop-ups is to enable users to fill out form elements. In the example 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.

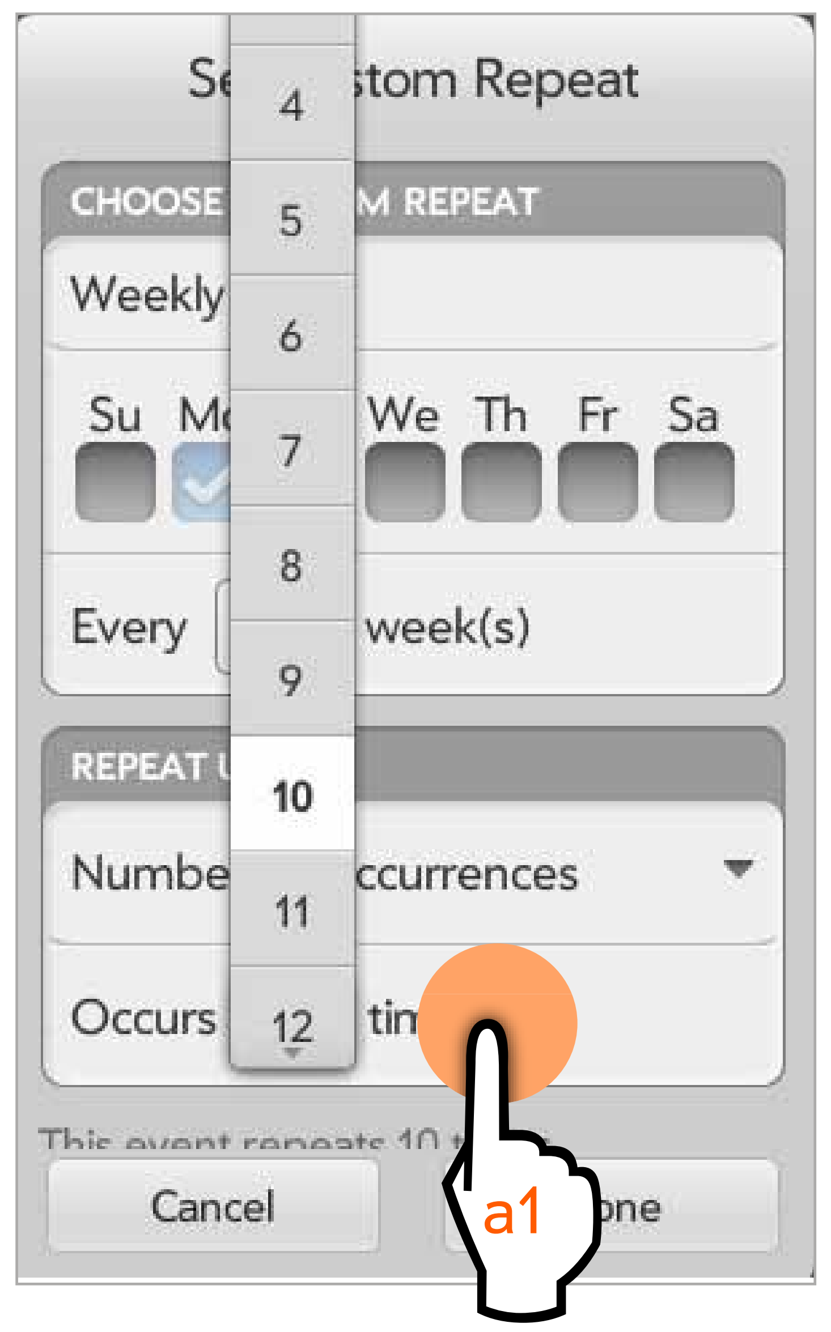
Interactive pop-ups can also be used for longer interaction sequences that guide the user through several views within a task flow. See the *Task flow* page within the *Navigation Patterns* section for further details. For general guidelines on form filling, see the *Form Filling* section.

a1. Pop-up with 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. Additional component in a new layer



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

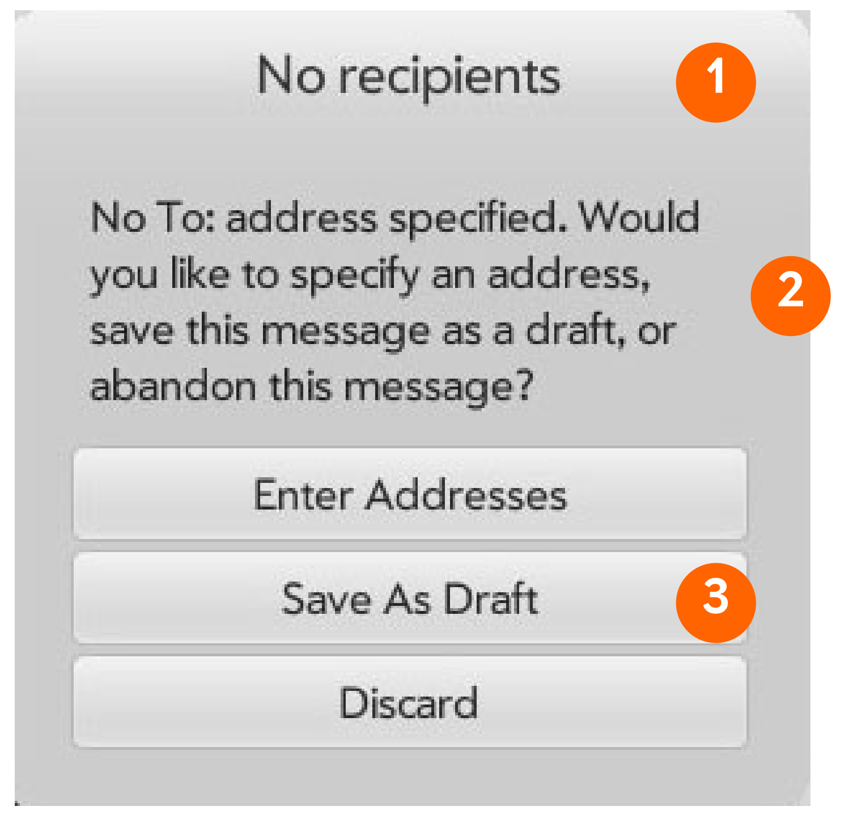
DIALOGS

WHAT THEY ARE

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

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 are shown side-by-side, whereas action buttons should be stacked vertically.


No recipients

No To: address specified. Would you like to specify an address, save this message as a draft, or abandon this message?

1 Title

2 Body text

3 Buttons

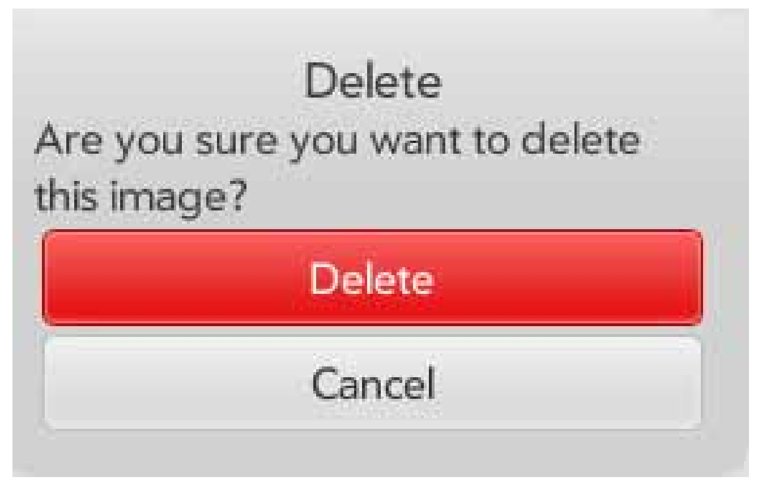
WHEN TO USE

Use dialogs to prompt the user to perform simple and quick interactions, such as to confirm an action or to inform the user. Use dialogs when your app requires confirmation or disambiguation from the user. For more complex interaction 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.

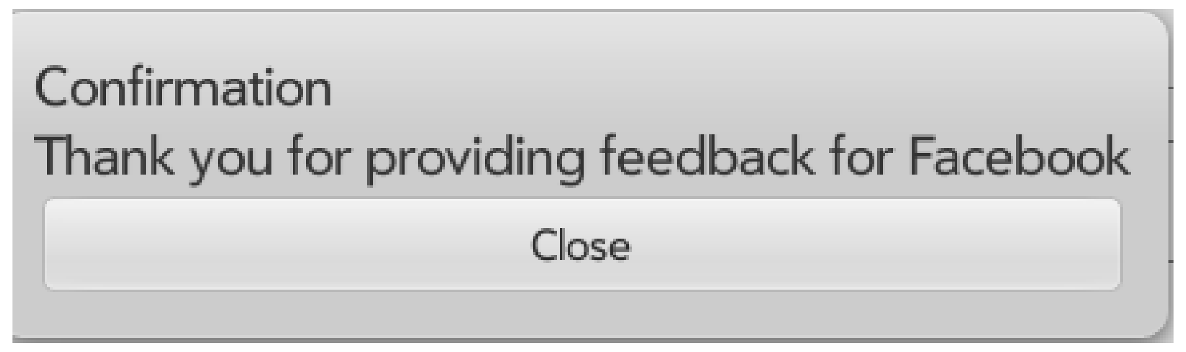
DIALOGS: EXAMPLES

CONFIRMATION DIALOG



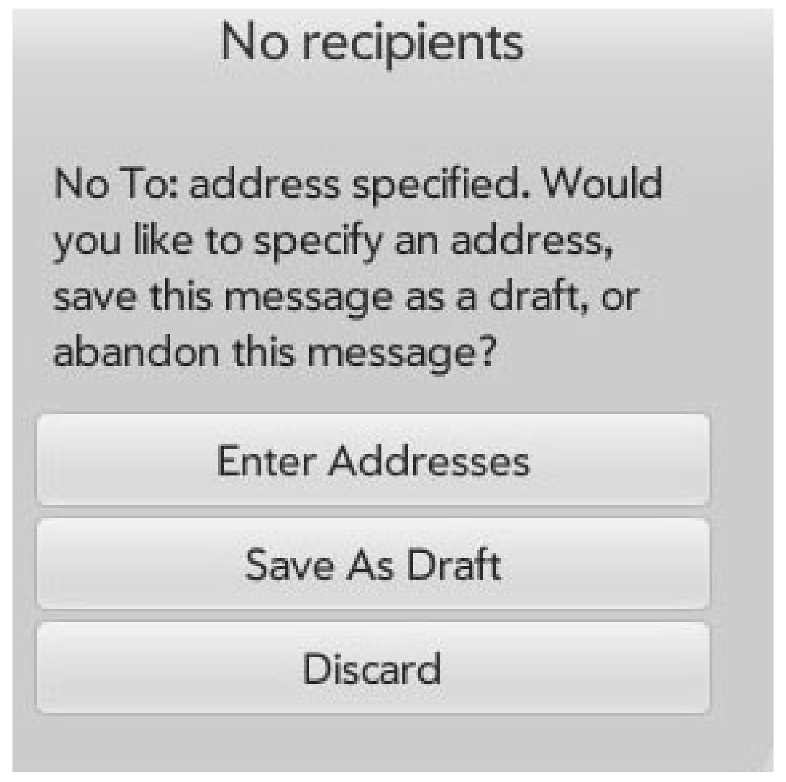
DESCRIPTION
- Confirmation dialogs enable users to confirm an action. Confirmation dialogs are typically shown only for irreversible actions.
- Confirmation dialogs should contain two buttons: a cancel button shown on the left or bottom, and a button to confirm the action on the right or top.

INFORMATIVE DIALOG



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

ACTION DIALOG


No recipients
No To: address specified. Would you like to specify an address, save this message as a draft, or abandon this message?
Enter Addresses
Save As Draft
Discard

DESCRIPTION
- 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 or other 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 app can publish a notification to the user if the app is running in the background or even if the app's card has been dismissed altogether by the user from the card view. Three types of notification can be published by an app:

• 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—Same as a banner-only notification, but after the banner text has been displayed, an icon from the app 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.

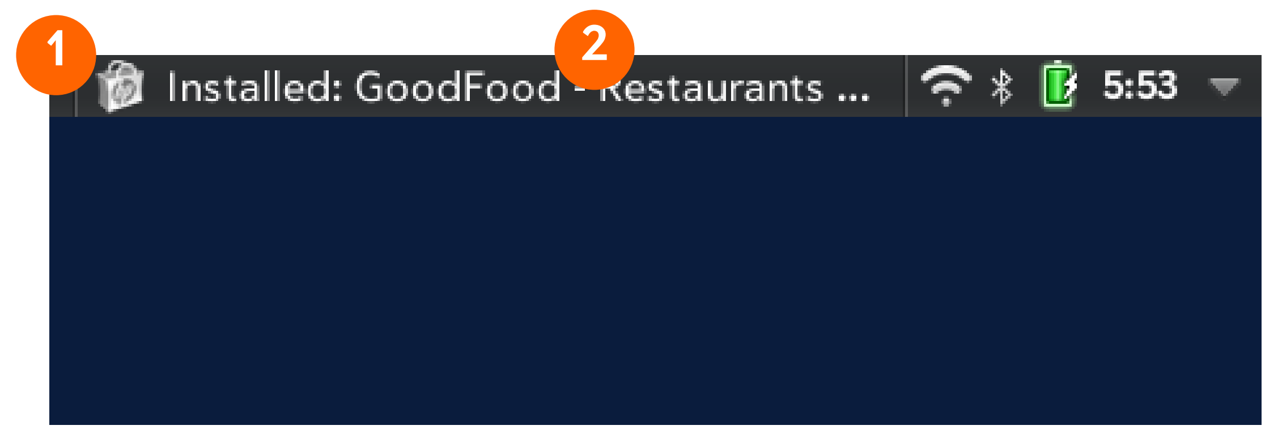
When a notification is received, the app can specify whether a sound is played by the device, or whether the device vibra is activated, or both. The app can customize the sound that is played by the device.

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

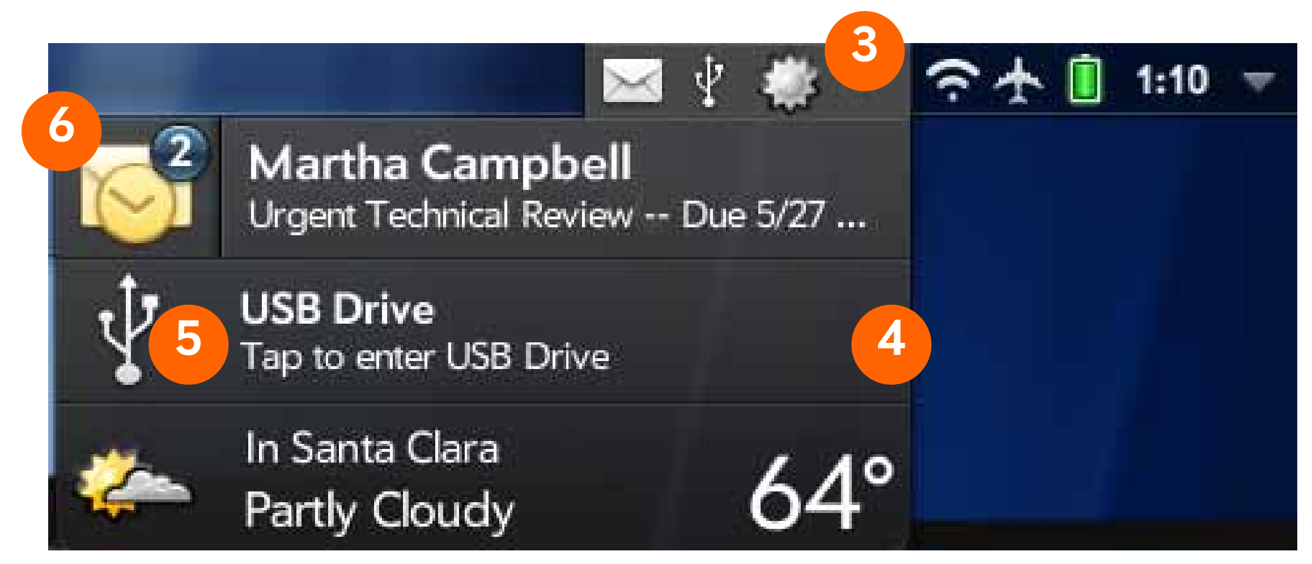
WHEN TO USE

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

Banner



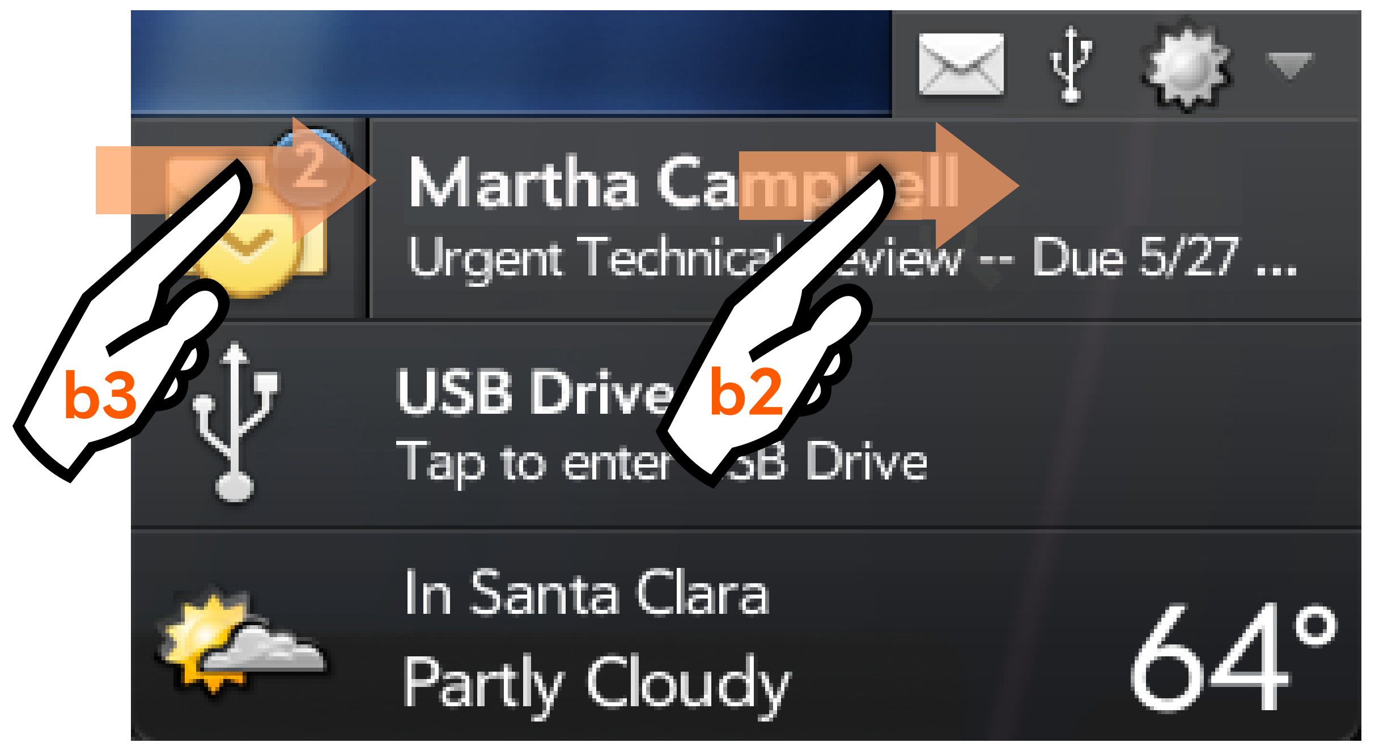
Dashboard



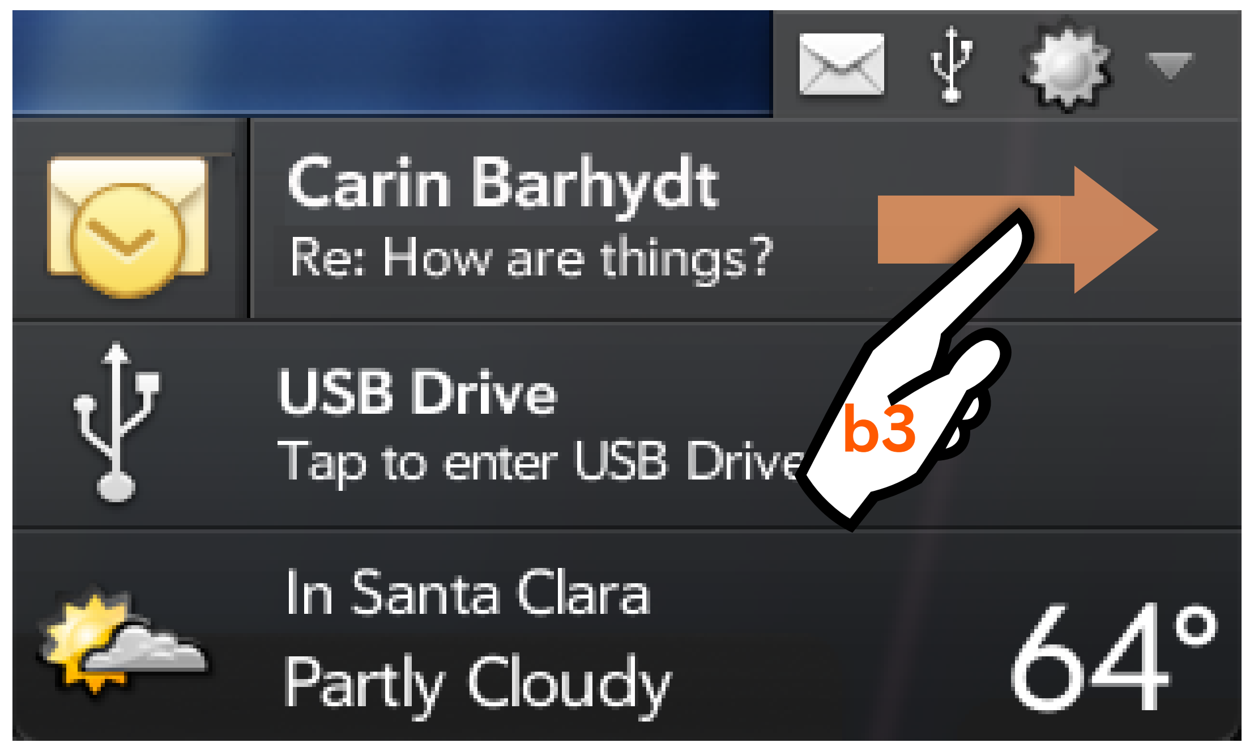
1	Status bar	4	Dashboard
2	Banner Text	5	App icon
3	Icon Summary Area	6	Grouped notification

GROUPED NOTIFICATIONS

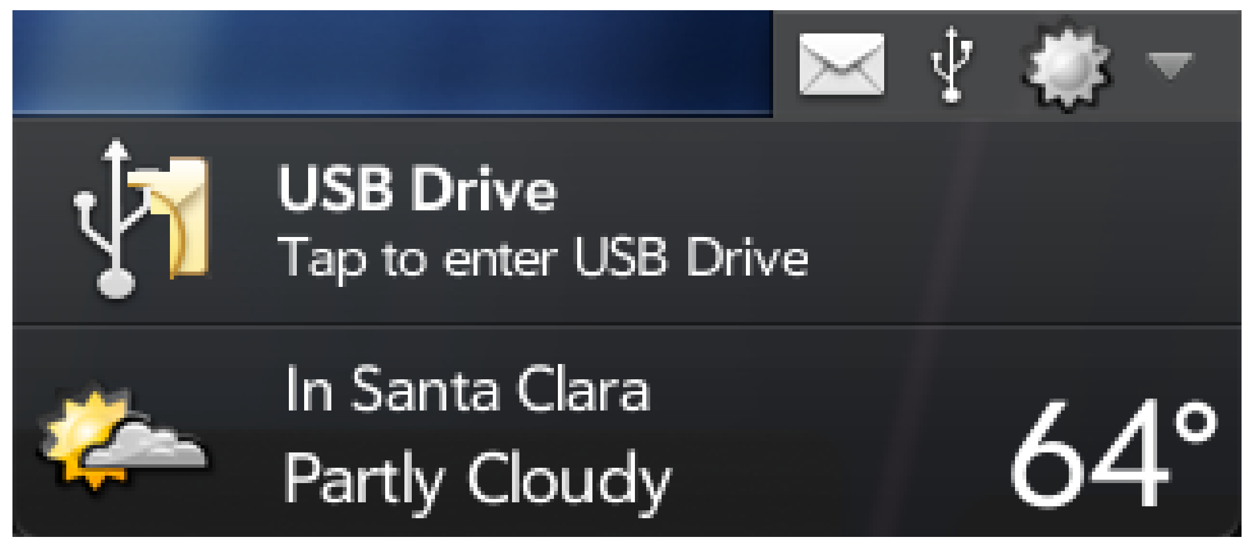
b1. Group of 2 notifications



b2. Single notifications



b3. Grouped notification dismissed



- The user can dismiss individual notifications by swiping them away to the right.
- Users can dismiss grouped notifications as a whole by swiping away the app 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 appear above app content until actively dismissed by the user or until the alerts time-out. The time-out schedule can be configured by the app that triggers the alert.

As with notifications, an app can publish an alert to the user if the app is running in the background or even if the app'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 app; however, please note the following guidelines:

• Include an app icon to inform the user which app 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.

