UNITED STATES DISTRICT COURT

NORTHERN DISTRICT OF CALIFORNIA

SAN JOSE DIVISION

APPLE INC., a California corporation,

Plaintiff,

v.

SAMSUNG ELECTRONICS CO., LTD., A Korean business entity; SAMSUNG ELECTRONICS AMERICA, INC., a New York corporation; SAMSUNG TELECOMMUNICATIONS AMERICA, LLC, a Delaware limited liability company,

Defendants.

Case No. 11-cv-01846-LHK

EXPERT REPORT OF KARAN SINGH, PH.D. REGARDING INFRINGEMENT OF U.S. PATENTS NOS. 7,864,163, 7,844,915 AND 7,853,891

****CONFIDENTIAL – CONTAINS MATERIAL DESIGNATED AS HIGHLY CONFIDENTIAL – ATTORNEYS' EYES ONLY PURSUANT TO A PROTECTIVE ORDER****

features of the '163 patent, although this alternative is, in my opinion, less appealing to users. The Browser on a smartphone or tablet computer could be programmed to use gestures to zoom in and out on portions of a structured web page without the additional ability, once zoomed in, to use a "second gesture" (in the language of the '163 patent) to translate to a different box of content. This appears, from Samsung's own Relative Evaluation Report (SAMNDCA00203880 at SAMNDCA00203937), to be precisely how a Galaxy S prototype functioned before it imitated '163 functionality from an Apple iPhone: the prototype allowed zooming in an zooming out, but translation to a second box of content via a second gesture in the zoomed in state was not possible. Samsung itself assessed this alternative functionality as inferior—it proposed an "[i]mprovement" to "supplement the double tapping enlargement/shrinkage feature" to include all of the '163 patent's features. (*Id.*) I agree that the '163 functionality is superior.

VI. DETAILED OPINION REGARDING THE '915 PATENT

A. Summary of the '915 Patent

282. The '915 patent is entitled "Application Programming Interfaces for Scrolling Operations." The application that resulted in the '915 Patent was filed on January 7, 2007.

283. The '915 patent is generally directed to methods and apparatus for responding to user inputs on a touch-sensitive display integrated with a device. The asserted claims of the '915 patent recite methods and apparatus that distinguish between a single-input point that is interpreted as a "scroll operation" and two or more input points that are interpreted as a "gesture operation."

284. The Background of the Disclosure section of the specification explains that various devices such as electronic devices, computing systems, portable devices, and handheld devices have software applications and application programming interfaces or "APIs" that interface between the software applications and user interface software to provide a user of the device with certain features and operations. ['915 patent, col. 1:7-8, 33-37.]

285. The specification further explains that various types of electronic devices, such as portable devices and handheld devices, have a limited display size, user interface, software, API interface and/or processing capability which limit the ease of use of the devices. User interfaces

of devices implement APIs in order to provide requested functionality and features, such as scrolling, selecting, gesturing, and animating operations for a display of the device. The '915 patent explains that one issue with these user interfaces is that they can have difficulty interpreting the various types of user inputs and providing the intended functionality associated with the user inputs. ['915 patent, col. 1:48-55.]

286. The '915 patent proposes a method for responding to a user input of a device, such as a portable electronic device (e.g., cellular phone, media player, multi-touch tablet device), in order to implement and distinguish between various desired input operations for a user interface, such as a scrolling operation and a multi-finger gesture operation. ['915 patent, col. 6:20-60.]

287. Figure 1 of the '915 patent illustrates one embodiment of a method for responding to a user input of a data processing device that is covered by claims 1, 8 and 15.