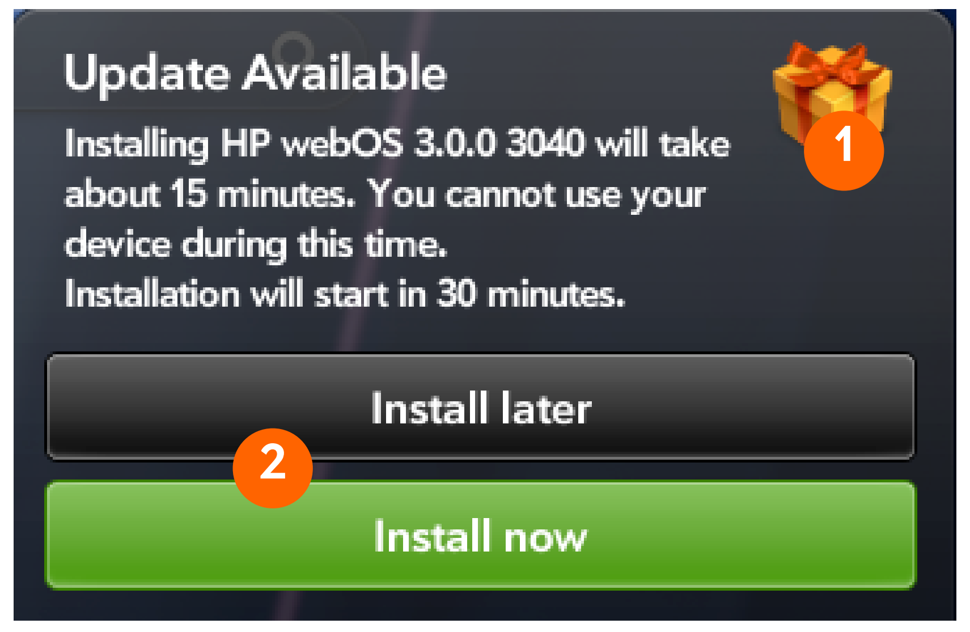
As with notifications, when a notification is received, the app can specify whether a sound is played by the device, or whether the device vibra is activated, or both. The app can customize the sound that is played by the device.

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



Update Available

Installing HP webOS 3.0.0 3040 will take about 15 minutes. You cannot use your device during this time.
Installation will start in 30 minutes.

1 App icon

2 Action buttons

APP MENU

WHAT IT IS

The app menu is an app-specific menu that is available from any view on the device. The app 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 app chrome. The app should provide an app name that is displayed in the title area. The app 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 app. If some menu items are not applicable in the context from which the app is launched, the item should be grayed out.

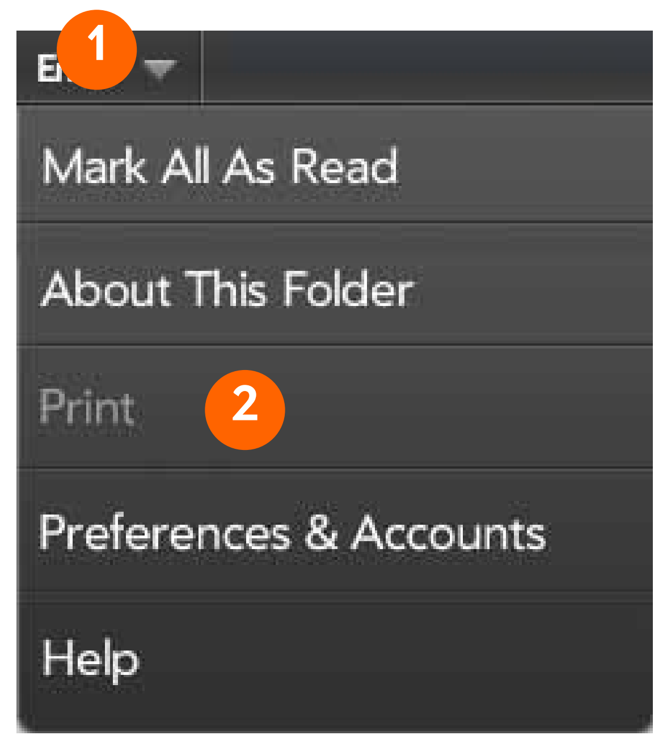
The app menu also contains Global Commands. These are common commands that are available across different apps. The Global commands are shown beneath the other commands in the following order:

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

WHEN TO USE

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

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



1 Title area

2 Disabled command

DESIGNING YOUR APP

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

THE DNA OF WEBOS

As a webOS developer, focus on what is important to users. HP webOS is built on these key principles. Following these helps to ensure an effortless and engaging user experience for your app:

- **Simple, effortless design...with a focus on enabling important features for users.** The user interface design is focussed on allowing users to accomplish their core activities with the minimal number of steps. Find the essence of your app and design it so that essence is obvious to the user and its core activities are immediately accessible.

- **Spatial metaphors ....with intuitive, familiar interactions that just make sense.** webOS uses spatial metaphors to make interacting with the system feel familiar and natural. Users flick between different cards that represent different activities and drag panes to navigate through apps. Let users manipulate objects in your app directly using gestures so that managing activities feels easy and natural.

- **Unobtrusive notifications....that don't take away user's focus.** Notifications are always visible, so users never miss a thing while working, playing, or relaxing with webOS apps. At the same time, a notification is discrete so that it doesn't disturb the user and allows the user to decide when they want to act on the notifications.

- **Synchronized data...makes it easier to find information with minimal navigation.** Information from multiple sources and accounts are seamlessly integrated together. Users can discover and use information without opening multiple views. Use the power of webOS to harness data in your app into a single, coherent experience.

- **Seamless multitasking...that allows different activities to be linked together effortlessly**. Multitasking is built into the core of webOS. The device's Card view allows the user to effortlessly move between different activities when using their device, either within a single app or across multiple apps. Users are not constrained to following a linear path but can move concurrently between different activities when managing their daily lives.

- **Protecting the user's data...so that apps are safe and reliable**. Protecting the user's data is a fundamental need. Apps should automatically save data so that the user should never fear for losing their work even as a result of unexpected app closures. Users should be given sufficient warning for destructive actions that could erase data irreversibly. When tasking away from an app, the app should save state so that the user can carry on from where they left when they return to the card.

DESIGN PRINCIPLES

Focus on benefits to users, not technical features

- Provide an efficient and balanced set of features to your app.
- When considering features, always weigh what could be gained in terms of user benefits with the added complexity that will be brought to your app.
- Start with only the essential set of features and add features only until you reach the sweet spot that provides the optimal user experience, and no more.

Streamline interactions

- Streamline your design to support key use cases.
- Common tasks should be accomplished with the minimum number of steps. If you can make the most common functions accessible by just one touch or one tap, users will love your app.
- Consider how users will hold the device— important actions should be visible and easy to reach while holding the device comfortably. See the guidance on *Laying Out Content and Controls.*
- Design your app layouts to minimize clutter—focus attention around the key activities and objects and place less frequent or important activities a tap away.

Familiarity leads to intuitiveness

- Build upon established and familiar interaction patterns to make your app feel natural and intuitive.
- Use the built-in webOS UI components and follow these design guidelines wherever possible.
- Using familiar components and patterns gives your app an instant sense of familiarity and usability.

Pleasantly surprise experienced users

- Once you have ensured that key interactions are highly accessible and easy to use, layer in shortcuts to allow more experienced users ways to more quickly accomplish common tasks.
- Adding these shortcuts should not be at the expense of creating visual clutter, nor should shortcuts be the primary or only way to accomplish a task.
- Rather, these shortcuts should become discoverable in time and provide a pleasant surprise to users as they gain experience with your app.

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

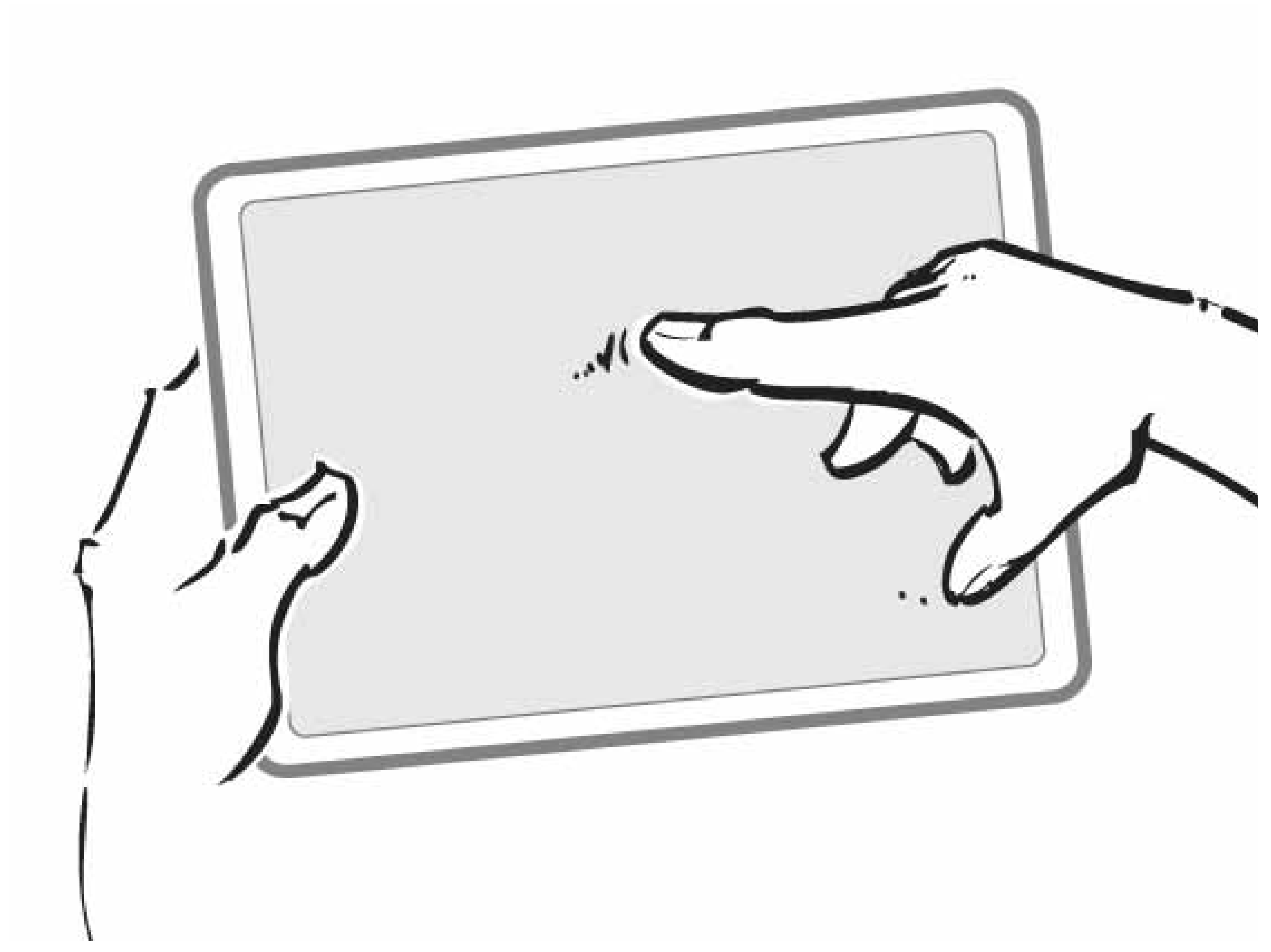
- Designing for a tablet form factor is not simply an exercise in scaling a phone app to a larger screen, but instead a bona fide new form of design, with its own particular contexts of use, ergonomics, and design considerations, which should be borne in mind when designing an app on the tablet.
- Tablet apps should generally provide a larger-scale experience than on phones. A user should not have to rely on a separate device, such as a PC, to complete tasks.
- The following page highlights other key differences in designing for phones and designing for tablets.

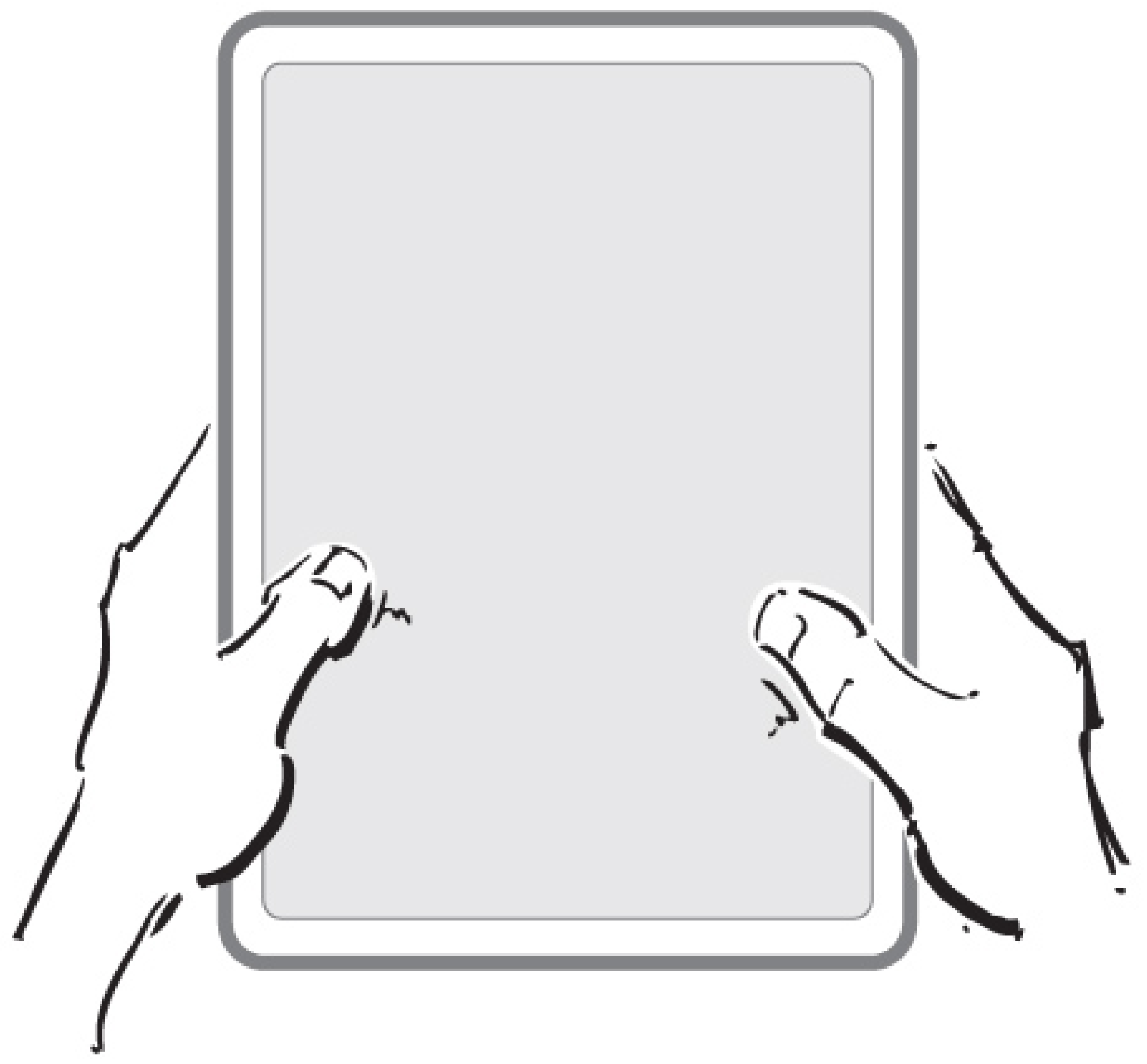
KEY DESIGN DIFFERENCES BETWEEN PHONE AND TABLET INTERACTIONS

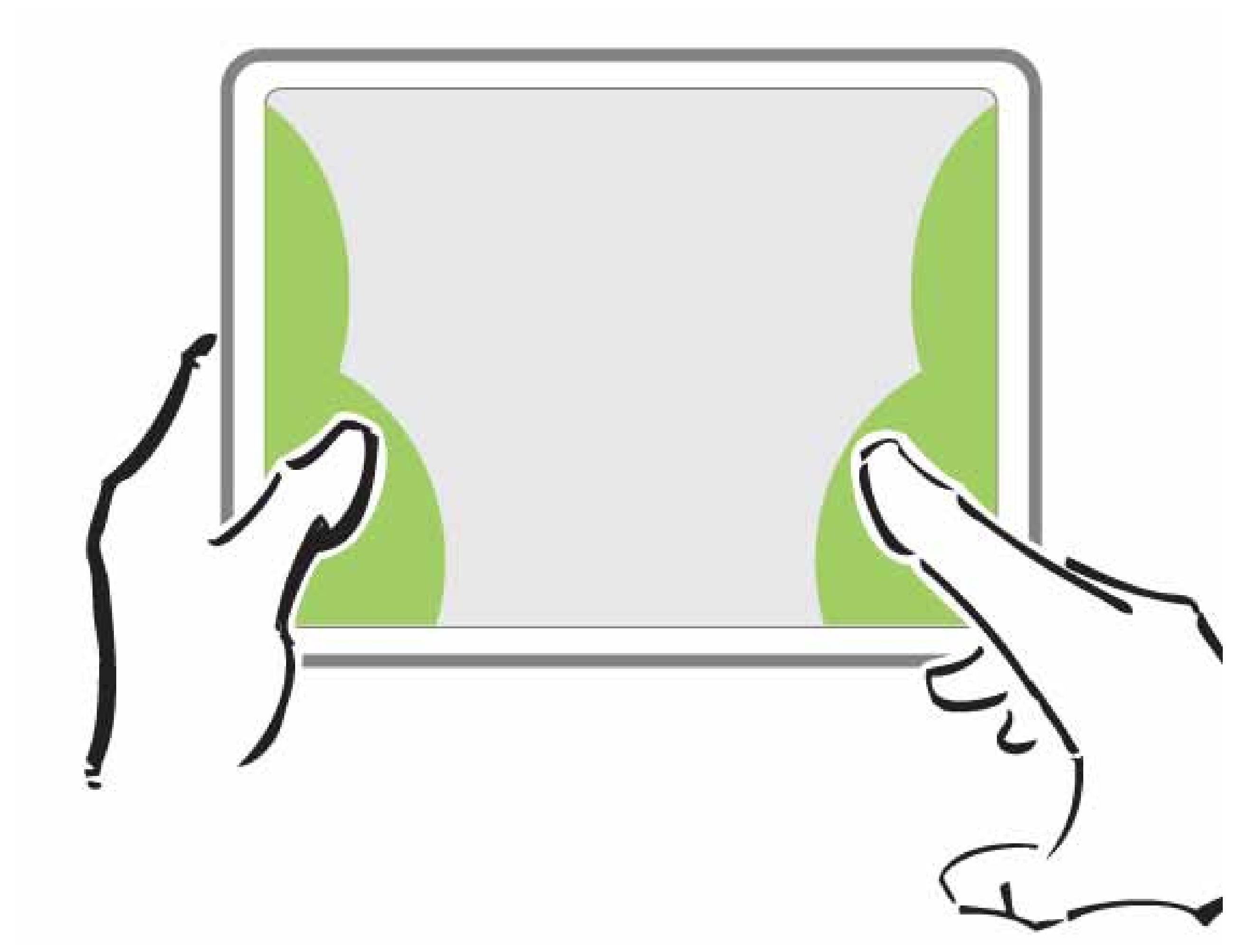
PHONES	TABLETS
Single, focused area for content.	Potential for **multiple areas** of content where appropriate.
Limited on-screen actions, exposing only highest priority functions.	Ability to **expose a greater number of on-screen actions** that are high priority.
Mainly **list-based navigation** optimized for the small screen.	Potential for more fun, intuitive navigation with **greater emphasis on visual objects**.
Navigation through **many** screens of content.	Navigation between screens kept to a **minimum**. Content can be displayed in multiple panes or brought in via interactive pop-ups.

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

Consider how people hold the TouchPad







TAP TARGETS

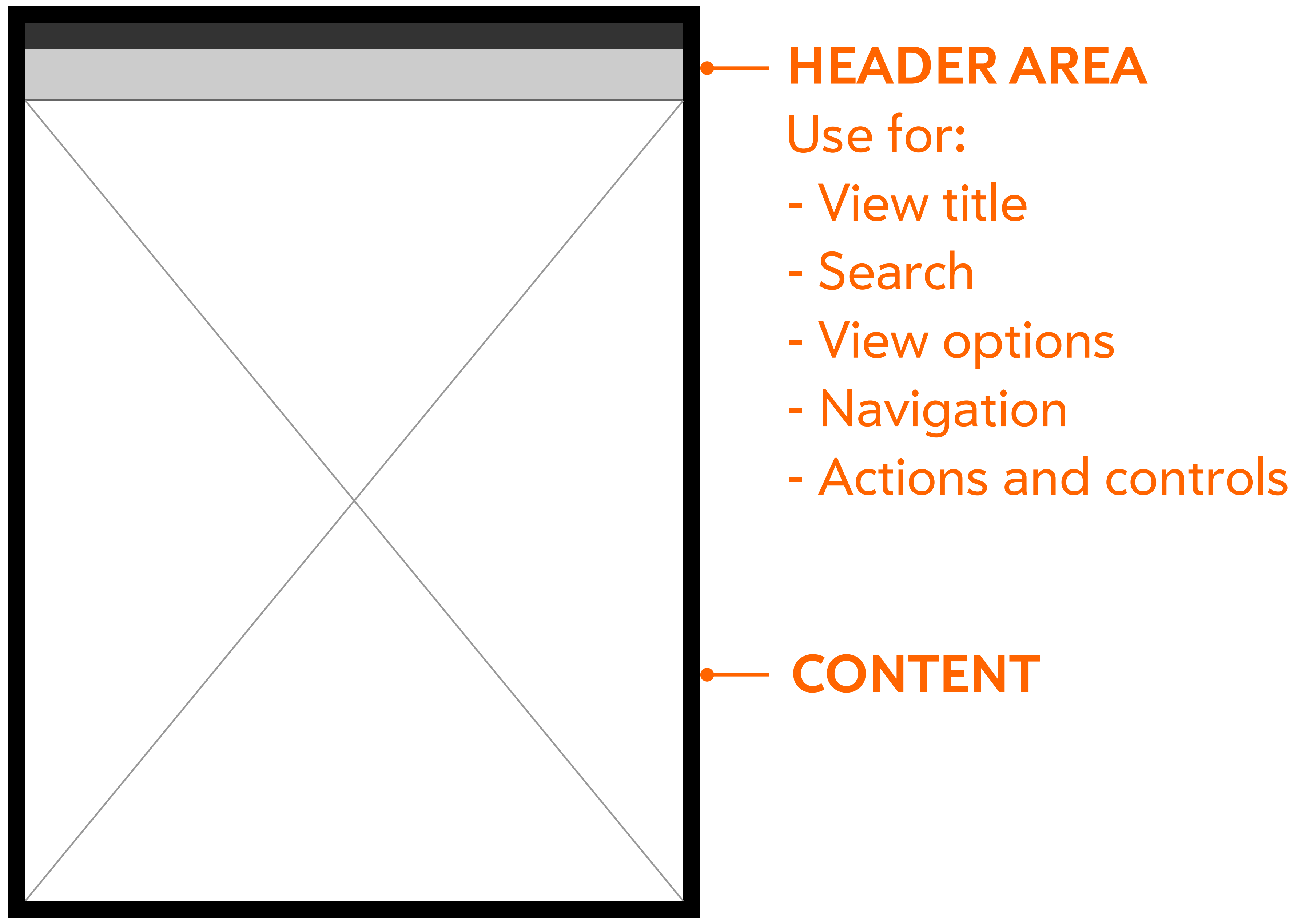
- Tap targets should be larger on tablets than on phones.
- Standard webOS controls on the TouchPad are 54px square. We recommend using standard controls wherever possible as these components have been designed specifically for use on the TouchPad and using these components will also lend familiarity to your application.
- When creating customs controls, however, we recommend that any tap targets should also be **at least 54px square.**
- Tapping on any tappable targets should provide immediate visual feedback to the user. Tap targets should be designed so that they have a pressed state when they are tapped - see the section on *Providing Feedback* for more details.

POSITIONING CONTROLS

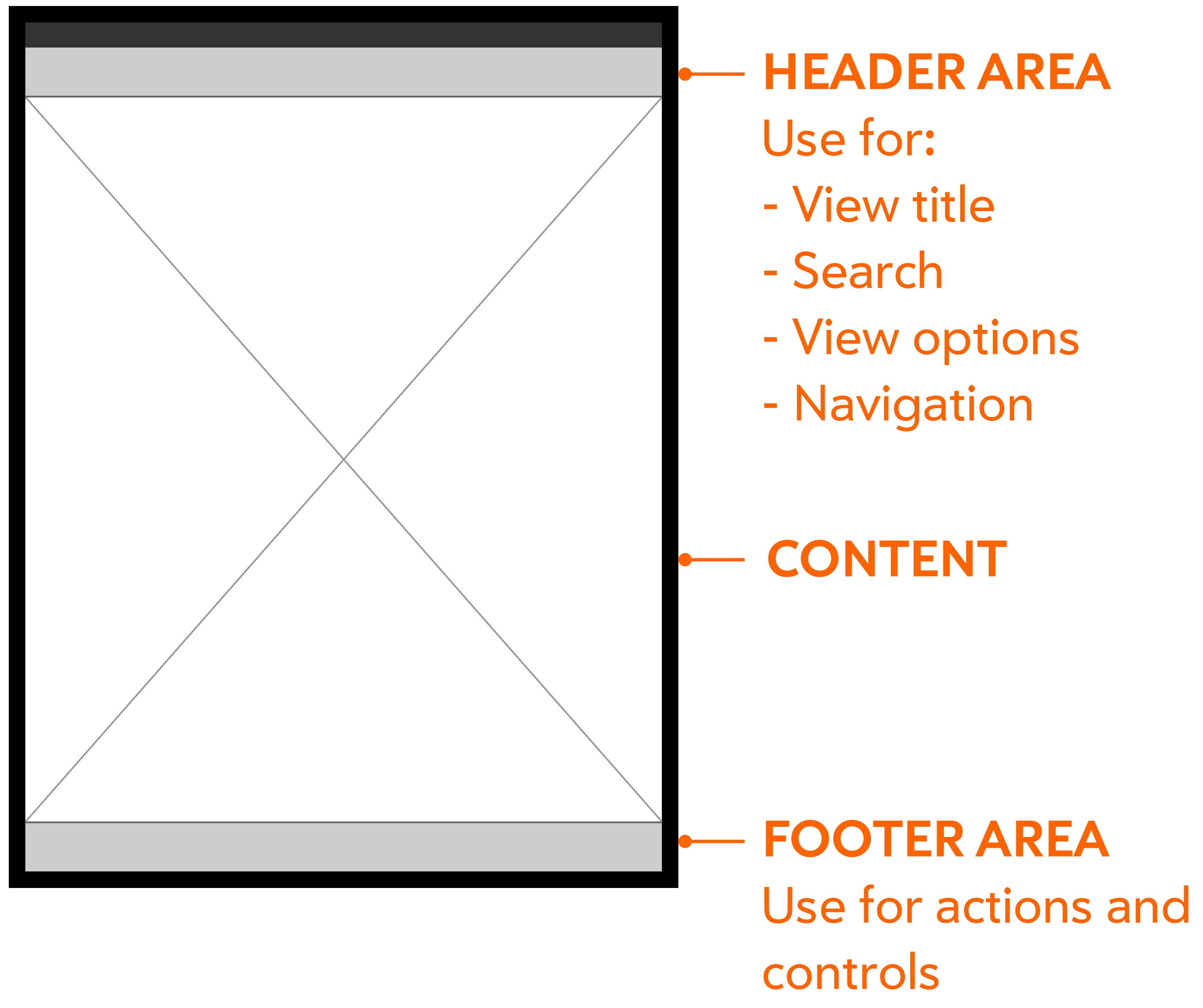
- The TouchPad has no conventional grip.
- People can hold the TouchPad either one- or two-handed in either orientation.
- It is harder to predict where to place controls on a tablet on a phone - however, consider placing controls on the side as these are easily thumb-able areas.
- Avoid placing destructive controls on the sides as these are areas are also easy to tap by mistake when gripping the device.

Create logical groupings of navigation and action controls

PANE WITH HEADER ONLY

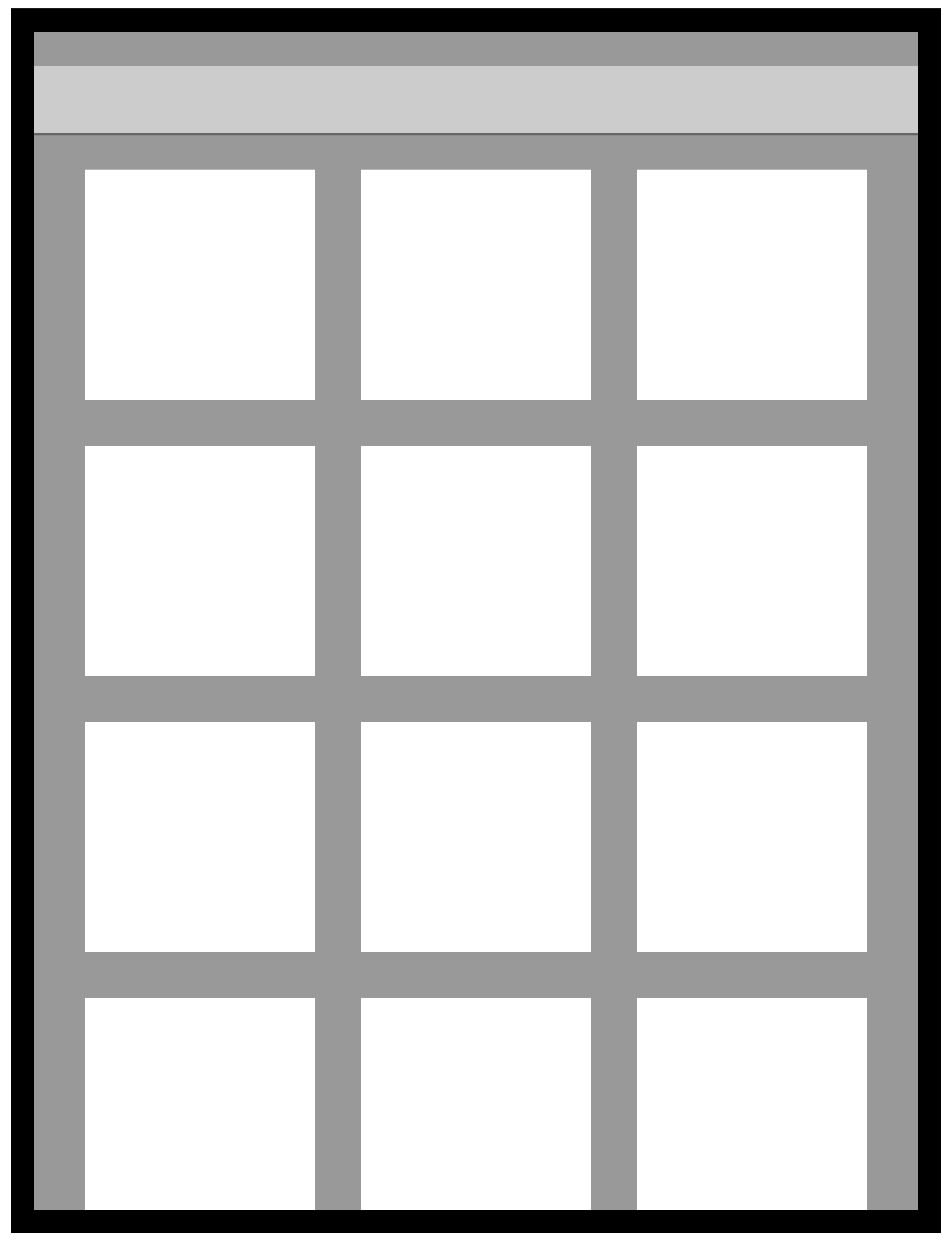


PANE WITH HEADER AND FOOTER ONLY



LAYOUTS: SINGLE PANE AND SPLIT VIEW

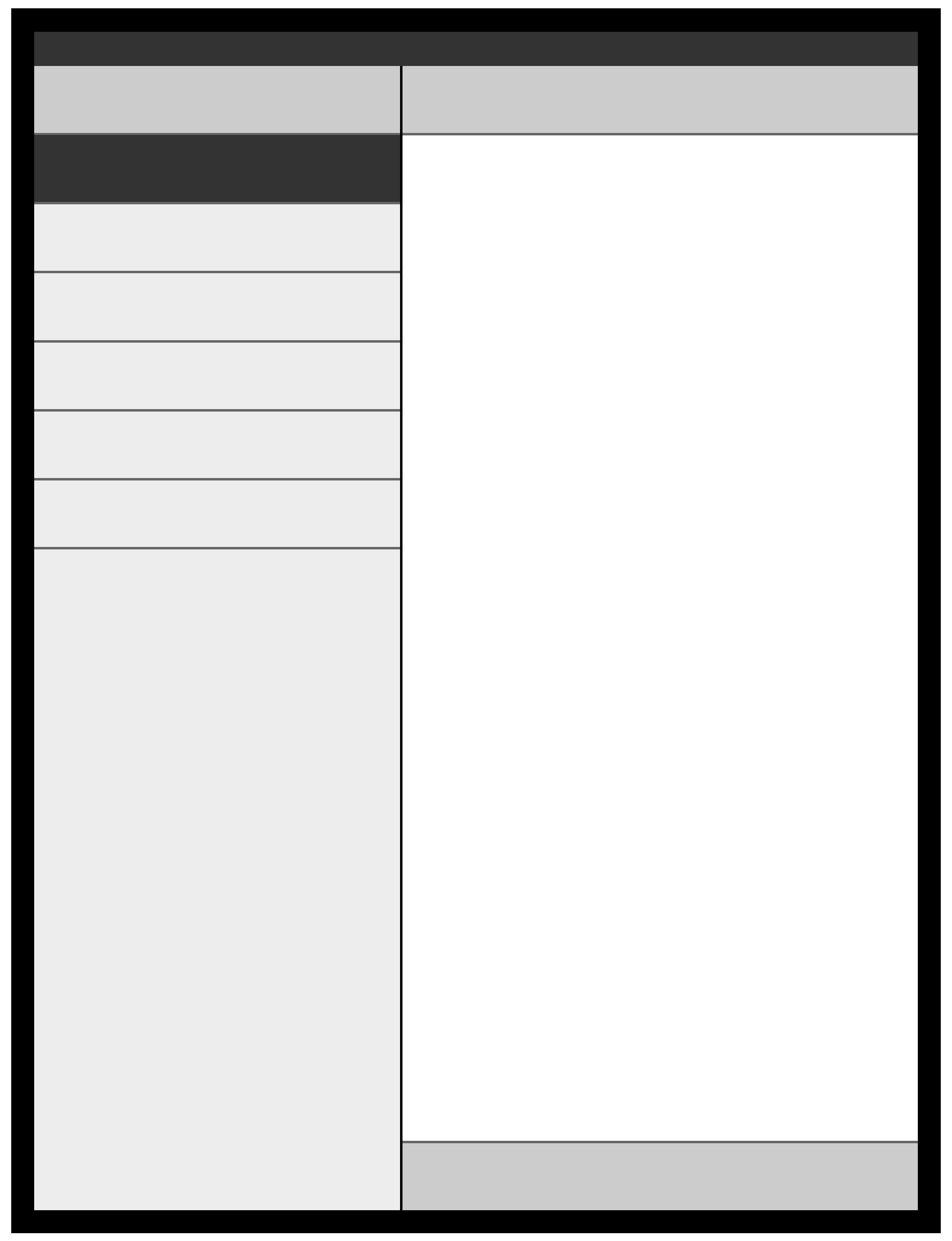
SINGLE PANE



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 app content.
  - Viewing options - for example, use buttons to select what content is displayed in the content area, such as grid versus 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

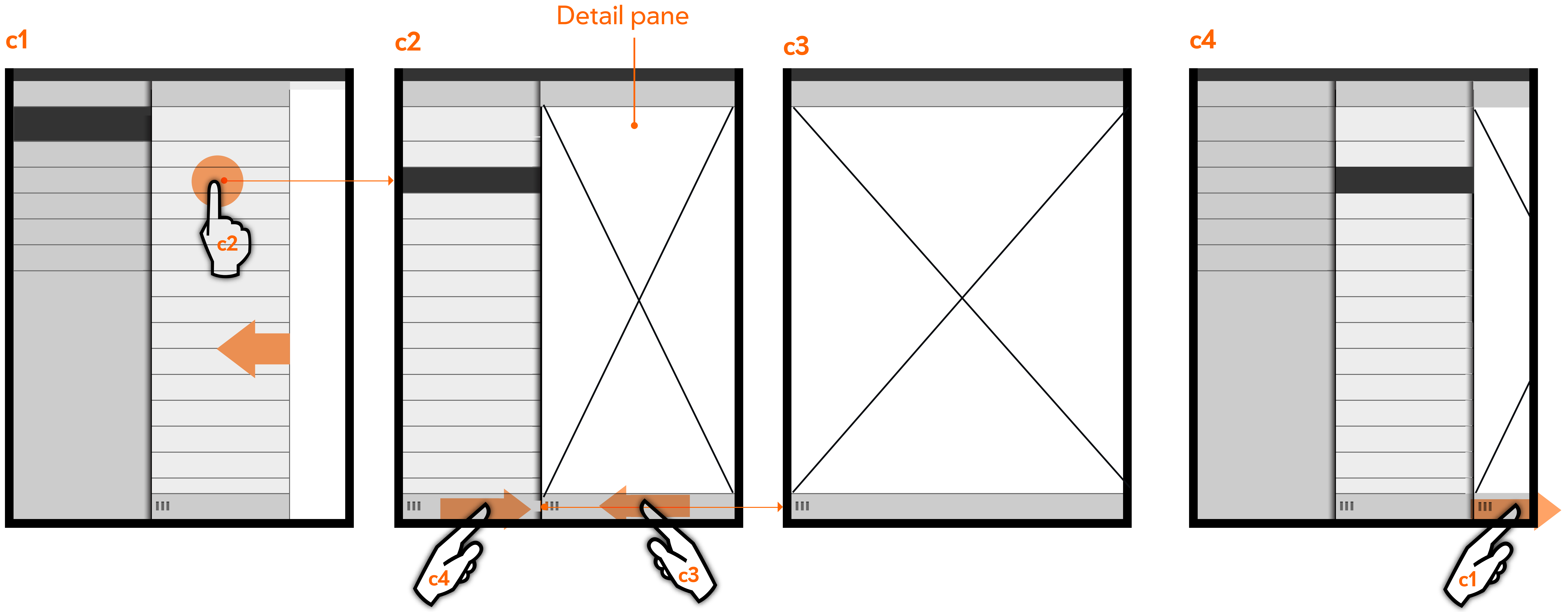


DESCRIPTION
- Use this layout to structure your app into two panes: the left hand pane is fixed in width whereas the content on the right hand pane is free and adjusts to fill the available space.
- The left pane contains a list of items. One item in the list should be selected and the right pane should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

APP EXAMPLES
- Photos & Videos

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.
- In landscape, up to three 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 slides 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 px 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.
- The user can dismiss the right-most pane by swiping it away from left-to-right.

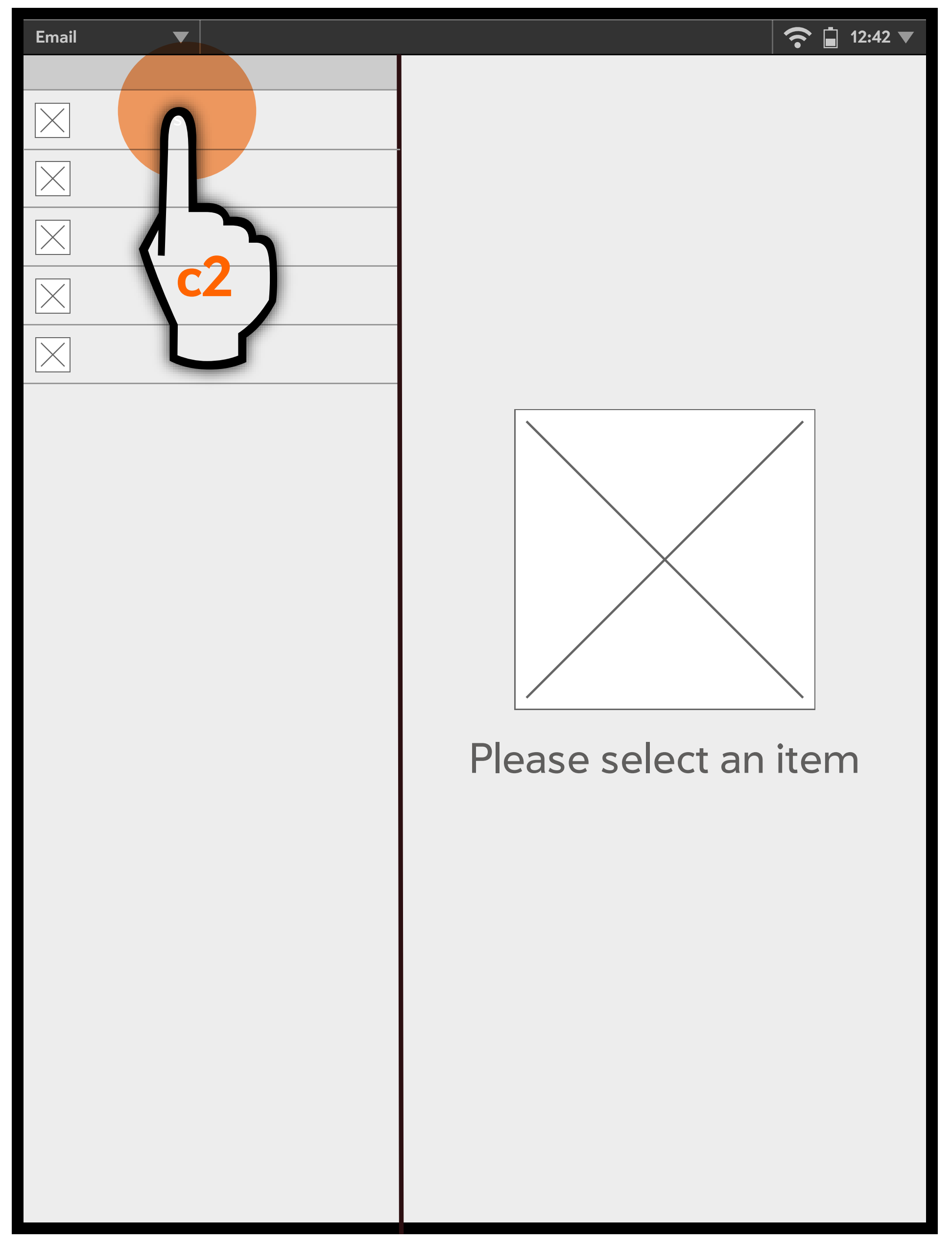
APP EXAMPLES

- Email

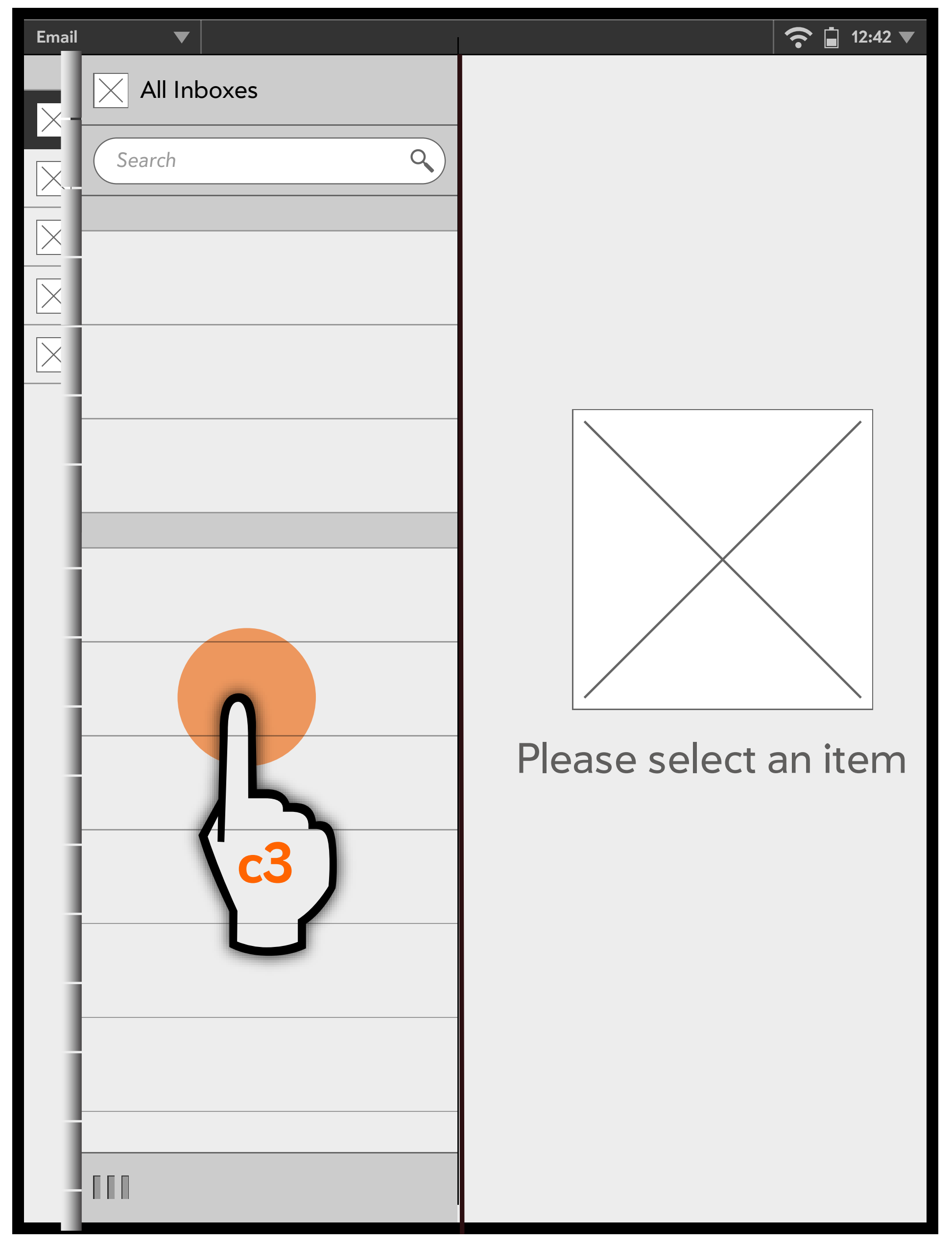
LAYOUTS: SLIDING PANES

PANE PEEKING

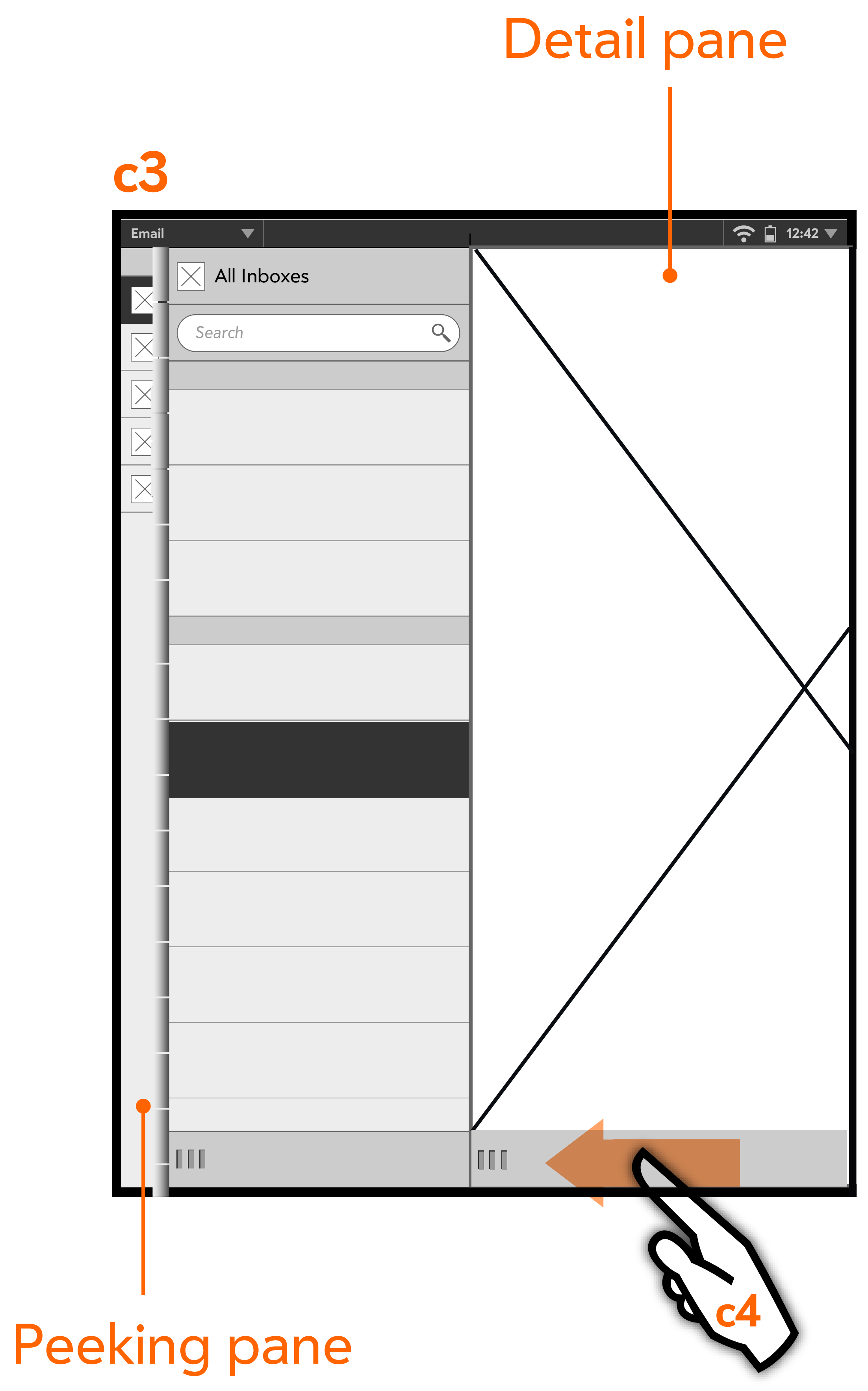
c1



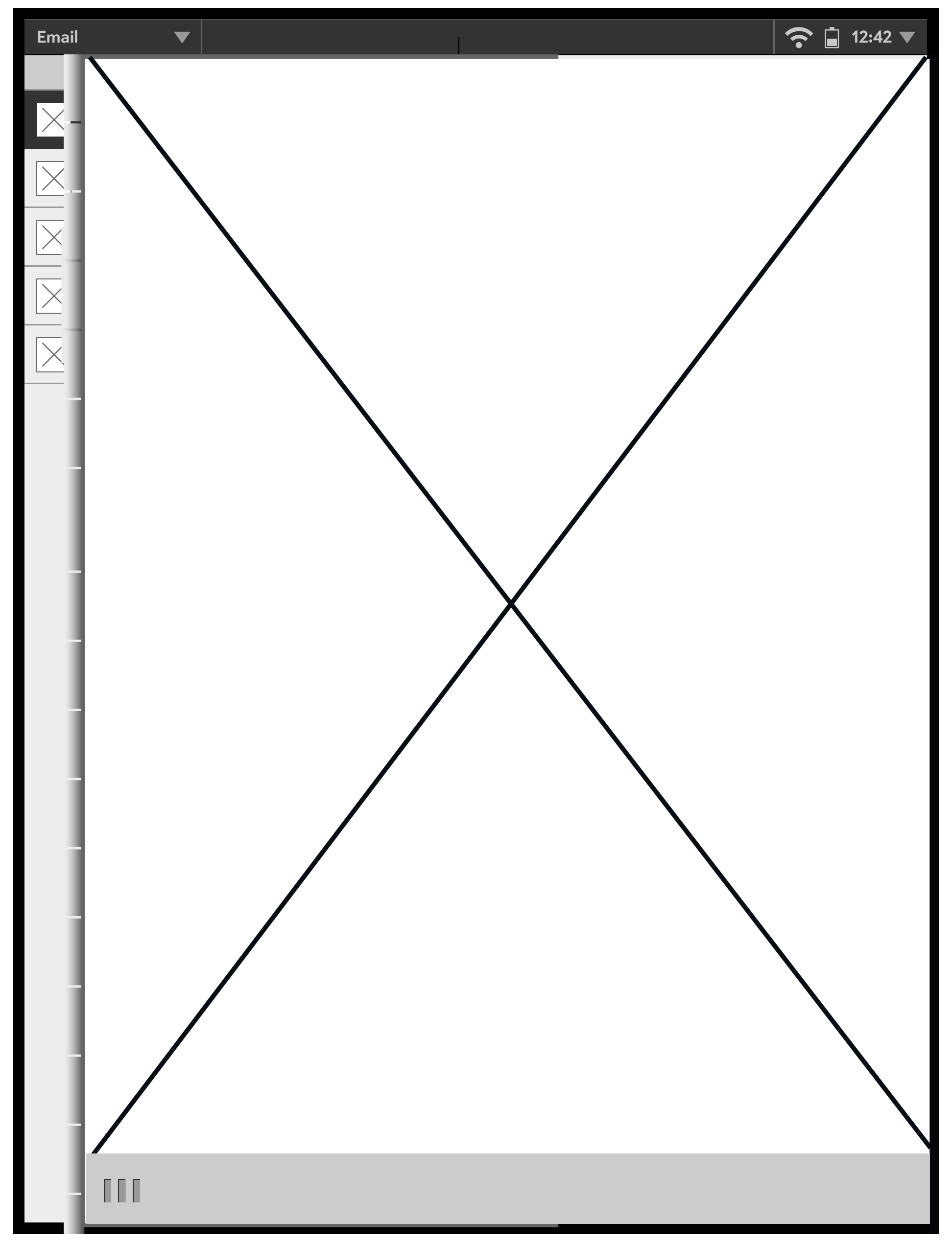
c2



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 the panes no matter how deep the user is within the hierarchy.