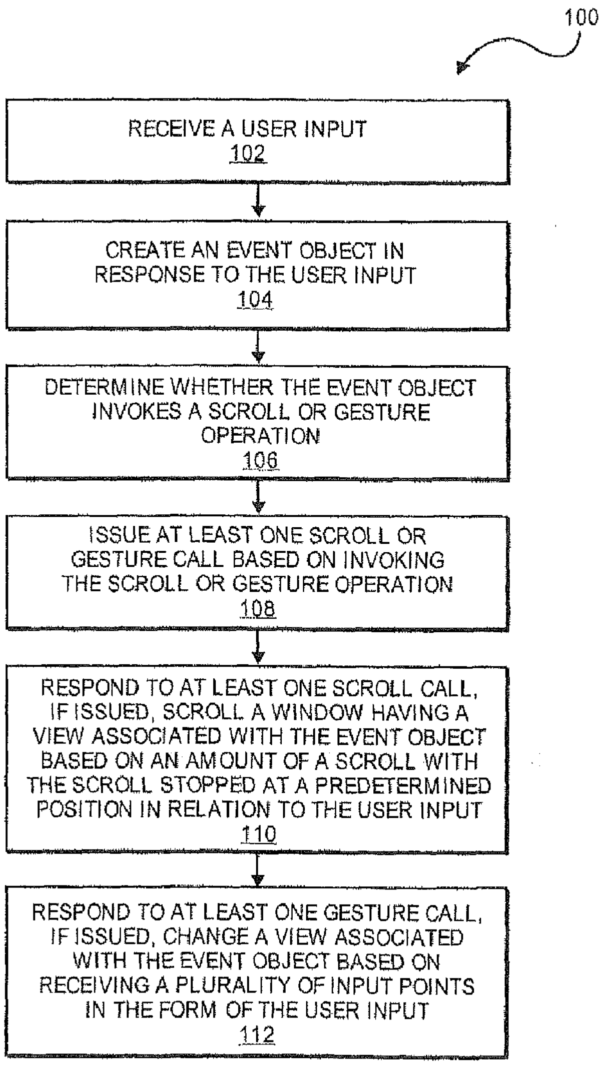

100
RECEIVE A USER INPUT
102
CREATE AN EVENT OBJECT IN RESPONSE TO THE USER INPUT
104
DETERMINE WHETHER THE EVENT OBJECT INVOKES A SCROLL OR GESTURE OPERATION
106
ISSUE AT LEAST ONE SCROLL OR GESTURE CALL BASED ON INVOKING THE SCROLL OR GESTURE OPERATION
108
RESPOND TO AT LEAST ONE SCROLL CALL, IF ISSUED, SCROLL A WINDOW HAVING A VIEW ASSOCIATED WITH THE EVENT OBJECT BASED ON AN AMOUNT OF A SCROLL WITH THE SCROLL STOPPED AT A PREDETERMINED POSITION IN RELATION TO THE USER INPUT
110
RESPOND TO AT LEAST ONE GESTURE CALL, IF ISSUED, CHANGE A VIEW ASSOCIATED WITH THE EVENT OBJECT BASED ON RECEIVING A PLURALITY OF INPUT POINTS IN THE FORM OF THE USER INPUT
112

FIG. 1

The method 100 begins by receiving a user input at block 102. ['915 patent, col. 6:32-34.] The user input may be from an input key, button, wheel, touch, or other means for interacting with the device. ['915 patent, col. 6:34-36.] The method 100 next creates an event object in response to the user input at block 104. ['915 patent, col. 6:36-37.] The method 100 determines whether the event object invokes a scroll or gesture operation at block 106. ['915 patent, col. 6:37-39.] The '915 patent explains, for example, that a single touch that drags a distance across a display of the device may be interpreted as a scroll operation, and that in one embodiment, a two or more finger touch of the display may be interpreted as a gesture operation. ['915 patent, col. 6:39-41.] Determining whether the event object invokes a scroll or gesture operation may also be based on receiving a drag user input for a certain time period. ['915 patent, col. 6:41-46.] The method 100 next issues at least one scroll or gesture call based on invoking the scroll or gesture operation at block 108. ['915 patent, col. 6:46-48.] If a scroll call is issued, the method 100 responds by scrolling a window having a view (e.g., web, text, or image content) associated with the event object based on an amount of a scroll with the scroll stopped at a predetermined position in relation to the user input, as shown in block 110. ['915 patent, col. 6:48-53.] For example, an input may end at a certain position on a display of the device, and the scrolling may continue until reaching a predetermined position in relation to the last input received from the user. ['915 patent, col. 6:53-56.] Finally, at block 112, the method 100 responds to at least one gesture call, if issued, by changing a view associated with the event object based on receiving a plurality of input points in the form of the user input at block 112. ['915 patent, col. 6:56-60.] Changing the view may involve scaling the view associated with the event object by zooming in or zooming out based on receiving the user input. ['915 patent, col. 7:4-10.]