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

- Peeking pane has minimal content (no scrolling).
- Peeking pane does not have collapsing or expanding list items.
- Icons are easy to recognize.
- There is a strong use case for being able to always access content in the peeking pane.

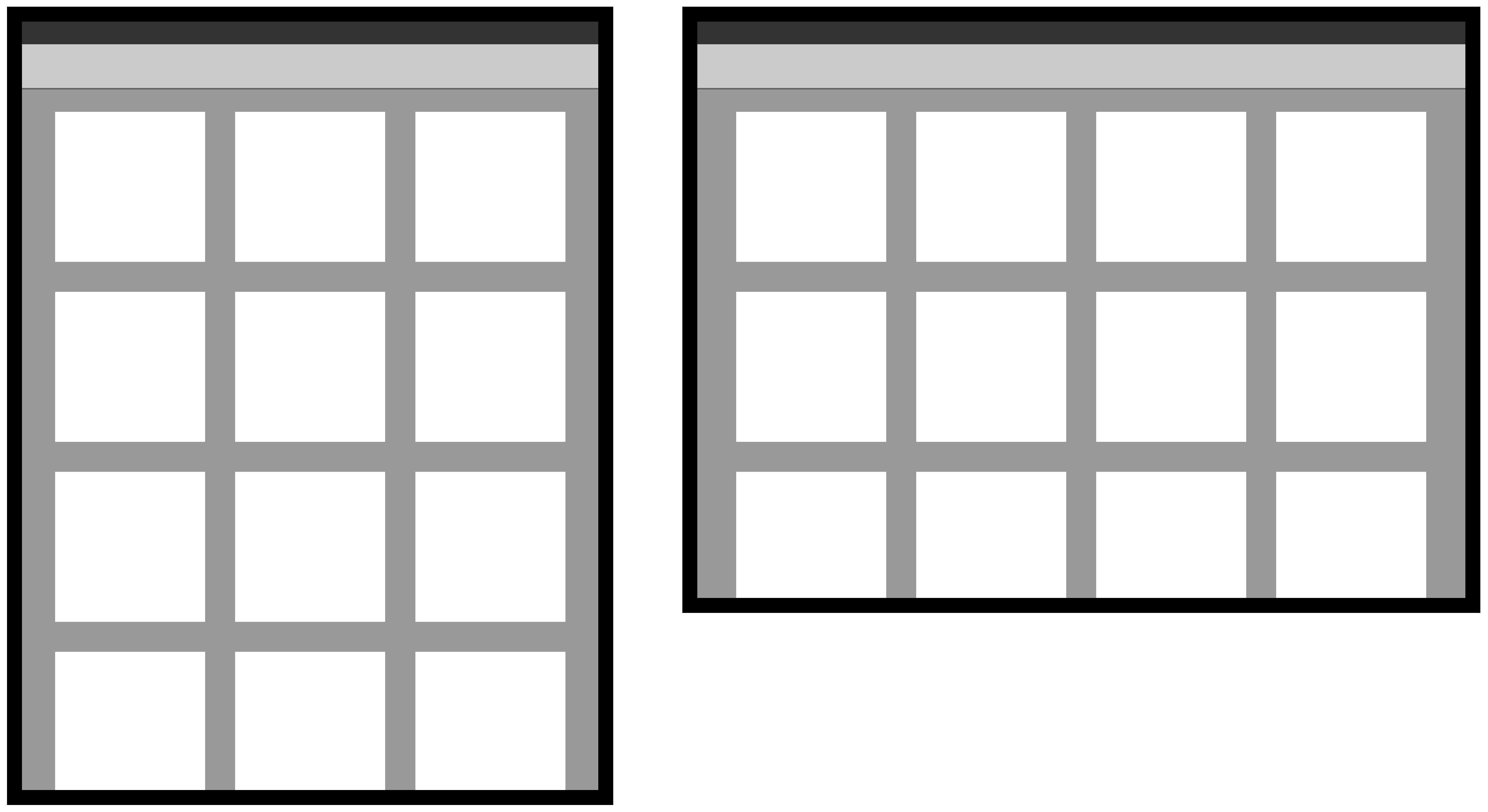
PORTRAIT AND LANDSCAPE LAYOUTS

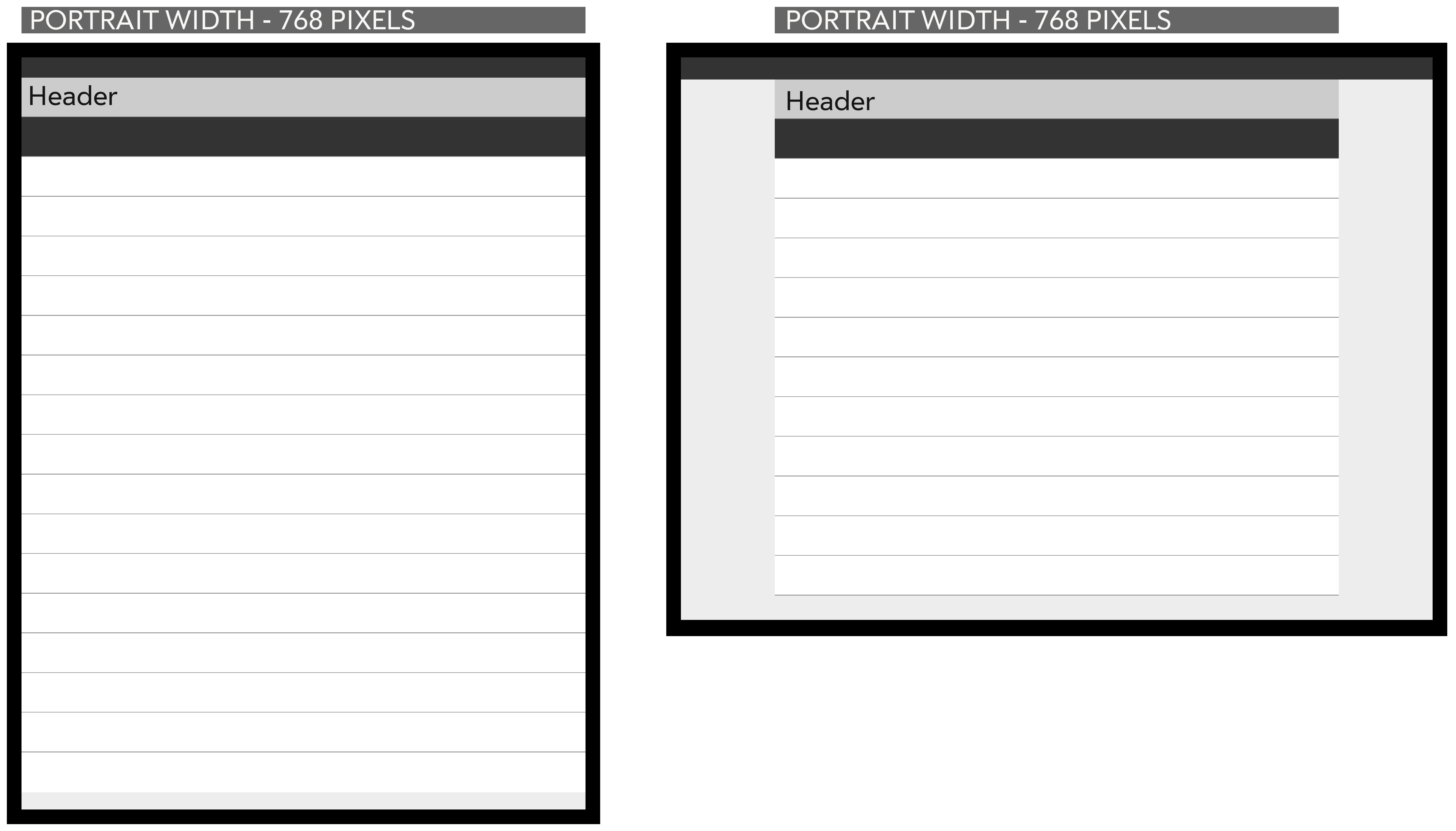
The device accelerometer can capture orientation events to detect whether the device is being held in a portrait or landscape orientation. Your app has the choice to constrain itself to a particular orientation, or to readjust its layout to match the device orientation between portrait and landscape. It is recommended, however, that apps generally provide both a portrait and landscape layout. As mentioned earlier, users hold tablets in a variety of ways and orientations, and your application should give users this flexibility.

Panes automatically re-size to fit the available width in the view unless the panes are fixed in width. Consider how your app content appears 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 in which the header or footer 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 not introduce new content or functionality.

SINGLE-PANE LAYOUTS





DESCRIPTION

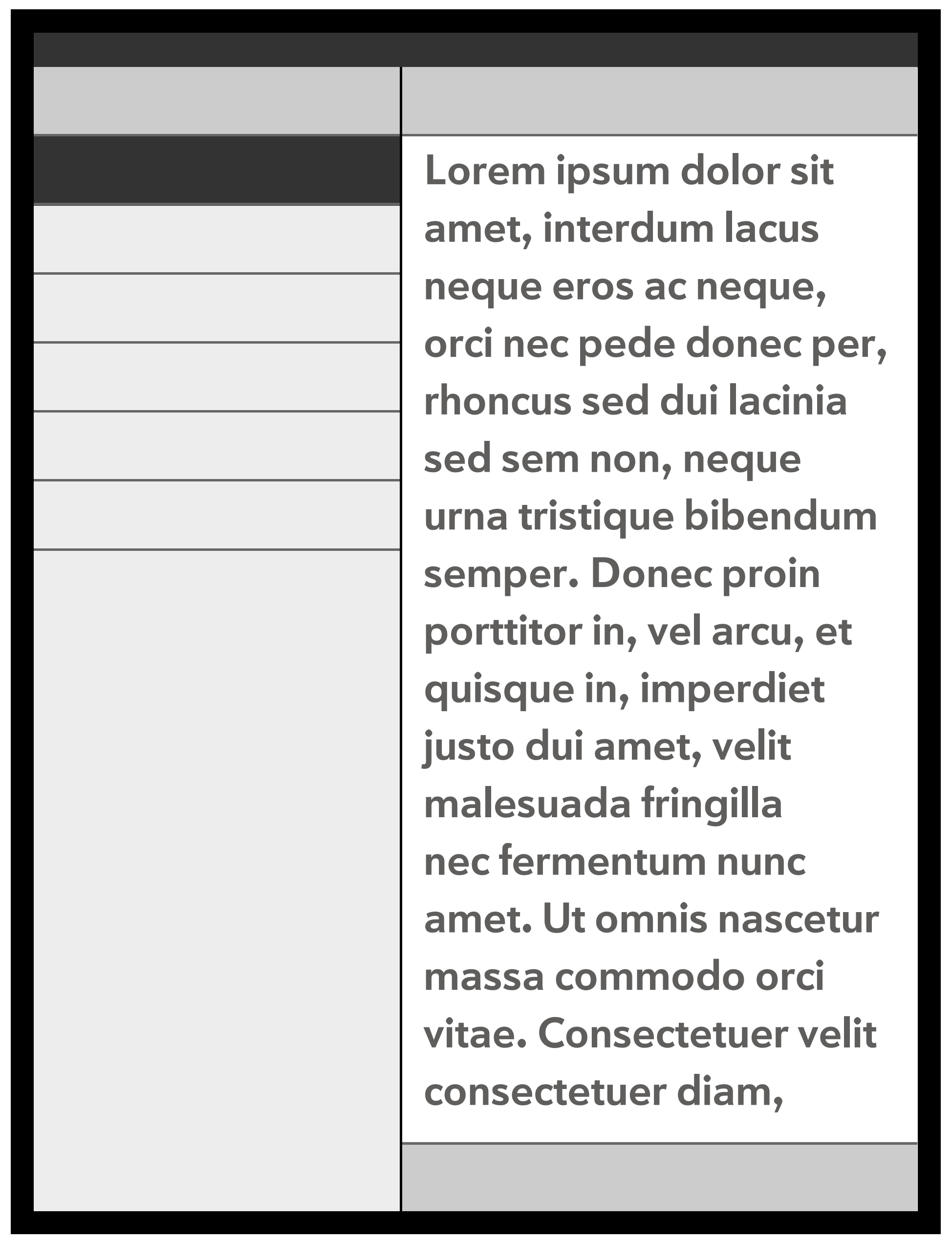
- Grid-based layouts scale well between portrait and landscape. A grid-based layout should re-order to fit as many grid items as possible within the horizontal space within the pane.

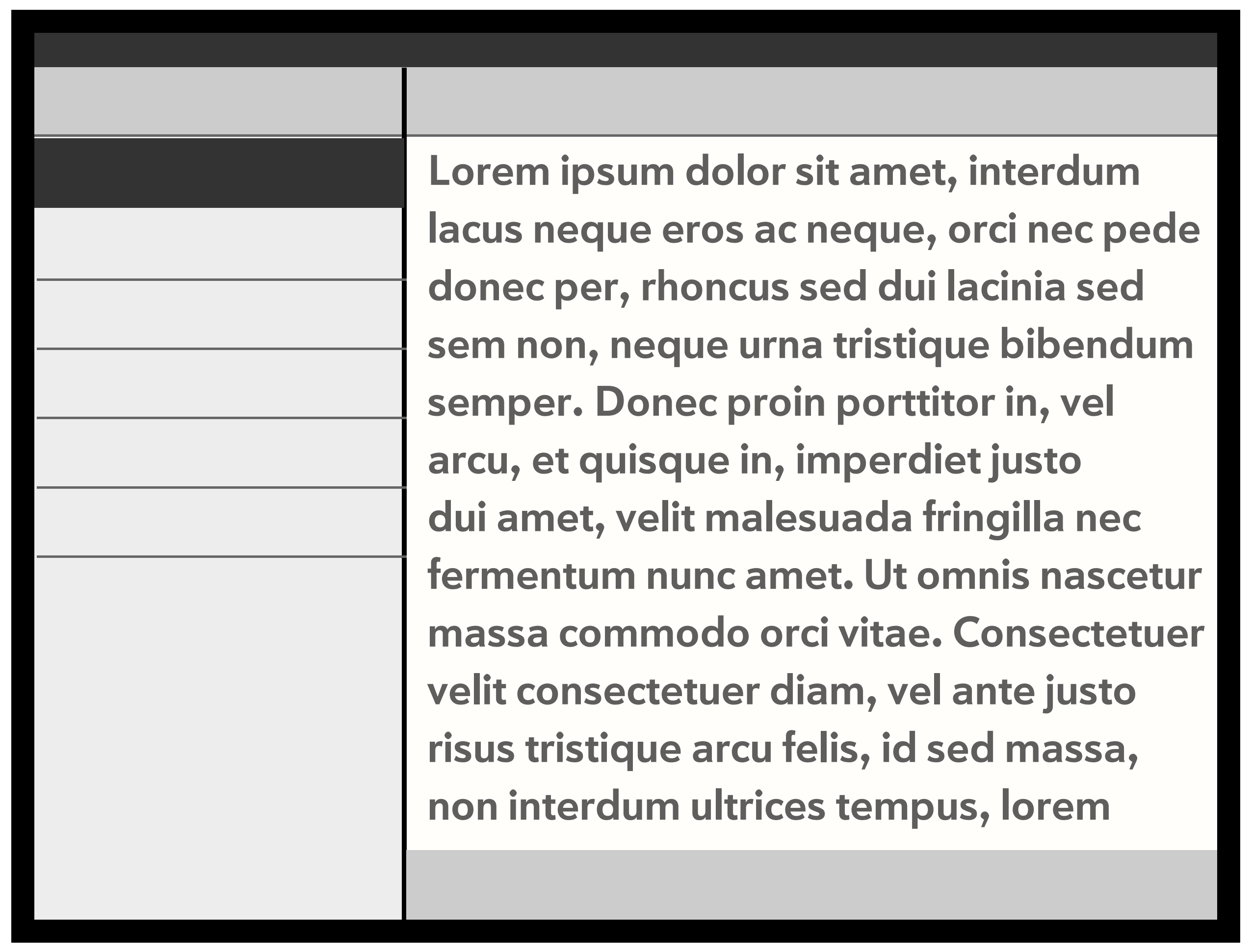
DESCRIPTION

- 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 remains the same between landscape and portrait orientations.

PORTRAIT AND LANDSCAPE LAYOUTS

SPLIT VIEW


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

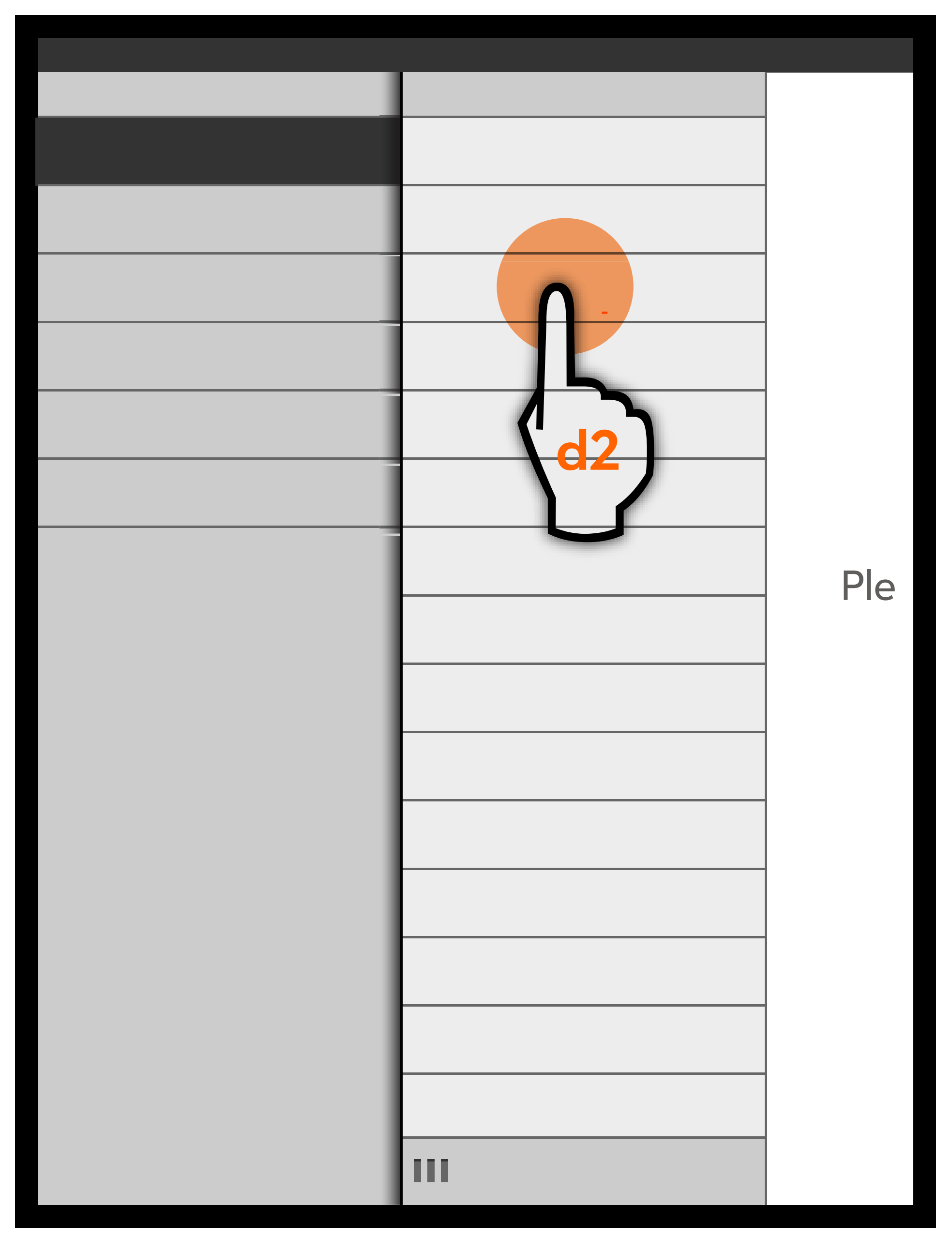
DESCRIPTION

- 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 typically change to a landscape orientation to see more content in this layout. Try to fulfill the user’s expectations by ensuring that the amount of content displayed in landscape is maximized.