288. Figures 6A-D illustrate the process of scrolling content on a display and "rubberbanding" when a scrolling region exceeds a window edge. ['915 patent, col. 8:61-67.] As the '915 patent explains, the user interface may display "a portion of a list of emails," as shown in Fig. 6A. ['915 patent, col. 9:13-14.]

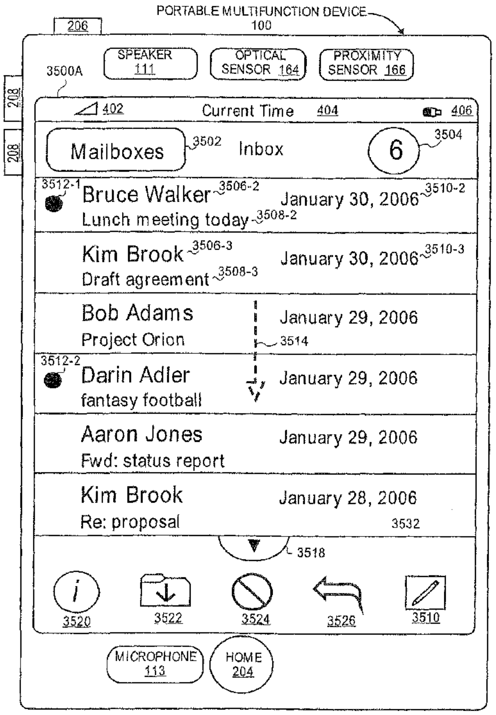


FIG. 6A

289. A user may scroll the list vertically (e.g., in the direction of arrow 3514) so that a different portion of the list is displayed, as shown in Fig. 6B. ['915 patent, col. 9:10-27.]

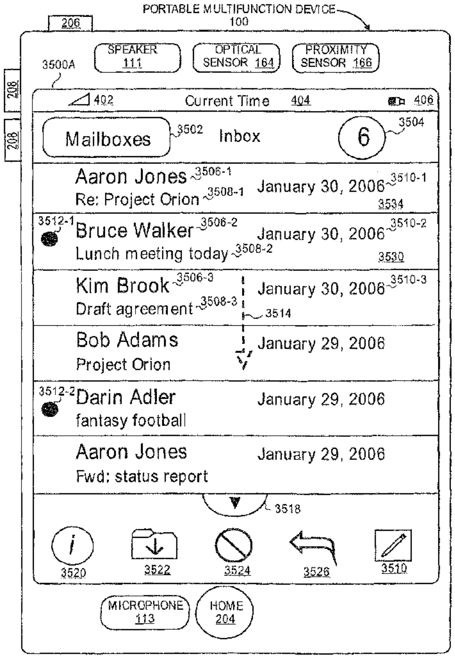


FIG. 6B

If the user continues to scroll past the terminus of the list, then an area beyond the edge of the list may be displayed (area 3536), as illustrated in Fig. 6C. ['915 patent, col. 9:29-38.]

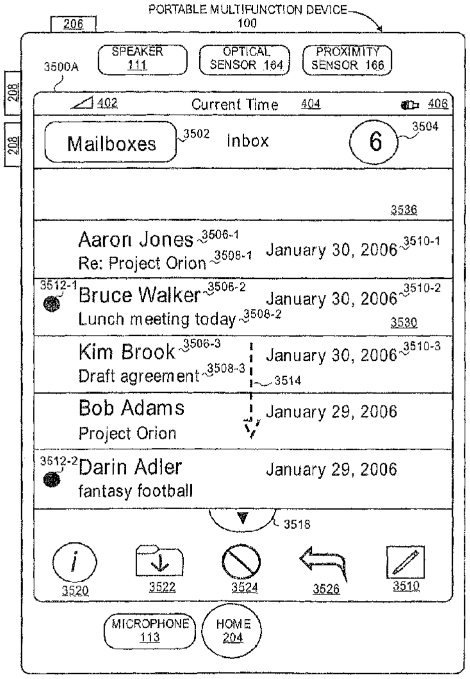


FIG. 6C

290. Once the vertical swipe is complete, e.g. the user lifts his/her finger off of the touch screen display, the list scrolls back in the opposite direction until the area beyond the terminus of the list is no longer displayed, as illustrated in Fig. 6D. ['915 patent, col. 9:39-46.]