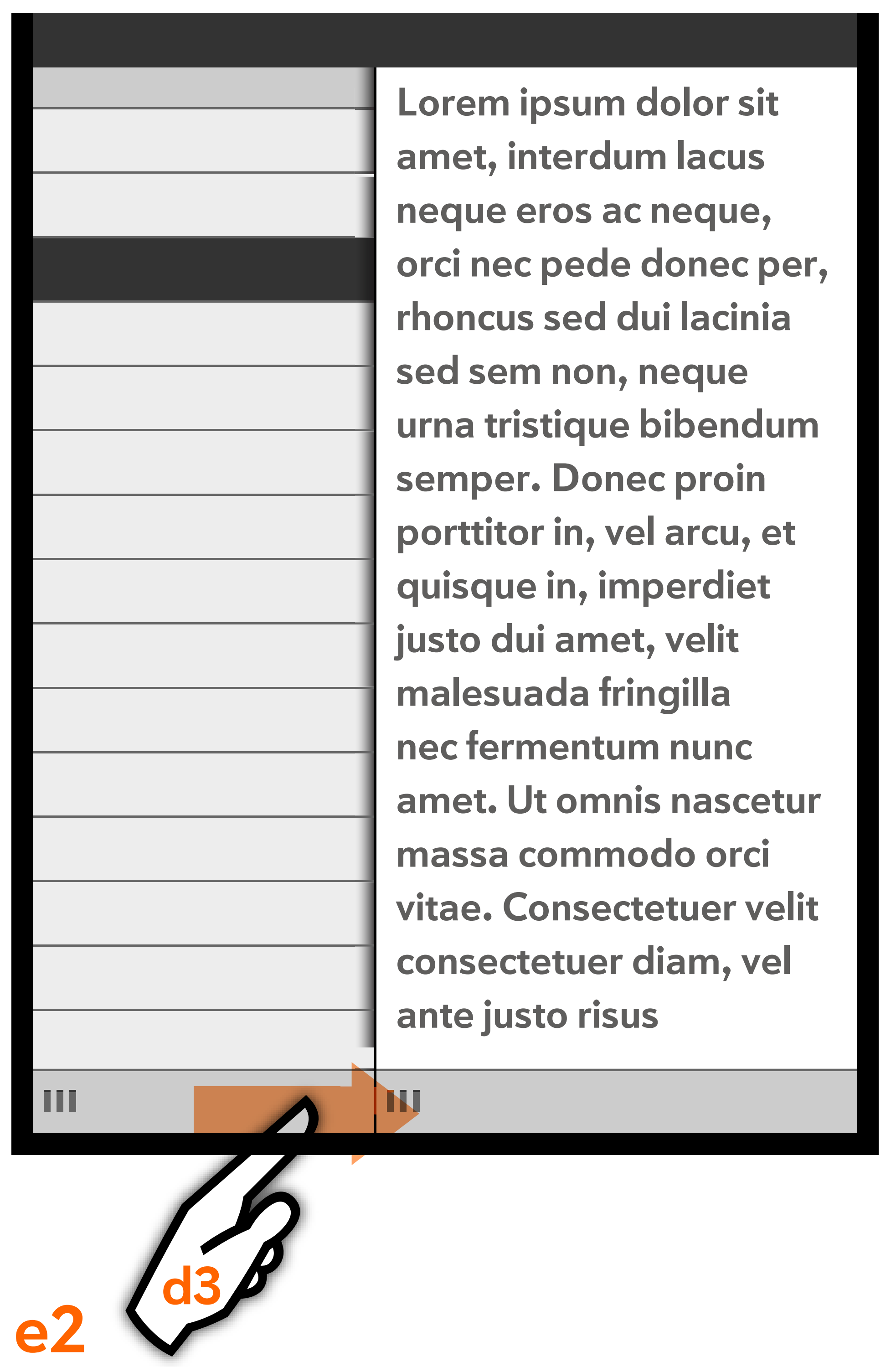
PORTRAIT AND LANDSCAPE LAYOUTS

SLIDING PANES

d1

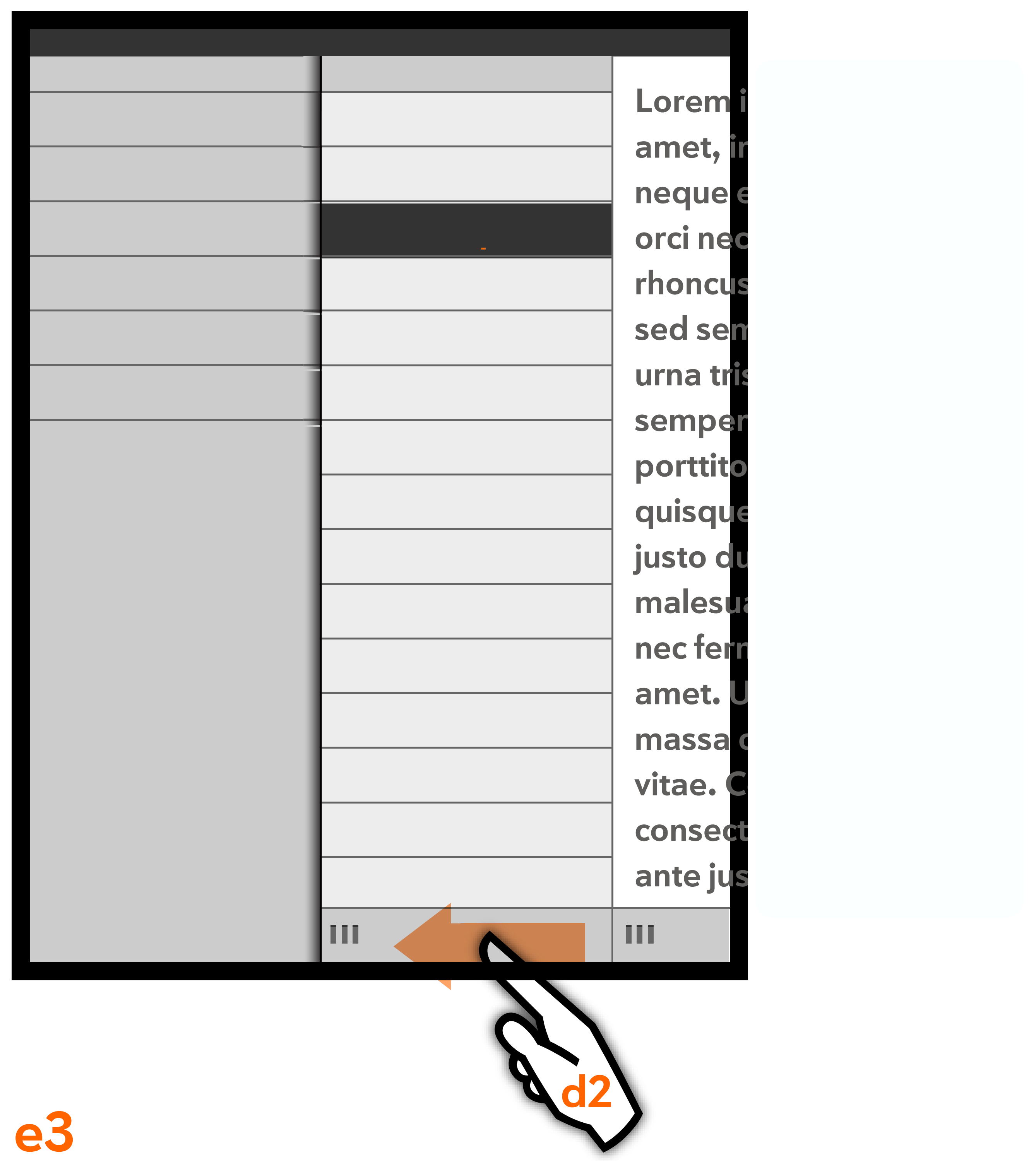


d2

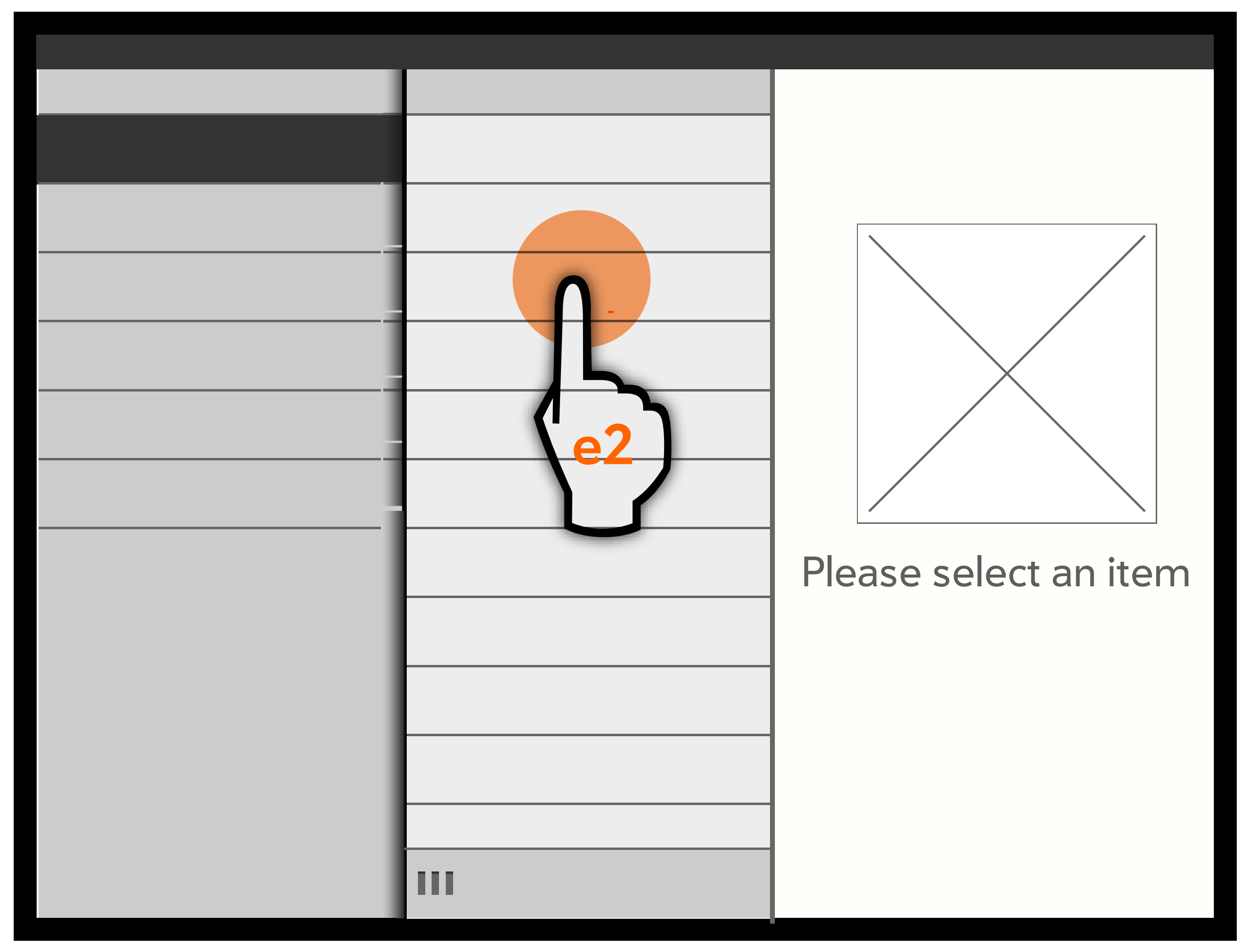


Lorem ipsum dolor sit amet, interdum lacus neque eros ac neque, orci nec pede donec per, rhoncus sed dui lacinia sed sem non, neque urna tristique bibendum semper. Donec proin porttitor in, vel arcu, et quisque in, imperdiet justo dui amet, velit malesuada fringilla nec fermentum nunc amet. Ut omnis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

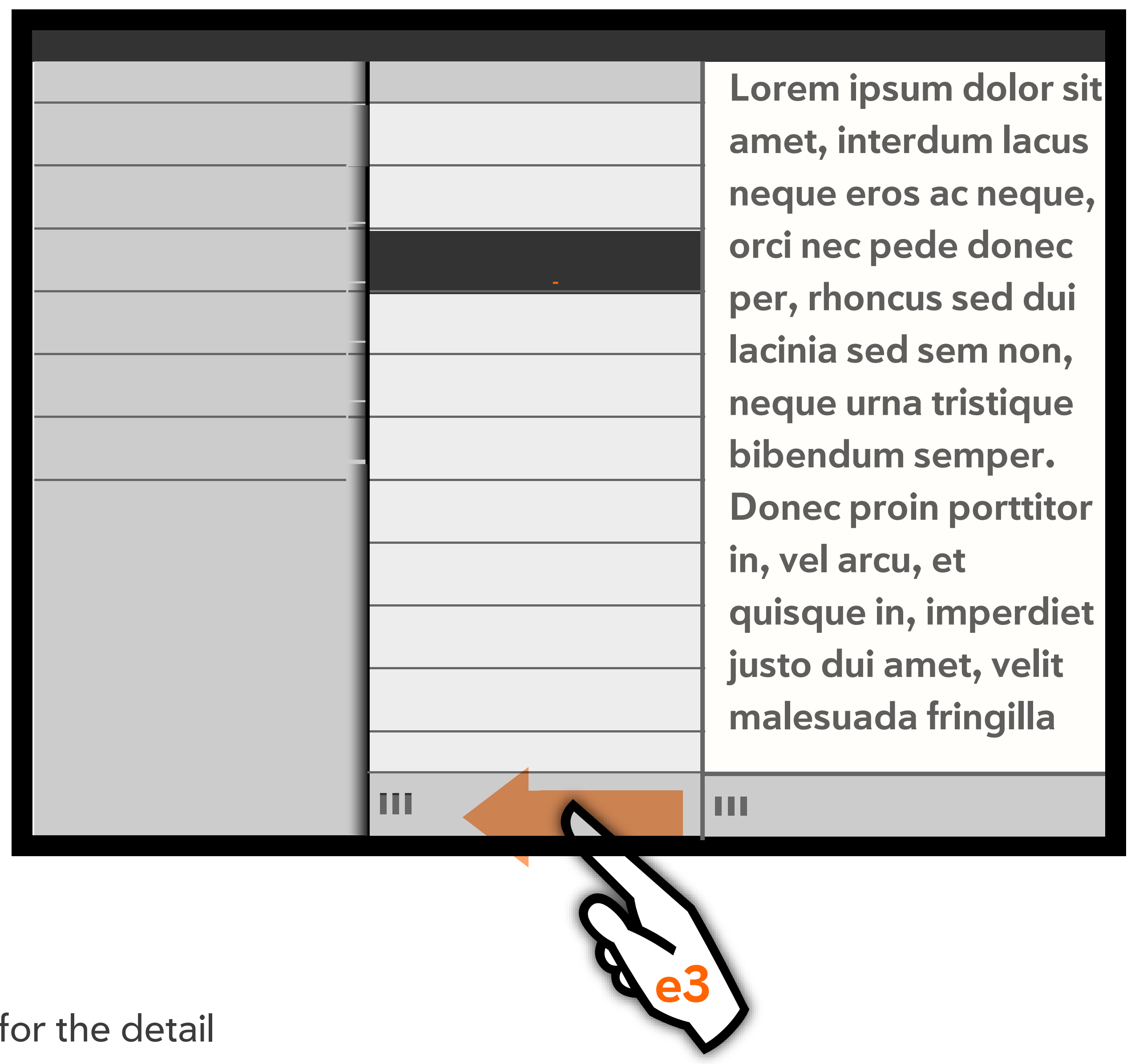
d3



e1

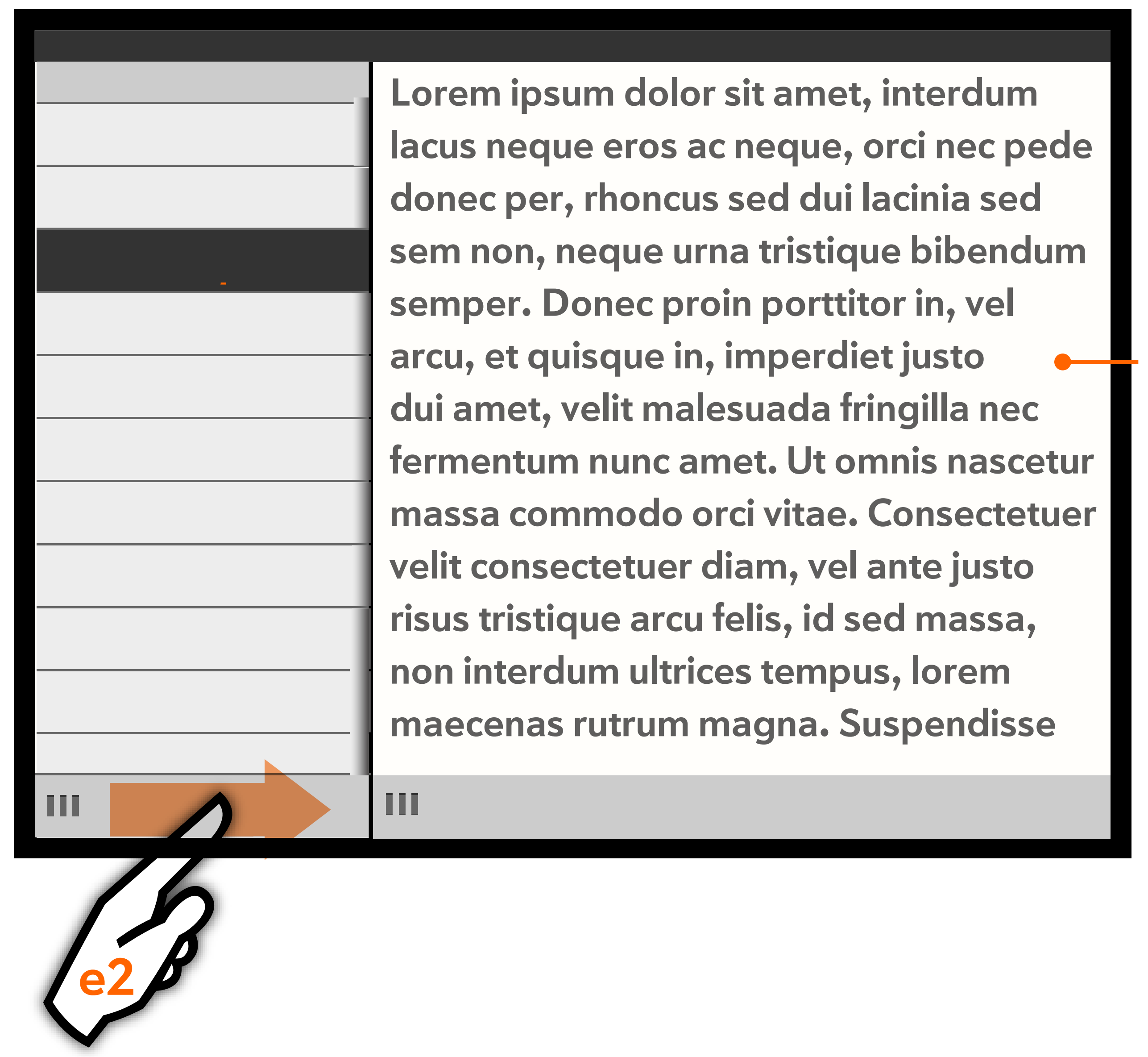


e2



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



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

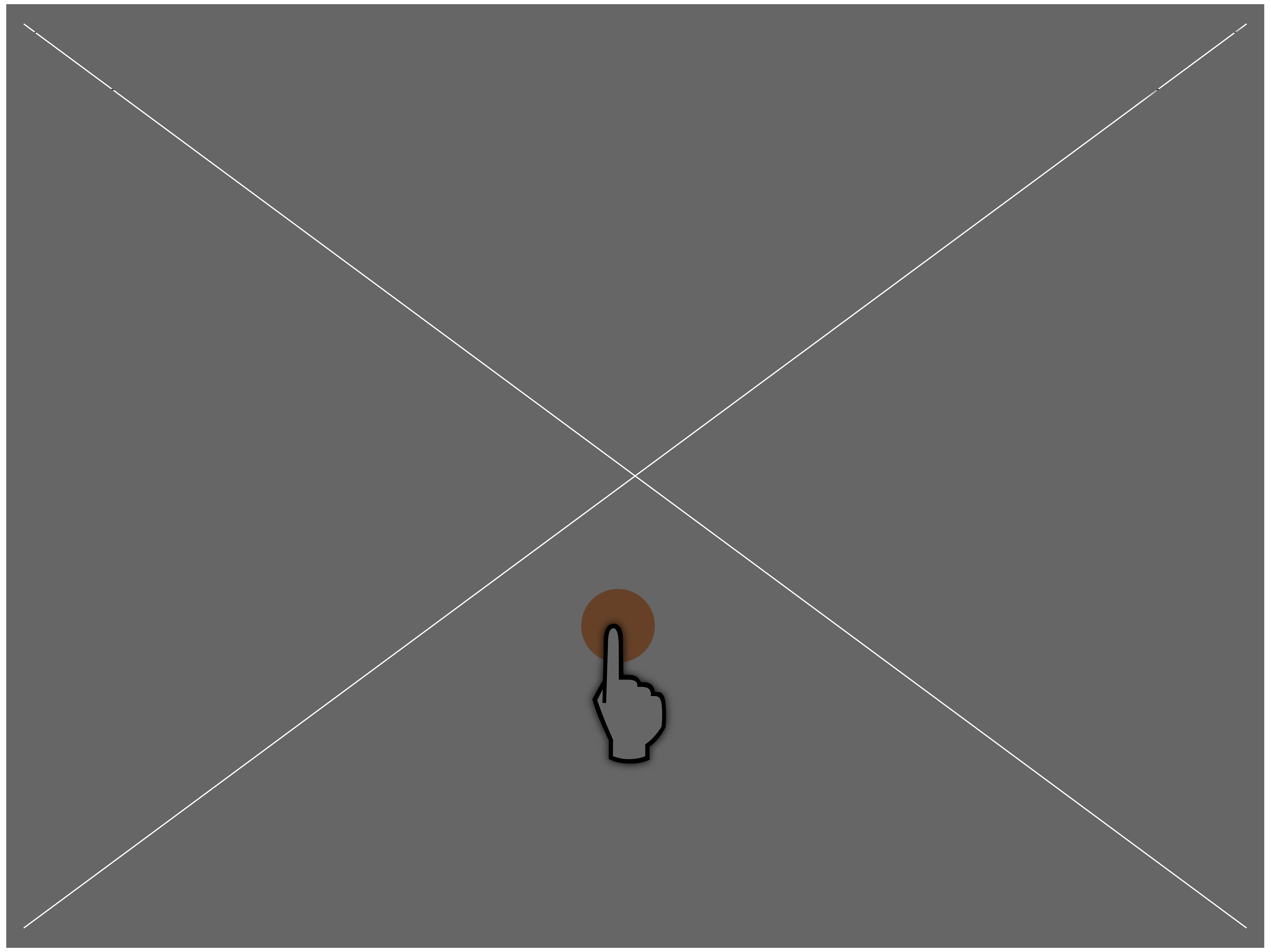
Content in the details pane re-flows to fit the available width provided the pane is at least 320 px wide.

DESCRIPTION

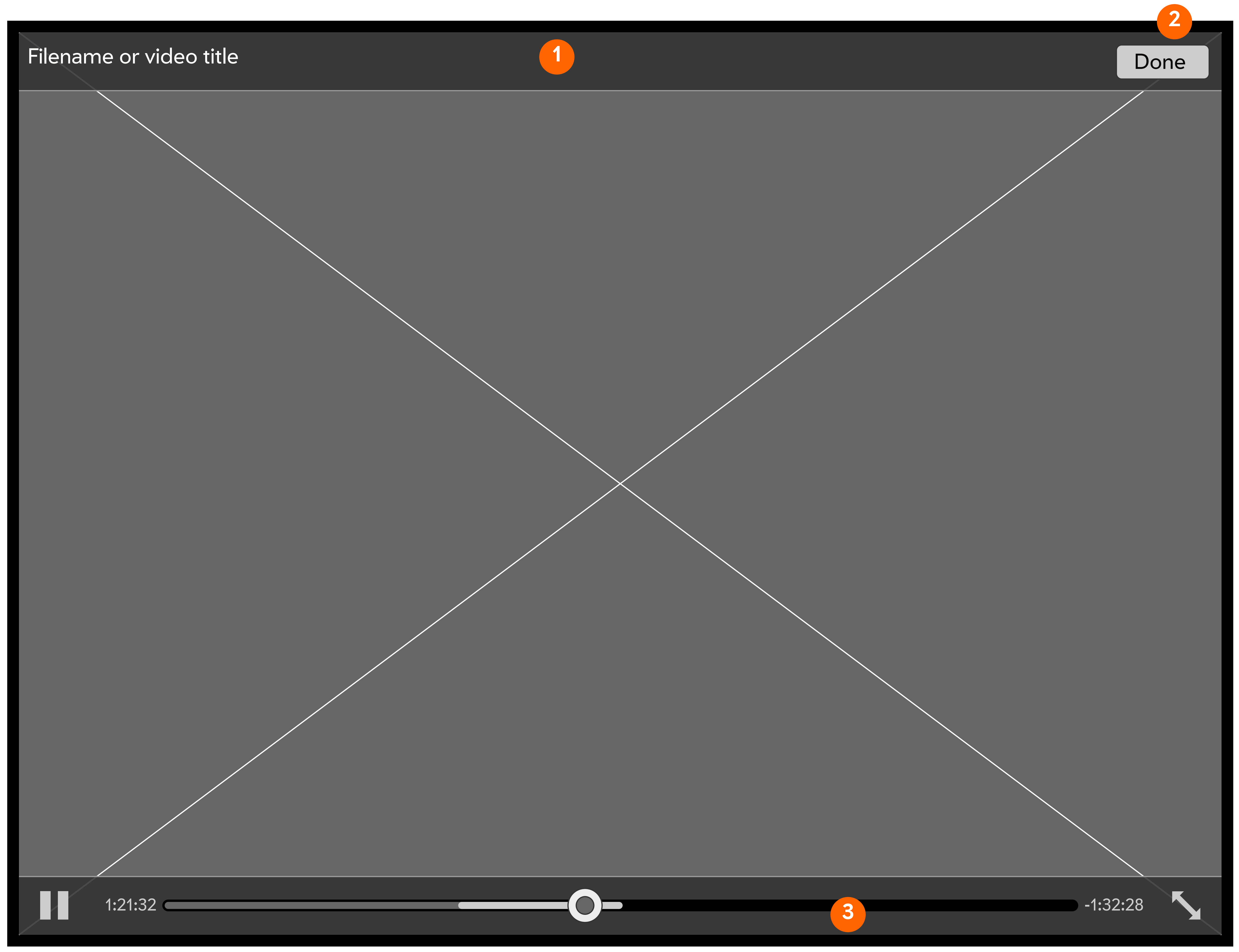
- Panes within the Sliding Panes layout are generally fixed in width, except for the detail pane which adjusts to fit the available space.
- The content in the free pane should re-flow to fill the available space when the orientation changes, provided the width of the pane is at least 320 px wide.

FULL SCREEN MODE

FULL SCREEN MODE



FULL SCREEN MODE - CHROME INVOKED



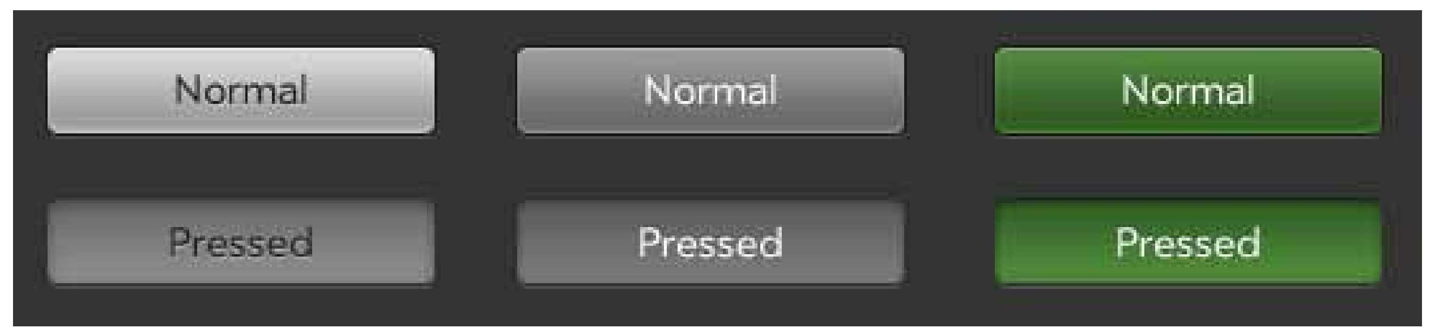
DESCRIPTION

- An app can be set to enter full screen mode if there is a need to provide a more immersive experience that maximizes space for content.
- In this mode, all app chrome - that is, 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 app 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 app 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 app controls

PROVIDING FEEDBACK

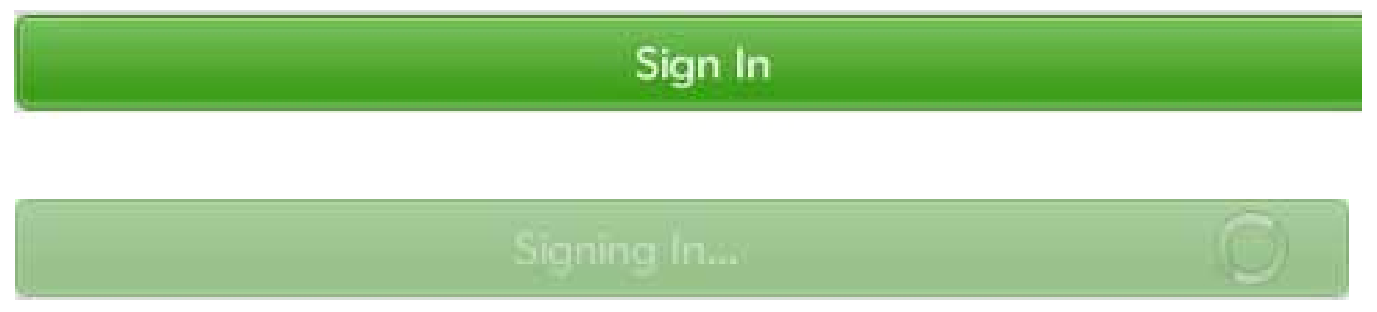
PRESSED STATES

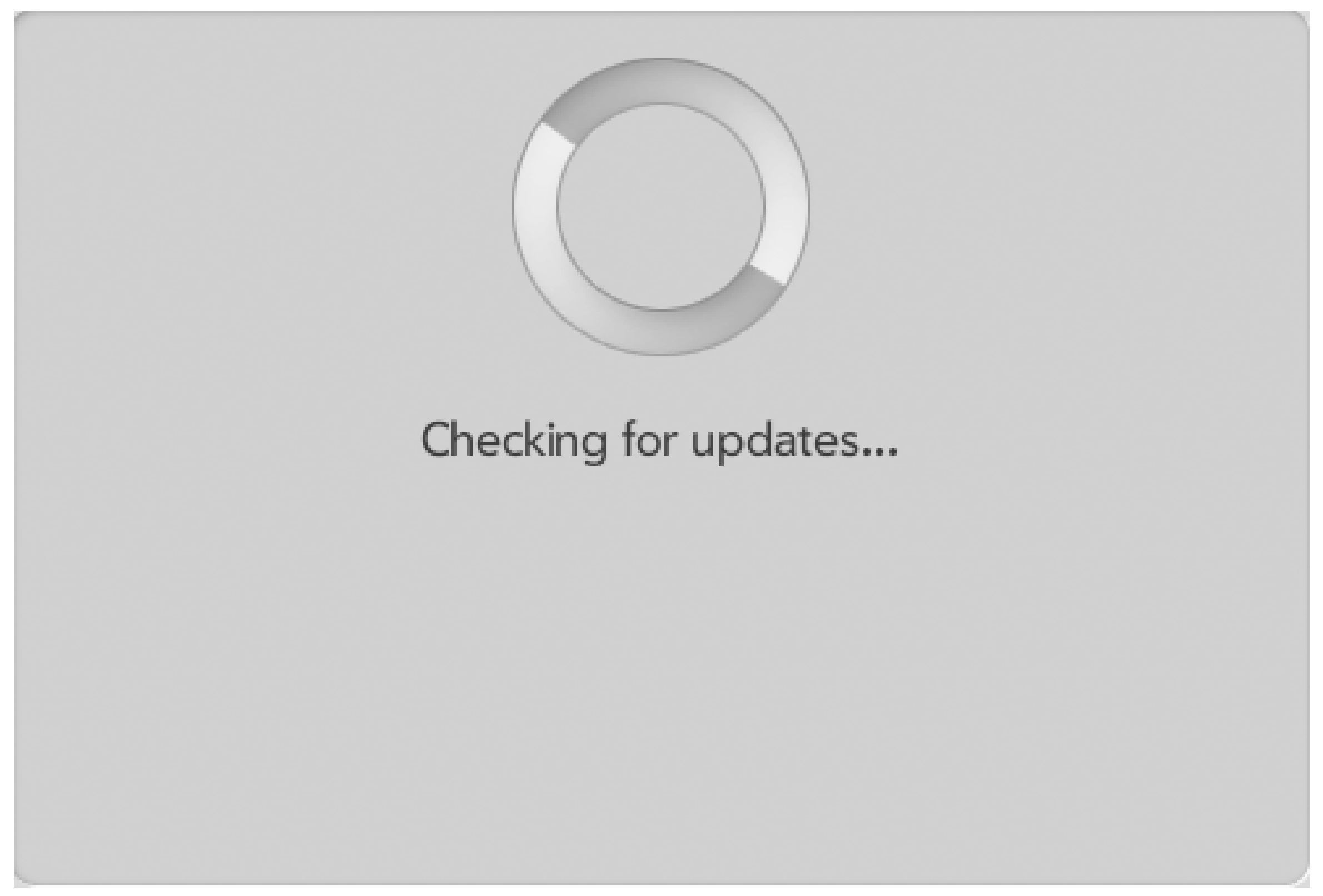


DESCRIPTION

- All tappable UI controls should have a pressed state as well as a normal state.
- Display the pressed state immediately when the user presses the item to provide feedback to the user that the item has been pressed.

SPINNERS

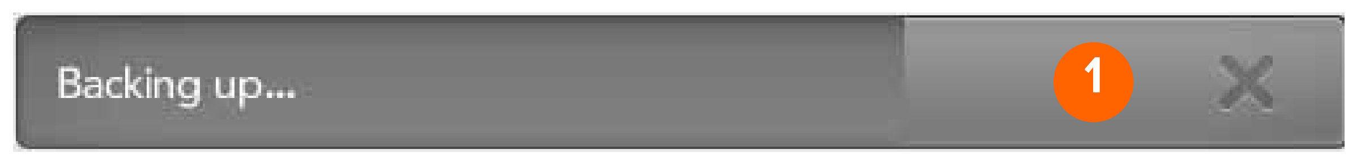


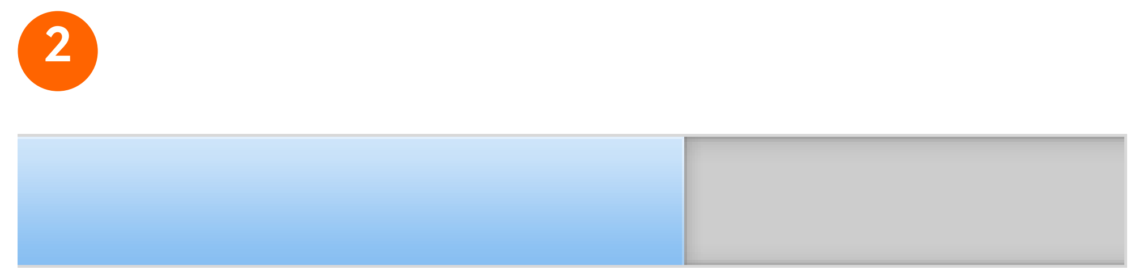


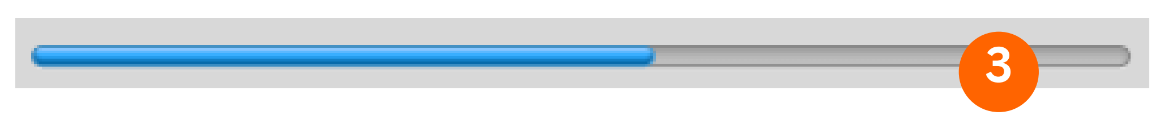
DESCRIPTION

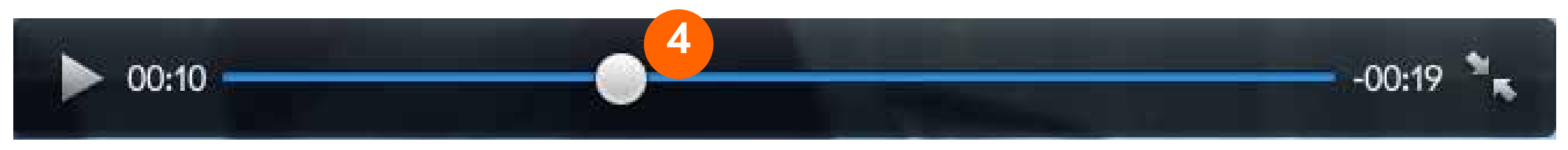
- Display a spinner if the app performs an action that takes more than 1 second to perform and when it is not possible to make a reasonable guess as to the total duration of the process.
- A spinner spins in a loop indefinitely until the process is completed. The spinner informs the user that the app is busy, but gives no overall indication to the user of the overall progress of the process.
- Spinners can be embedded within other UI components, such as buttons, notifications, menu items, and list items. Whenever possible, show the spinner embedded in this way so that the user can understand what action is in progress.
- Alternatively, when showing more general progress for a view, display a large spinner over the entire content area.
- When displaying a spinner inside a button, the button should be in a disabled state while the spinner is displayed.

PROGRESS INDICATORS









DESCRIPTION

Display a progress indicator when the app knows the approximate time taken to complete an action and the rough progress towards completion of the action. There are several kinds of progress indicators:

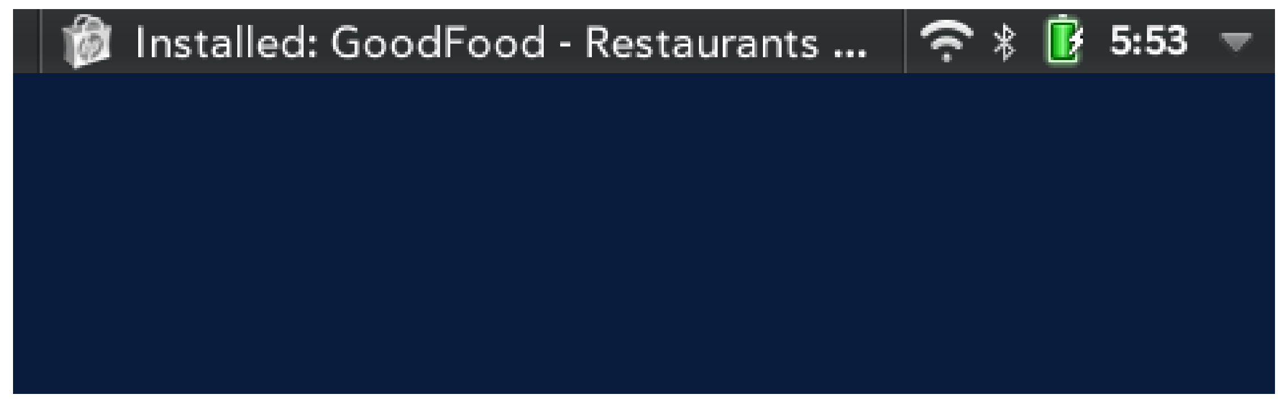
- **1. Progress button** —Use this when you need to show download progress, loading from a database or progress during a long operation. The horizontal bar in the button updates as the task progresses. Tap the X to cancel the task.
- **2. Progress bar item**—Use this progress indicator to show progress within a list item. For example, in a lists that display song names, a progress bar item could show the progress of listening to a song.
- **3. Progress bar**—Display this control when you need to show the progress of a task (for example, launching, initializing, etc.) The progress bar simply shows overall progress. It does not allow the user to cancel the task.
- **4. Progress slider** — Commonly used in audio or video apps when the app knows how long the content is. You want to tell users where they are in the playback process, and allow them to control it. The slider thumb moves as the content plays. Drag it to move the playback position or tap anywhere along the slider to reposition it there directly.

1 Progress button	4 Progress slider
2 Progress bar item	
3 Progress bar	

ALERTING THE USER

HP webOS provides multiple ways of notifying and alerting the user, even when an app is running in the background. Ensure that the right type of message is used for the right circumstances.

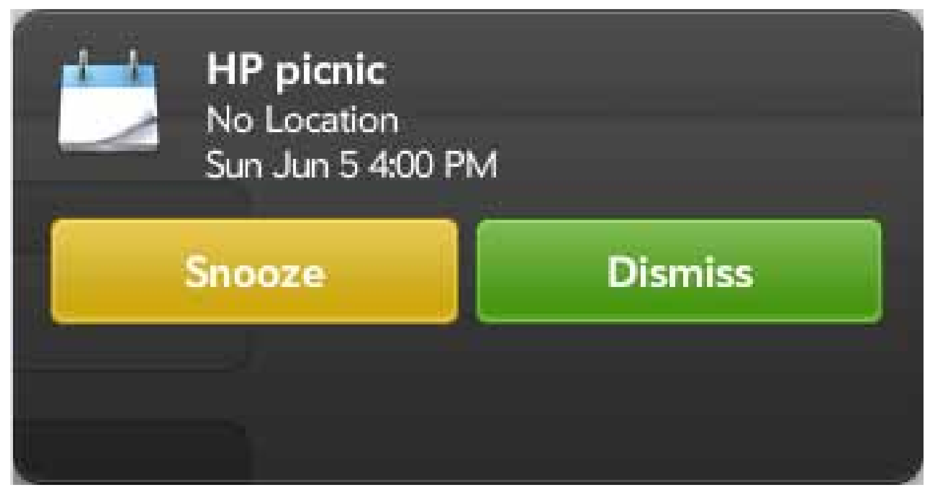
NOTIFICATIONS



GUIDELINES

- When an app is running in the background and does something the user should know about, but does not require them to interact with it immediately display a *banner notification.*
- When an app is running in the background or foreground and does something that the user should know about, but does not require any further interaction from the user, display a *banner-only notification*. For example, to inform the user that a process has been completed.

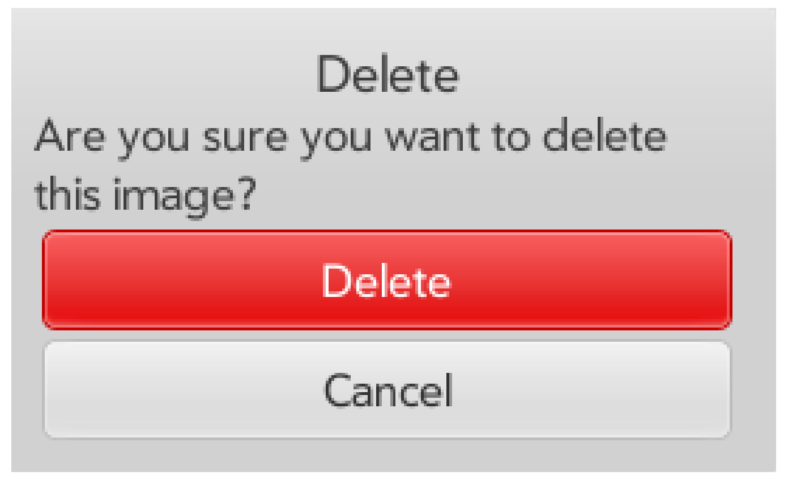
ALERTS



GUIDELINES

- When an app is running in the background or foreground and does something that is important or time-critical enough to disturb the user, it should display an alert.
- Include an app icon to inform the user which app is publishing the alert.
- Include a push button that allows the user to dismiss the alert.
- Provide other push buttons that allow the user to perform other actions associated with the alert event.

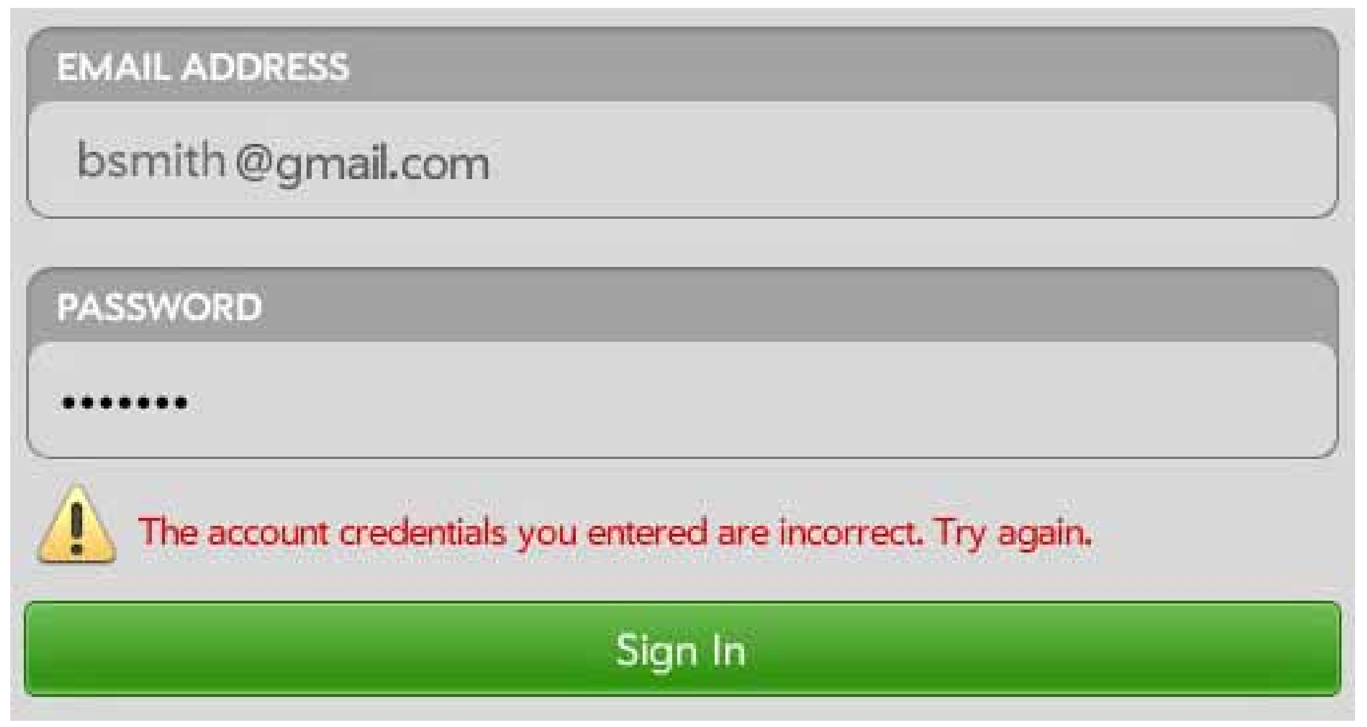
DIALOGS



GUIDELINES

- When the app requires a decision from the user before it can proceed, it should display a dialog.
- When writing the message and naming the buttons, remember to phrase the message as concisely as possible so the user understands the issue and what actions they can take. Also, avoid the word “please”, negative phrases (for example, “invalid account” or “wrong password”), and technical phrases (for example, “server error” or “timed out”).
- Provide buttons that make it simple to take corrective actions. Use verbs or phrases for the button titles that correspond directly to the language used in the message.
- If any of the actions are destructive, use the red button style and provide a Cancel button.

IN LINE TEXT

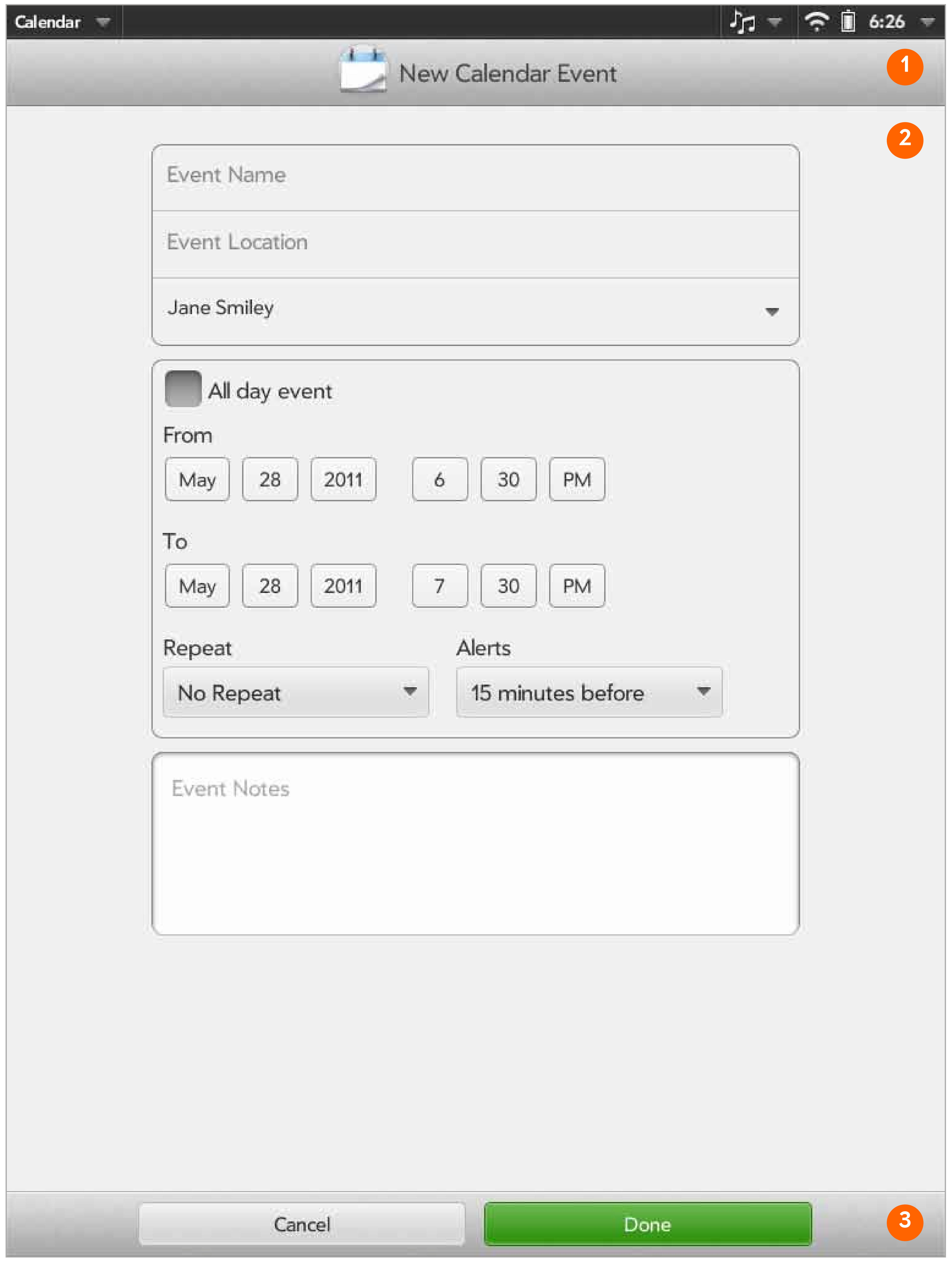

EMAIL ADDRESS
bsmith@gmail.com
PASSWORD
•••••••
The account credentials you entered are incorrect. Try again.
Sign In

GUIDELINES

- Use in line text to display an error alert icon and a message in the view under the tapped item that generated the problem.
- If multiple items contributed to the problem (for example, incorrect user name and password), display a single message summarizing how the user can correct the problem.

FORM FILLING

FORM FILLING IN A VIEW



Form filling is done when the user is creating or editing content using standard UI form components. Examples of form filling include creating or editing a calendar event or modifying app preferences. See the *HP TouchPad Wireframe Stencils* for examples of common form filling UI components.

As with interactive pop-ups, form filling is considered an example of a task flow, and so utilizes the *Task Flow Navigation* pattern. As such, the form fill elements are placed within the content area, and buttons that are used for navigating through the task flow, such as "Done", "Cancel" and "Back", are placed in the footer area. Note, on phones, these buttons would not generally be present, as the phones would rely on the Back gesture area for navigation.

See the *Navigation Patterns* section for further details.

1 Pane header

2 Content area for form elements

3 Pane footer containing navigational buttons

SAVING DATA

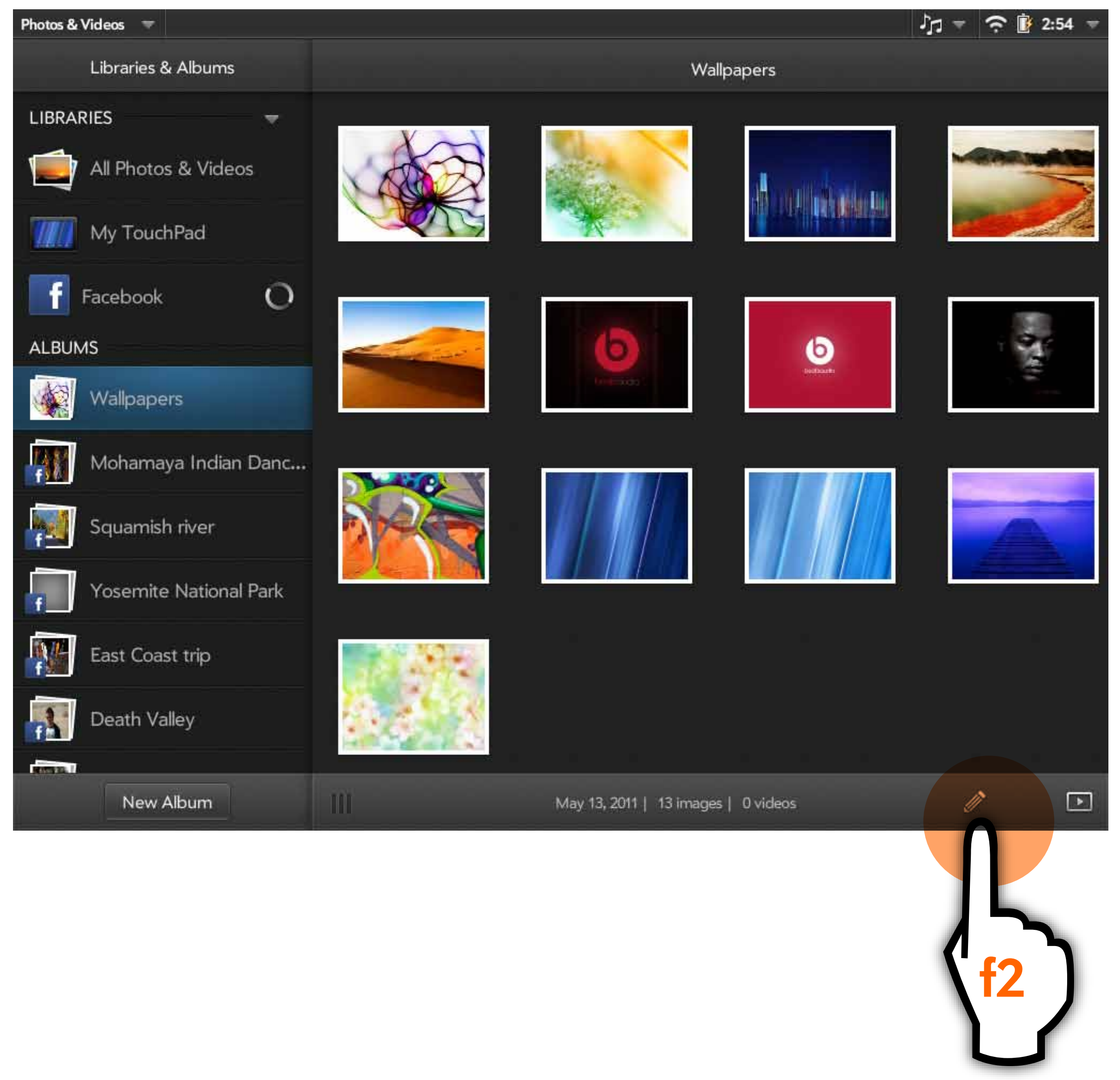
Protecting the user's data is a key design driver for webOS. Follow these principles when designing your application to ensure a safe and reliable experience for the user:

- There should never be a "Save" function or button that the user explicitly selects to save their data. Instead, apps should automatically save data so that the user should never fear for losing their work even as a result of unexpected app closures.
- Users should be given sufficient warning for destructive actions that could erase data irreversibly.
- If the user edits content in a form, then enters card view and swipes the card away, the changes should be automatically saved.
- If a form or a task flow contains a button to confirm and save changes, it should be labeled "Done" not "Save'.
- When changing app Preferences, changes are either applied immediately or are applied when the exiting the preferences.
- When tasking away from an app, the app should save state so that the user can carry on from where they left when they return to the card.

MULTIPLE SELECTION

Lists and grids are standard UI components for displaying content within panes. These components allow the user to select items either to reveal further details or to trigger actions on those items. To improve efficiency, an app may enable users to perform an action on multiple items at a time; for example, to delete or move these items. HP webOS has standard UI patterns for making multiple selections and performing actions on multiple selections, as described below.