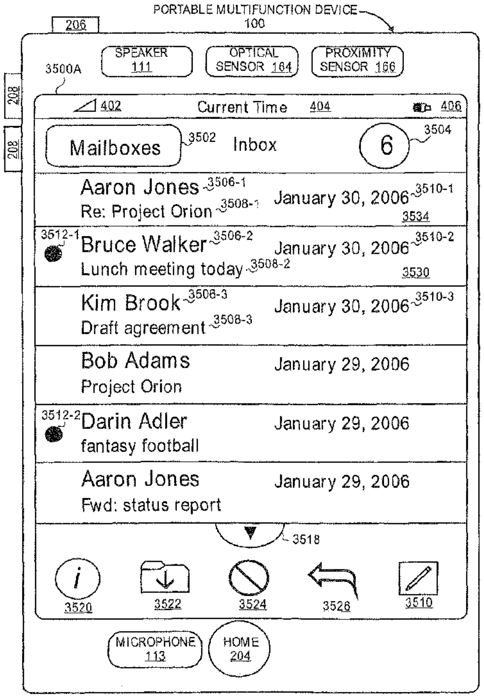


FIG. 6D

291. Figures 16A-C illustrate the process of scaling (e.g., zooming) content on a display in response to a multi-input point gesture. ['915 patent, col. 13:37 – col. 14:24.] As the '915 patent explains, in certain embodiments, a user input in the form of two or more input points (e.g., two fingers) moves together or apart to invoke a gesture event that performs a scaling transform on the view associated with the user input. ['915 patent, col. 13:37-40.]

292. FIG. 16A illustrates a display 1604 of a device having a first scaling factor of a view 1616. A user input (e.g., two fingers 1608 and 1610 moving toward each other) associated with the view 1614 is interpreted as a gesture event to zoom in. ['915 patent, col. 13:52-57.]

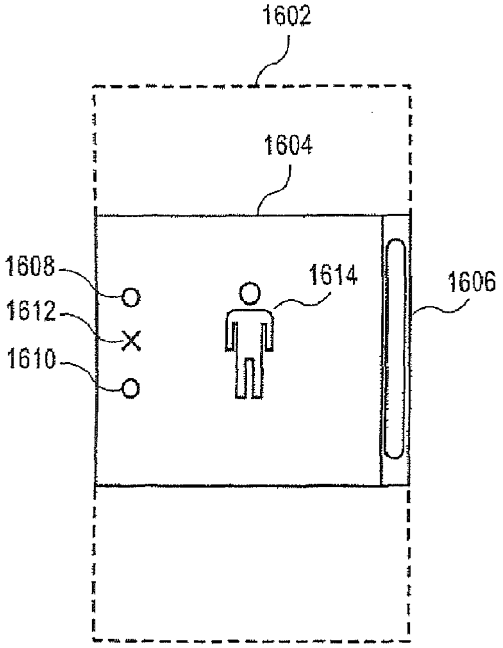


FIG. 16A

293. The gesture operation zooms in from view 1614 to view 1664 having a second scale factor as illustrated in Figure 16B. ['915 patent, col. 13:52-57.] The dashed regions 1602 and 1650 represent the total area of the content with the only content being displayed in the display area 1604 and 1652. ['915 patent, col. 13:57-59.]

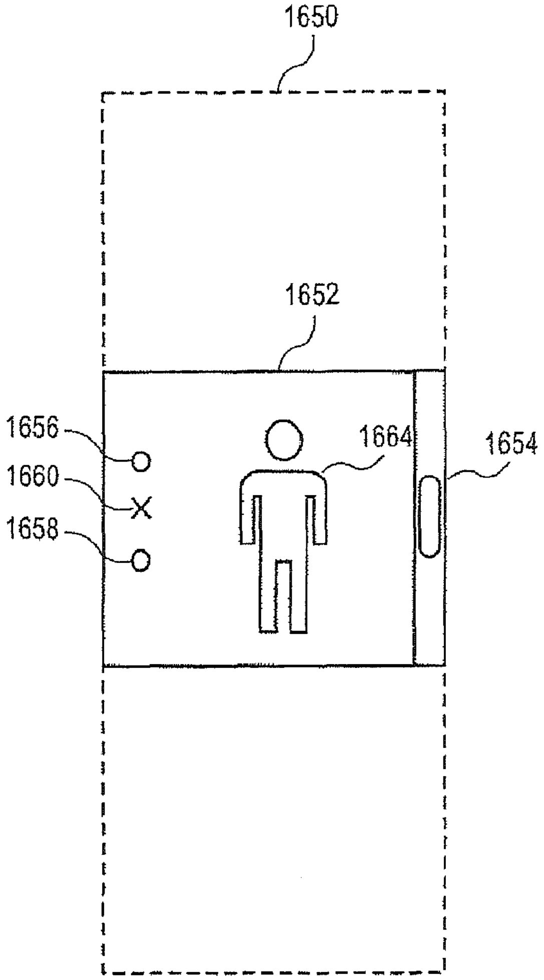


FIG. 16B

294. In performing the scaling transform from Figure 16A to Figure 16B in this embodiment, the center of the gesture event, center 1612 for Figure 16A and center 1660 for Figure 16B, remains in the same position with respect to the display 1604. ['915 patent, col. 13:59-63.] In the embodiment, the scroll indicator 1606 also shrinks to become scroll indicator 1654 during the transform to indicate that a smaller portion of the total content 1650 is being displayed on display 1604 as a result of the zoom in operation. ['915 patent, col. 13:63-66.] The dashed region 1650 is larger than the dashed region 1602 to represent that a larger portion of content is not being displayed on display 1652 in FIG. 16B as a result of the zoom in operation. ['915 patent, col. 13:67 – col. 14:3.] The '915 patent also teaches that in some embodiments, the

scale factor of a view can be reduced (e.g., from scale factor of 2X to 1X) by moving a pair of input points (e.g., fingers) together. ['915 patent, col. 14:4-24; Fig. 16C.]

B. Apple's Practice of the '915 Patent

295. My use of Apple's iPhone and iPad products, along with my review of related materials detailing their operations, confirms that Apple's products practice the claims of the '915 patent. It is readily apparent that Apple's products have touch-sensitive displays that permit single-touch scrolling, with the amount of scrolling determined by the user input (with scroll-indicators at the content edge of windows); multi-touch gestures such as pinch zooming, with the direction and amount of zooming based on user input, or the rotation of a view based on user input; and rubberbanding by a predetermined amount when scrolling exceeds a window edge.

296. Related materials confirm that these features are implemented via objects generated in response to user input. For example, the "Event Handling Guide for iOS," explains how the "Multi-Touch Interface of iPhones, iPads, and iPod touches" generates event "objects" when users touch their displays, which in turn call various functions, based on the characteristic of the touch. (Guide at 6, 9 ("An event is an object that represents a user action detected by hardware on the devices . . . for example, a finger touching the screen."); see Guide at 16-36 generally.) The Guide explains that "a pinch-close gesture has two touches," while there are also "single-finger gestures" such as "a drag." (Guide at 17.) Supported "gestures include tapping (one or multiple times), pinching (to zoom a view in or out), swiping, panning or dragging a view, and using two fingers to rotate a view." (Guide at 18, 40.) And the Guide describes the "Gesture Recognizers" specific to pinch-zooming, dragging, swiping, and rotating, along with exemplary code for handling such gestures. (Guide at 40-45.) iOS uses the number of touches, location of touches, duration of touches, and distance between touches to distinguish between and implement these various features. (Guide at 17-20, 27, 40-45.)

297. The testimony of one of the inventors of the '915 patent confirms that Apple's products practice the claims of the '915 patent. At his deposition, Andrew Platzer confirmed that Apple's products have touch-sensitive displays that permit rubberbanding, single-touch scrolling, multi-touch gestures (including pinch-zoom or "scaling"), and create event objects in response to

user input. (Platzer Depo. (Oct. 18, 2011) Tr. at 37, 45, 51, 70, 72, 80-81, 84-85, 96, 108, 112-13, 118.)