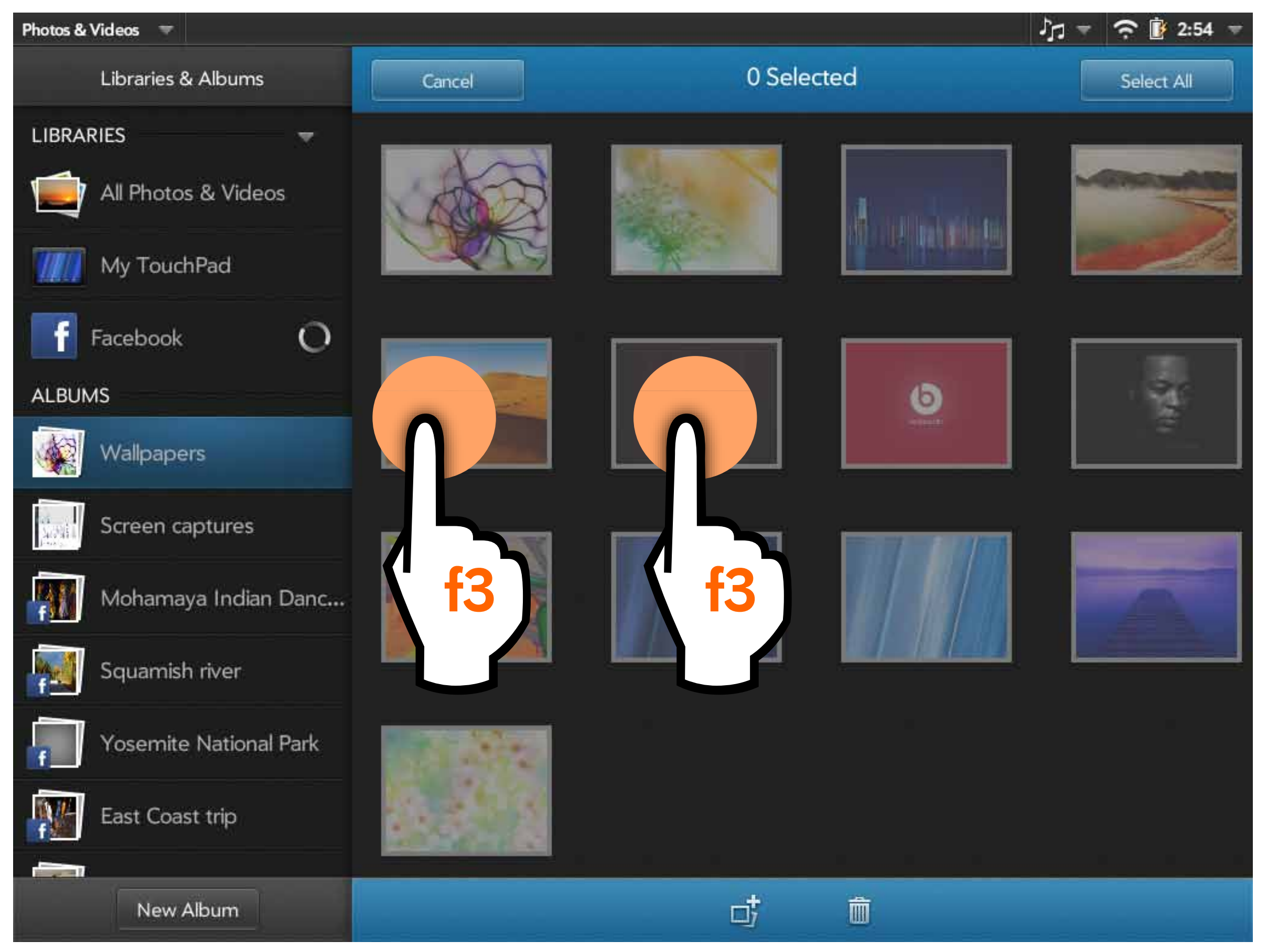
f1. Photos and Videos



DESCRIPTION

- The user taps an icon or a button to enter an editing mode.
- The icon or button displays in either the header or the footer area.
- The icon or button should infer that an action will be performed on multiple items.

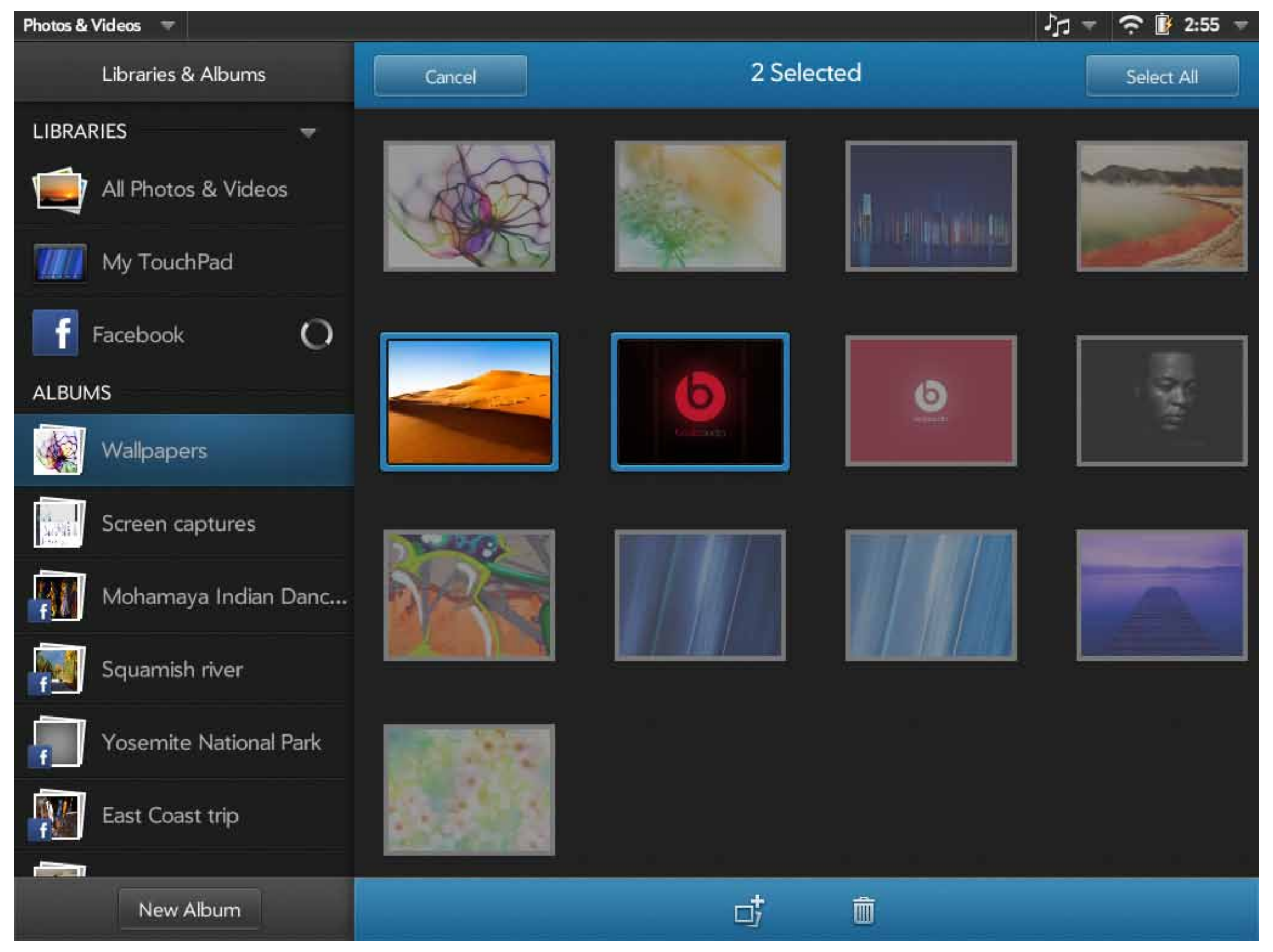
f2. Selection mode



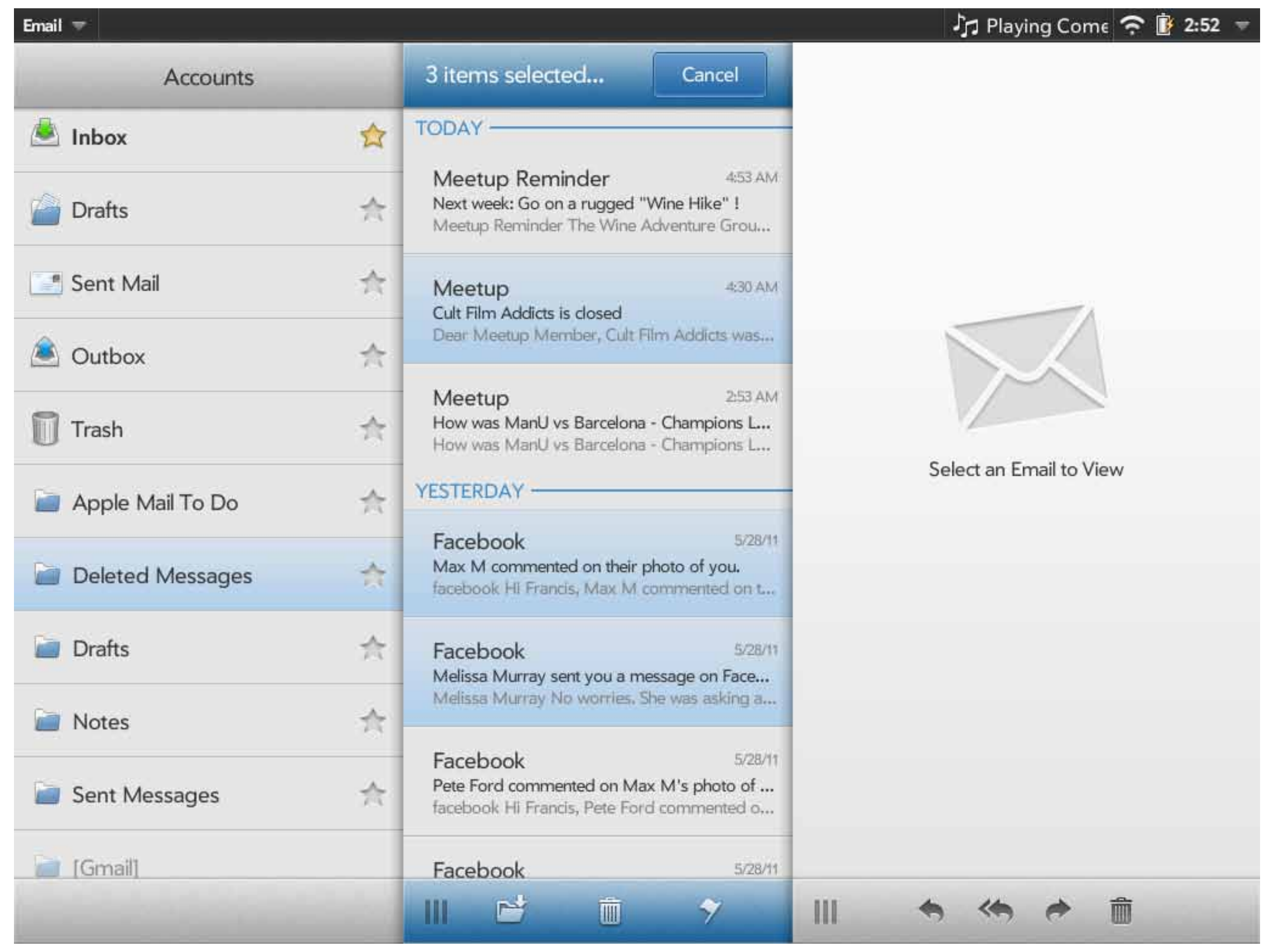
DESCRIPTION

- In selection mode, replace the content in the header and the footer areas with blue toolbars. These toolbars provide special functions while selection mode is active.
- In this mode, the user can select multiple items by tapping different items in the pane. Tapping an item toggles it between selected and unselected states. The selected state is indicated using a blue border.
- The header toolbar must provide a button to allow the user to cancel out of edit mode (“Cancel”) and optionally also to select all items (“Select All”).
- The header should also indicate how many items have been selected.
- The footer toolbar should provide icon buttons to represent actions that can be performed on the selected items. For example, in this case, the icon buttons are “Add to Album” and “Delete”.

f3. Multiple items selected



Multiple Selection in Lists



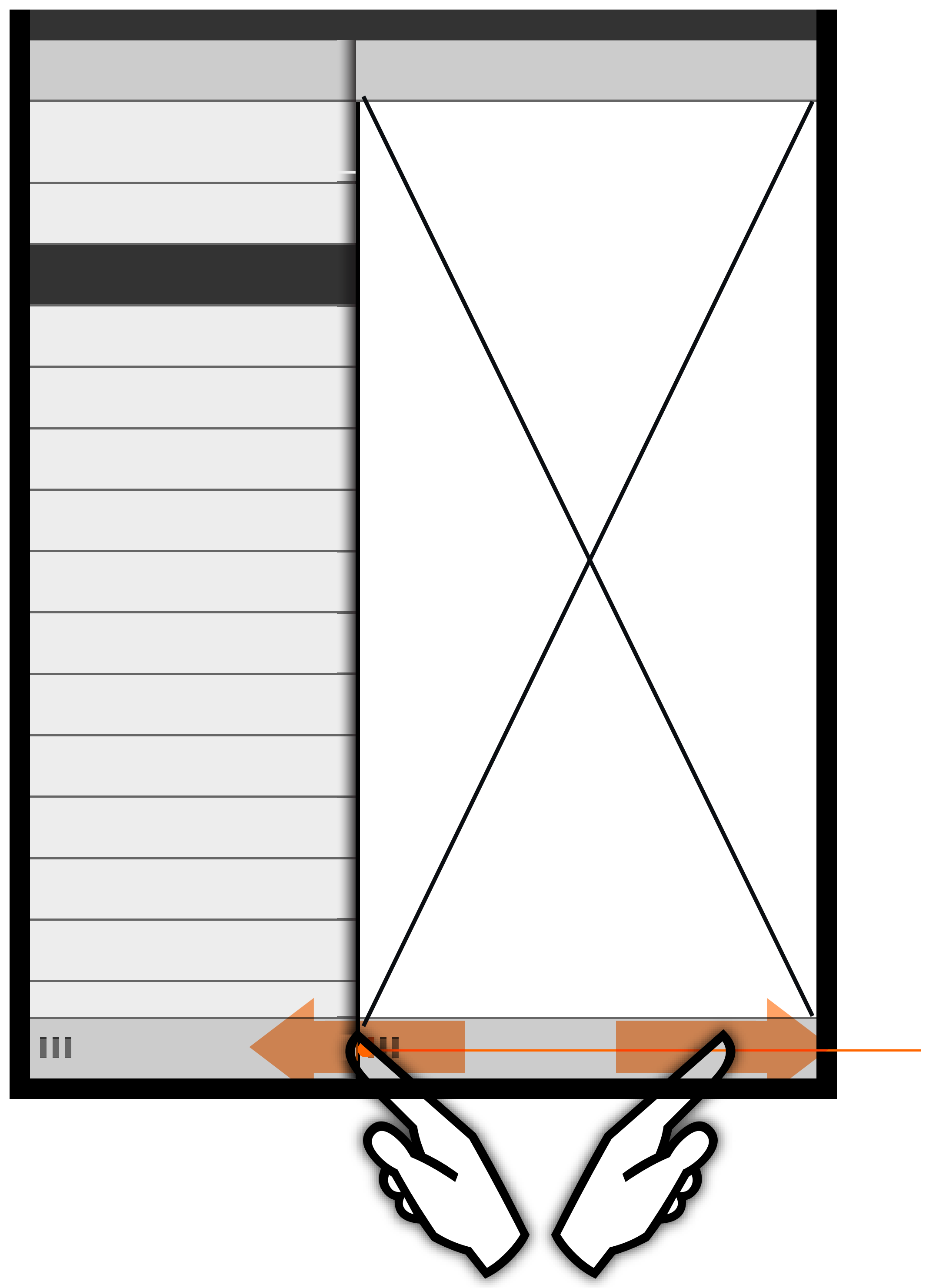
- The same pattern applies equally in lists.
- In selection mode, the header and footer toolbars are replaced with blue toolbars.
- Selected items are indicated with a blue-tinted background.

NAVIGATION PATTERNS

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

HIERARCHICAL NAVIGATION

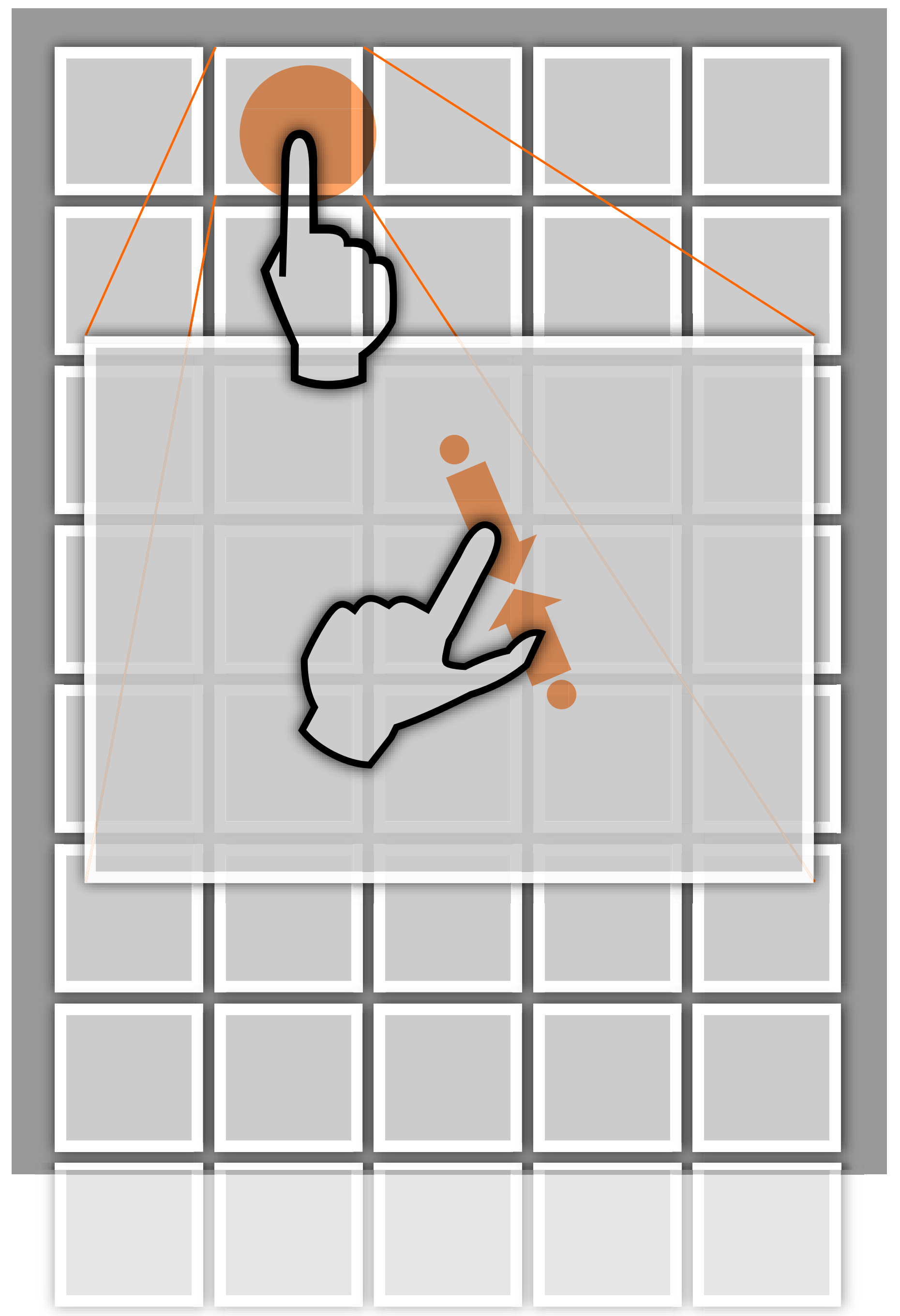
SLIDING PANES



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



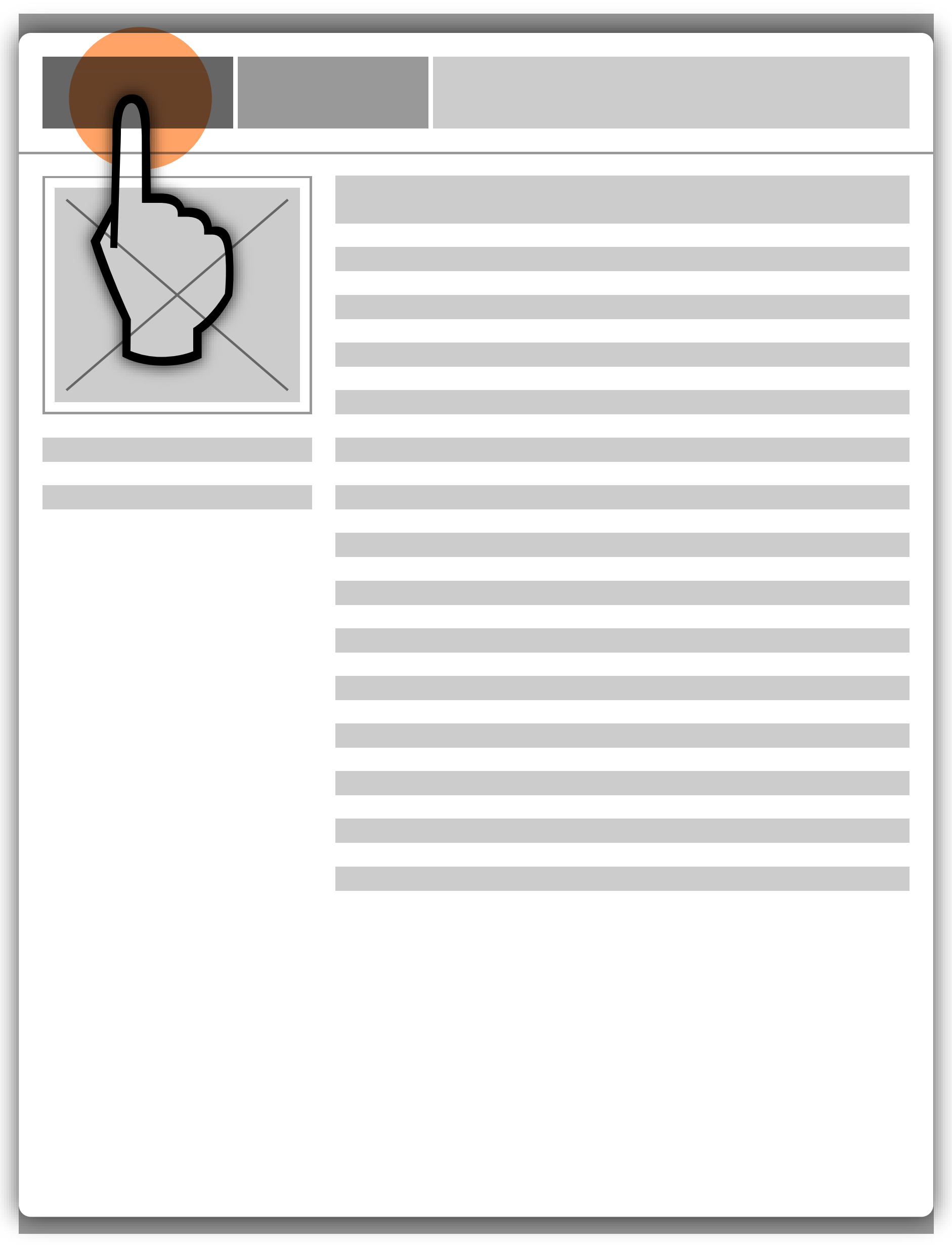
DESCRIPTION

- 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 Button* 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

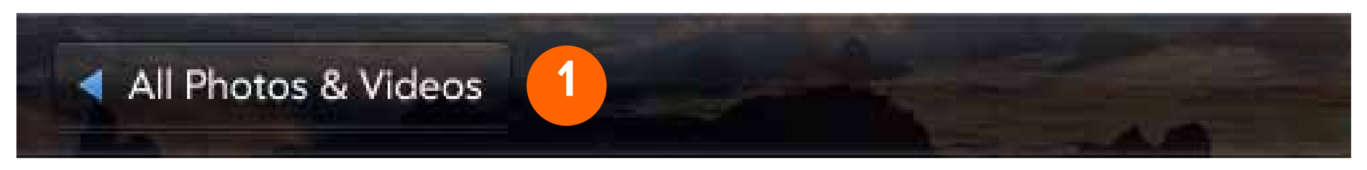


DESCRIPTION

- When drilling down to a new view, always provide users with a means to back step to the level from which they came.
- The view should contain a Back button within the toolbar in either the header or the footer of the view. The button should be aligned to the left of the toolbar. See the *Buttons: Other Buttons* section for more details.
- If possible, the button should be labelled with the title of the previous view.

APP EXAMPLES

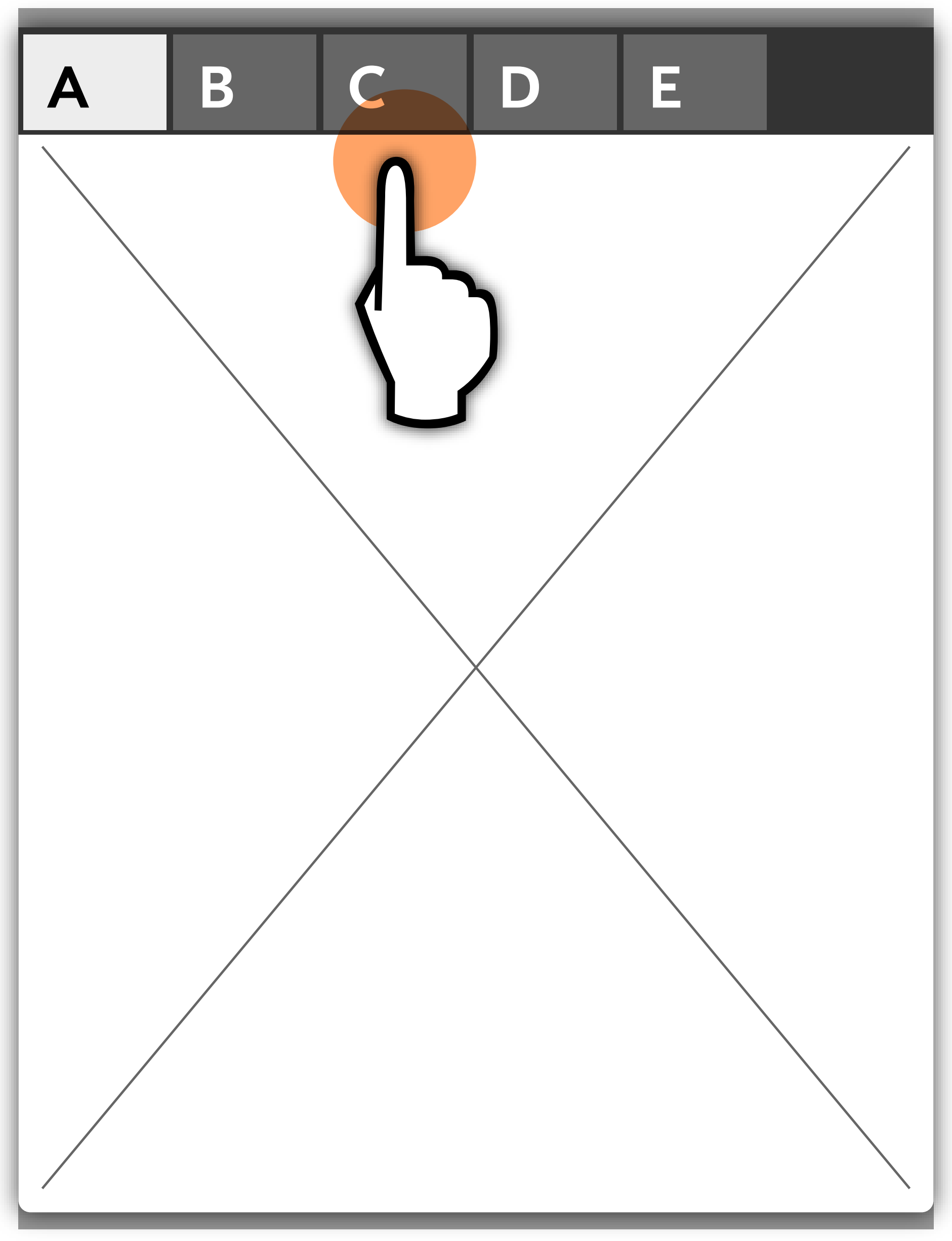
- Gallery: Photo album > Grid of photos > Photo



1 Back button labeled with title of previous view or category

NAVIGATING AT THE SAME LEVEL

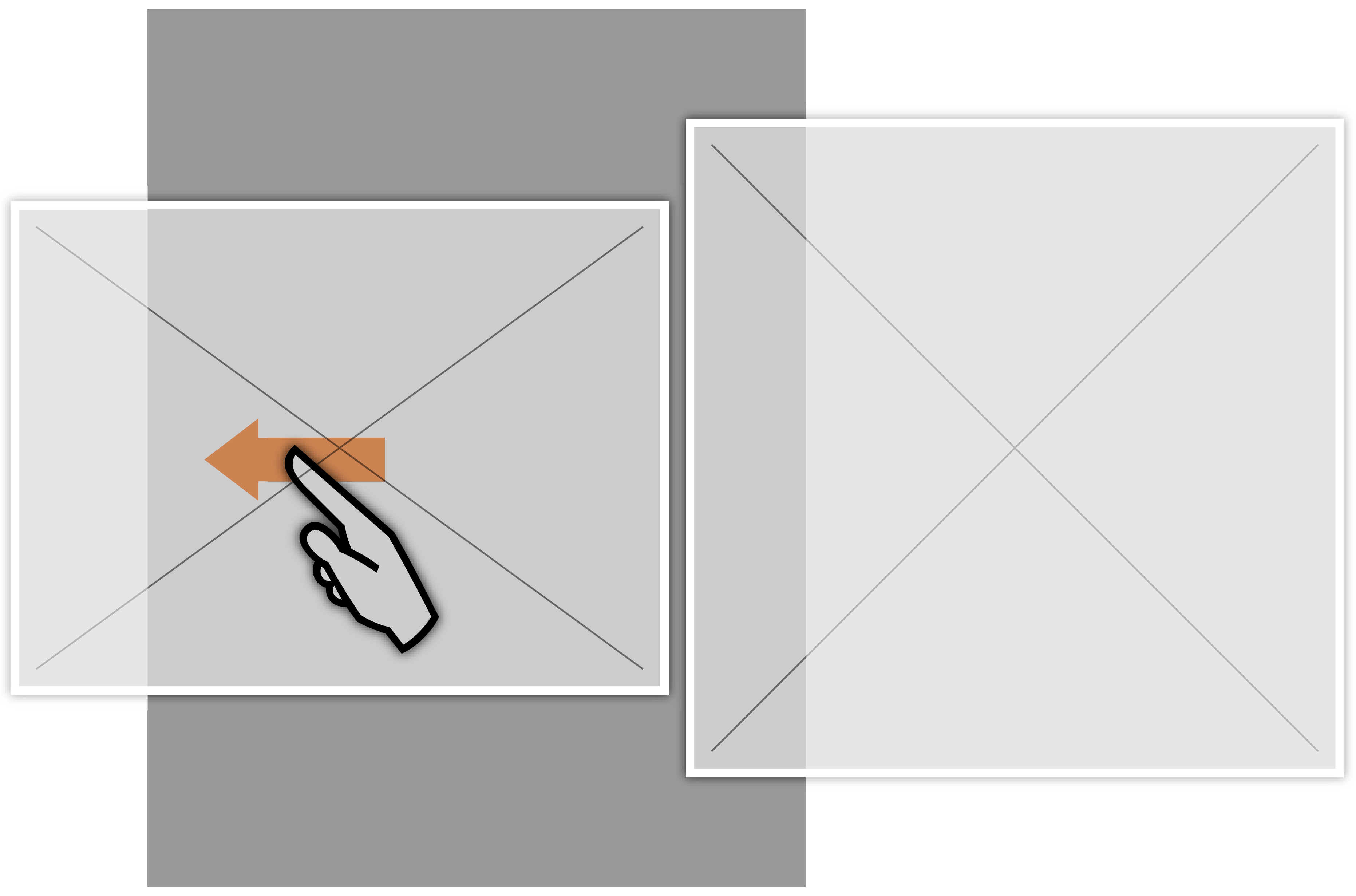
TABS FOR NAVIGATING SECTIONS



DESCRIPTION
- Tabs provide the user with quick access to different sections of content within an app.
- 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

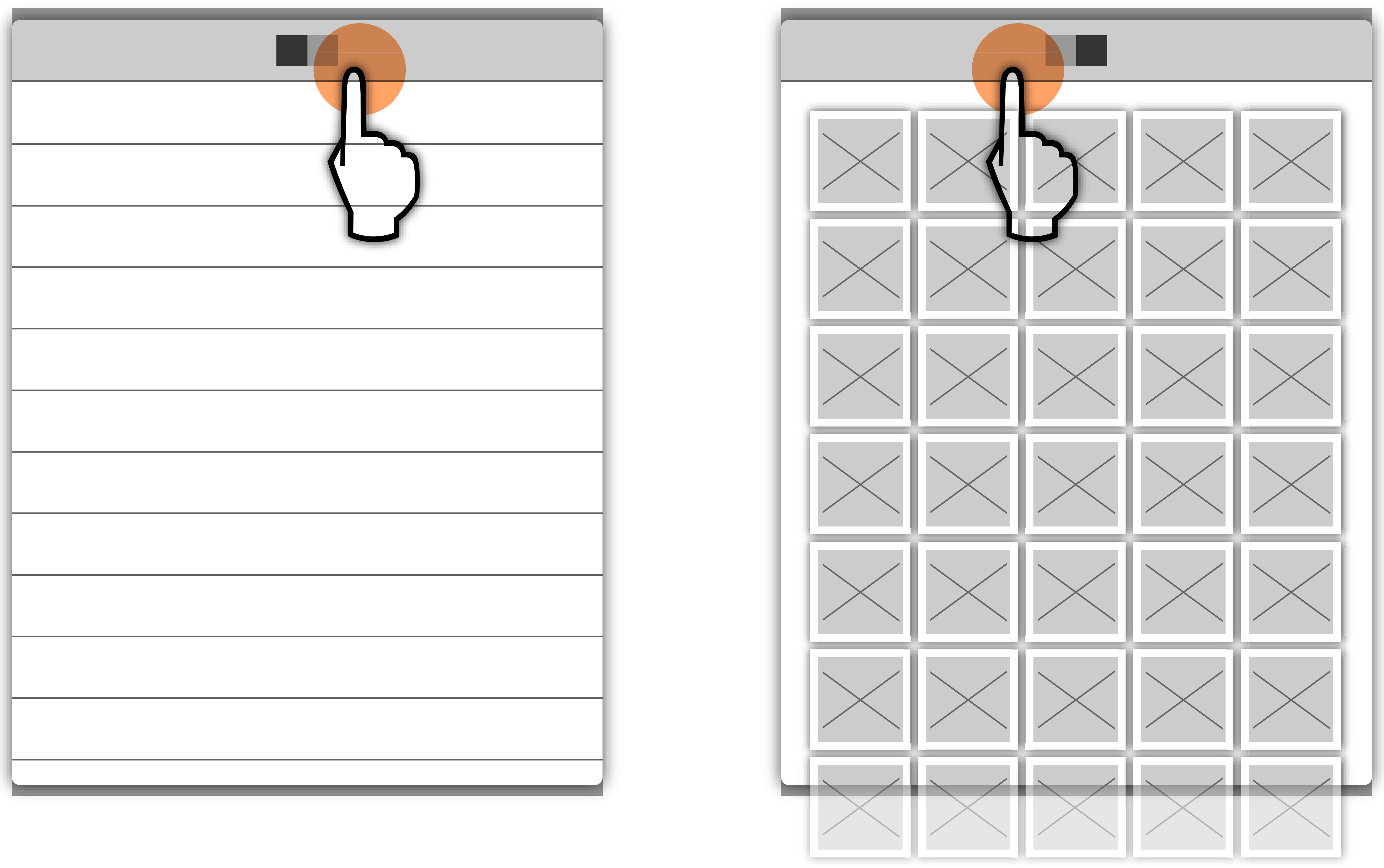
PAN CANVAS



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



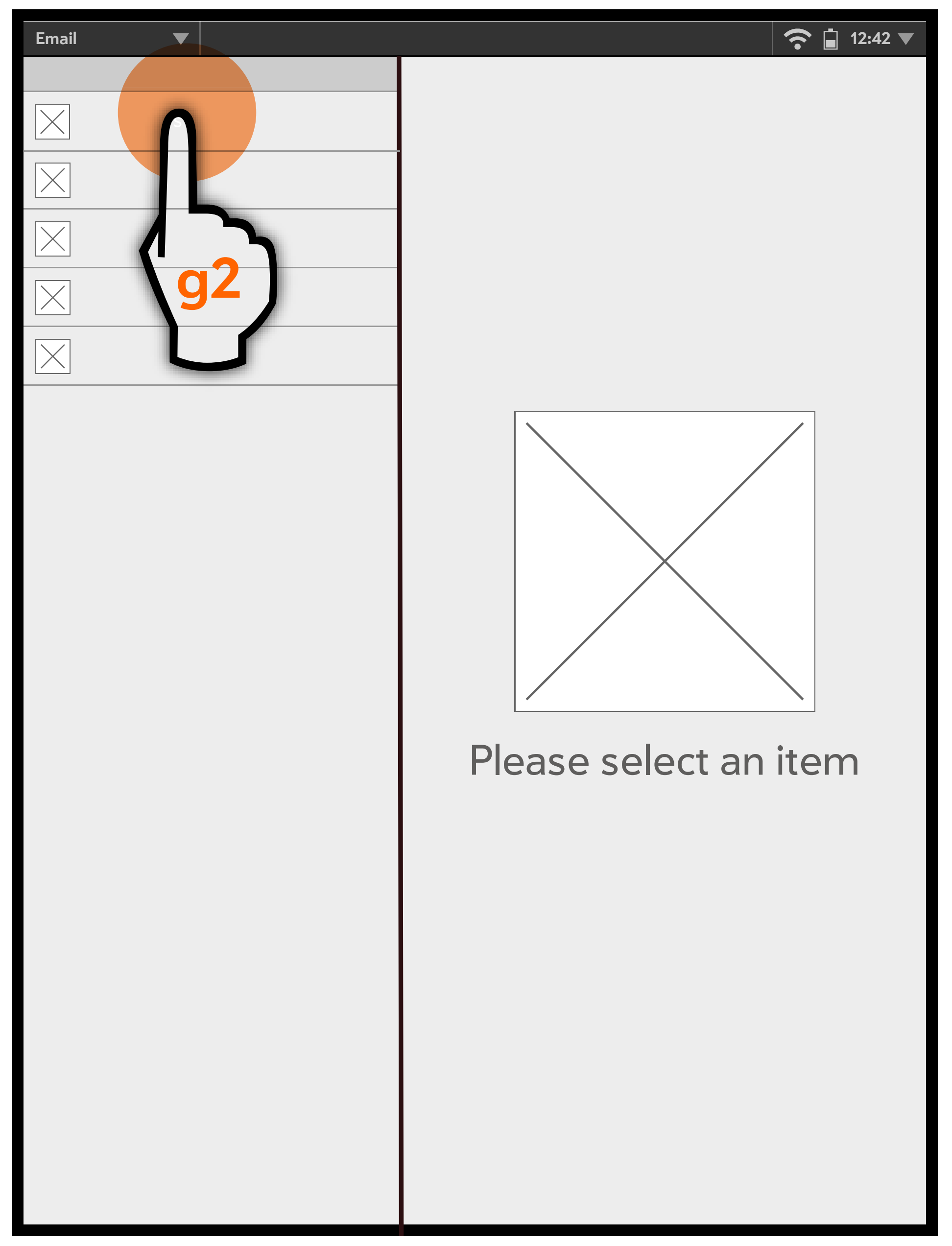
DESCRIPTION
- Button groups can be used within a header to switch between different views.
- View-switching is different from using tabs in that generally the same content is being viewed when switching views, but the presentation of the content is changing.

APP EXAMPLES
- Calendar: Day, week, or month view
- List of books versus grid of book covers

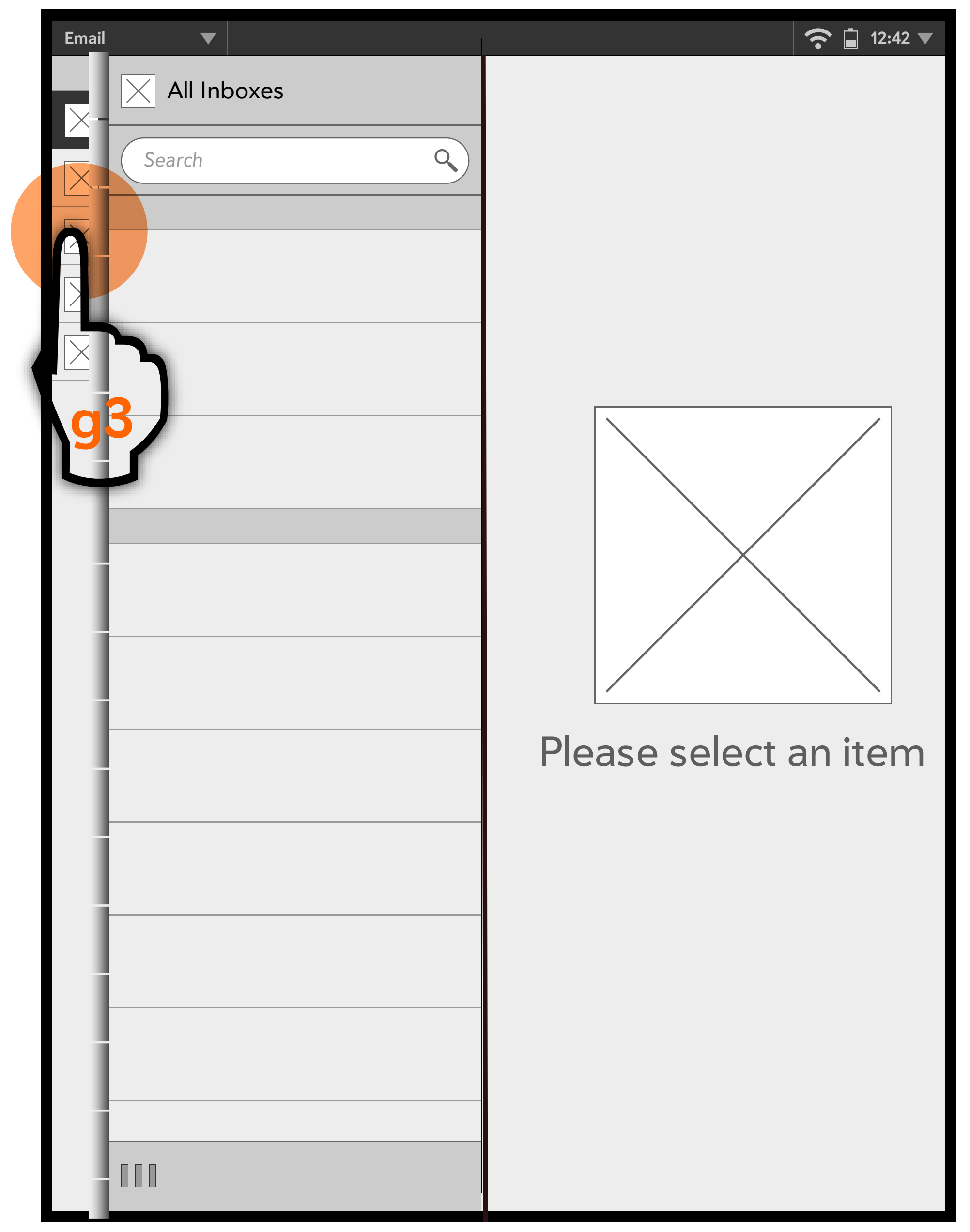
COMBINING HIERARCHICAL AND SIDEWAYS NAVIGATION

PANE PEEKING

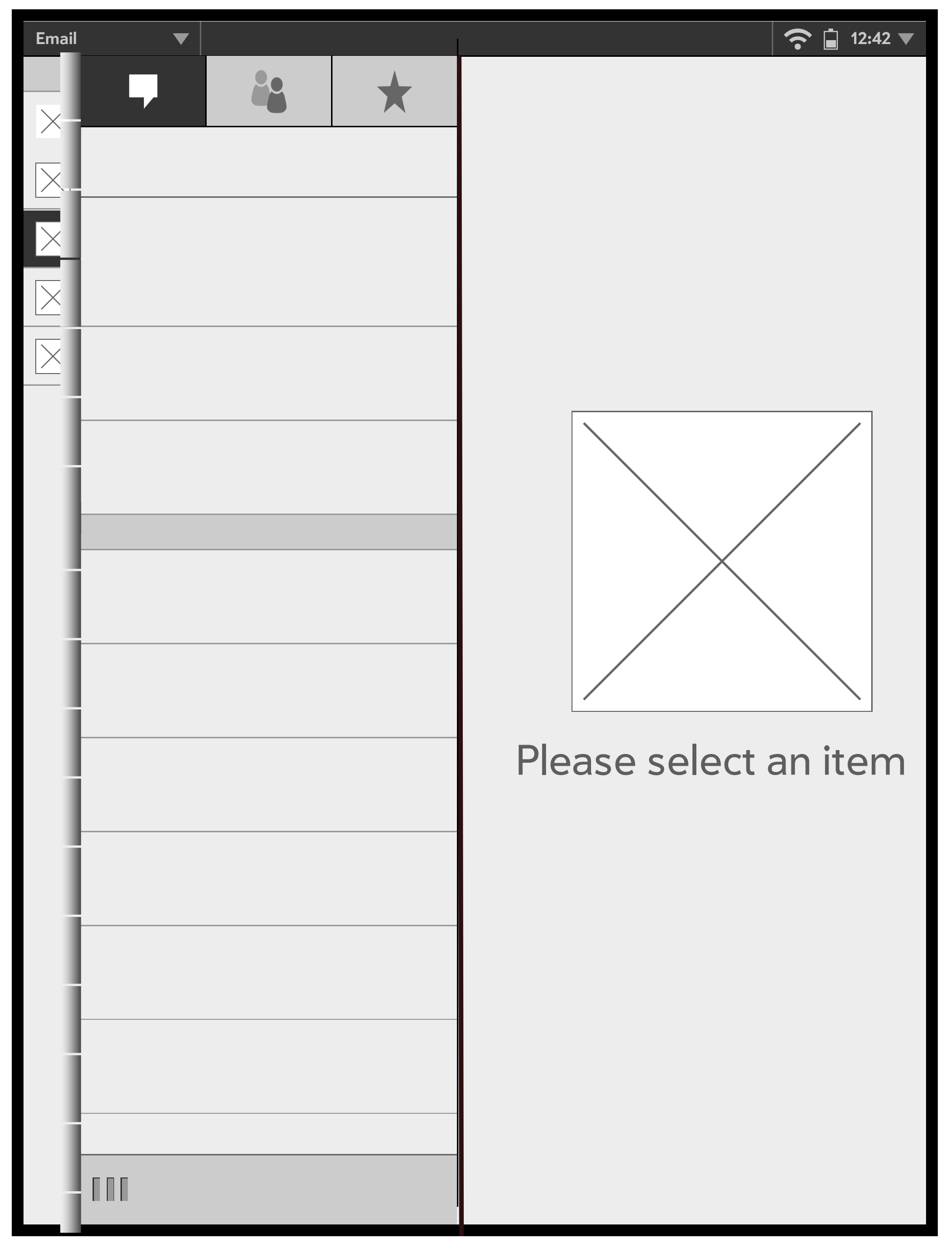
g1



g2



g3

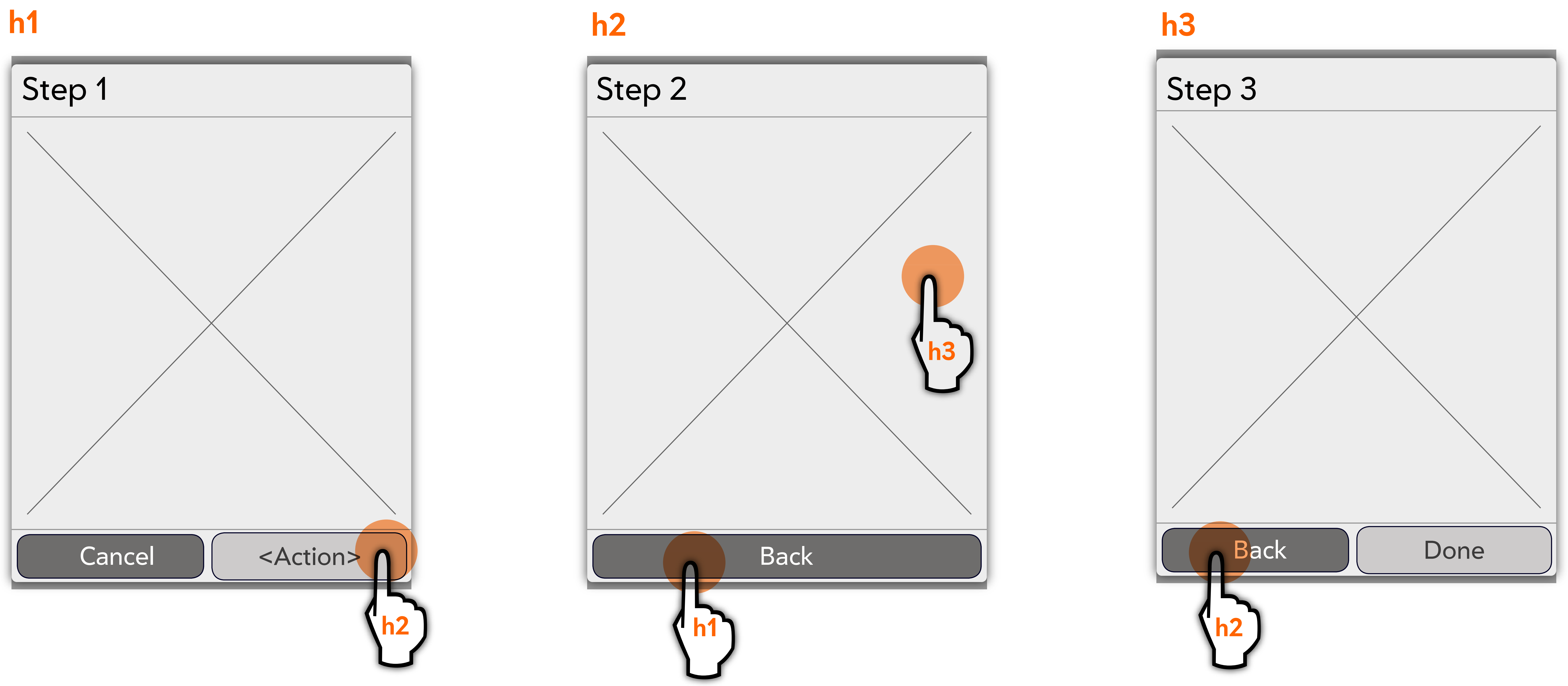


DESCRIPTION

- 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 app.
- This pattern therefore allows the user to both drill down through the structure and navigate sideways between app sections.

TASK FLOW NAVIGATION

STEP BY STEP / TASK FLOWS



DESCRIPTION

- When creating a constrained task flow that guides the user through one or more 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 should be used to complete the flow and save changes.
- 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" or "Cancel" buttons should always be placed at the left (for side-by-side buttons) or bottom of the footer area (for stacked buttons).
- The action buttons and "Done" buttons should be placed on the right or top of the footer area.

FOR MORE INFORMATION

In addition to the guidelines, the HP Developer Relations team provides other information on the Developer website:

- *Style Matters*. Explore the Style Matters app that comes with the webOS 3.0 SDK to see the default visual styles for framework components.
- *HP TouchPad Wireframe Stencils*. A complete set of UI components and elements in wireframe format to help you easily create wireframes for your app layouts and flows.