298. Accordingly, it is my opinion that Apple's touch screen products practice the asserted claims of the '915 patent, and their ordinary and intended use practices the asserted method claims of the '915 patent.

C. Priority Date of the '915 Patent

299. I intend to rely upon the documentary evidence and testimony of the named inventors of the '915 patent or other witnesses to testify regarding facts relevant to the conception and reduction of to practice of the claimed invention prior to the filing date of the patent.

300. I have reviewed the documentary evidence regarding the design and implementation work done on the inventions claimed in the '915 patent, including the deposition transcript of Andrew Platzer and Scott Herz, and source code. (*See* Platzer Depo. Tr. (Oct. 18, 2011) at 118-120; Herz Depo. Tr. (Oct. 14, 2011) at 148.) From that evidence, it appears that the claims of the '915 patent were conceived no later than the summer and fall of 2005, and that the asserted claims were wholly or substantially reduced to practice by the fall of 2005. I am informed that Mr. Platzer and Mr. Herz worked on an application framework known as "UIKit" used on the iPhone to build other iPhone applications. UIKit provides shared code that other applications can use. As part of their work on UIKit, the inventors added certain functionalities to the UIKit that embodied claims of the '915 patent. For example, by August 2005 the inventors had added scrolling improvements to the UIKit and by November 2005 they had incorporated a rubberbanding feature to the UIKit. I also understand the claims were constructively reduced to practice on January 7, 2007 in U.S. Patent Application No. 11/620,717. Documents relating to these facts are found in, for example: APL-ITC796-0000079762-768; APL-ITC796-0000079776-787; APL-ITC796-0000079794-801; APL-ITC796-0000079816-821; and APL-ITC796-0000079825-830.

D. Samsung's Infringement of the '915 Patent

301. In the discussion that follows, I analyze whether certain Samsung products embody the apparatus claims of the '915 patent and whether the ordinary and intended use of the

Samsung Accused Products would practice the method claims of the patent. For purposes of this section of my Report, the "Samsung Accused Products" include all of the following Samsung products: Acclaim, Captivate, Continuum, Droid Charge, Epic 4G, Exhibit 4G, Fascinate, Galaxy Ace, Galaxy Prevail, Galaxy S (i9000), Galaxy S 4G, Galaxy S II (including the i9100, T-Mobile, AT&T, Epic 4G Touch and Skyrocket variants), Galaxy S Showcase (i500), Galaxy Tab 7.0, Galaxy Tab 10.1, Gem, Gravity Smart, Indulge, Infuse 4G, Intercept, Mesmerize, Nexus S, Nexus S 4G, Replenish, Sidekick, Transform, and Vibrant.

302. In performing this analysis I reviewed the '915 patent and its file history, tested the operation of these Samsung Accused Products, reviewed source code that Samsung produced prior to the March 8 fact discovery cutoff, and reviewed other materials described in this Report. Because the Samsung source code is built upon the foundation of publicly-available Android code, I reviewed portions of that Android code and its accompanying documentation. I have analyzed Samsung source code on at least one Accused Product representative of each major release of Android that appears on the Accused Products. I reviewed source code that implements the accused functionalities of the '915 patent on, among other devices, the Samsung Captivate (Android 2.1), the Samsung Vibrant, (Android 2.2), the Samsung Galaxy S II (Android 2.3), and the Samsung Galaxy Tab 10.1 (Android 3.1). I have compared portions of the relevant code on each of these devices to analogous code (where available) on other Accused Products running that version, as well as the publicly available version of each major Android release. Based on those comparisons, I conclude that, for each major Android release, all of the Accused Products based on that release implement the accused functionalities of the '915 patent in substantially the same way as the representative device for that release whose source code I have analyzed and cited in this Report.

303. In the paragraphs that follow, I will set forth the claims of the '915 patent for which it is my opinion that Samsung Accused Products, or the ordinary and intended use of Samsung Accused Products, meets every limitation of the claim.

304. By "ordinary and intended use" in this section of my Report, I mean actions that virtually every user of a Samsung Accused Product would perform when using the Accused

Product, and which Samsung encouraged and intended the user to perform. For example, manuals included with Samsung Accused Products instruct users to use a finger to scroll and two or more fingers to zoom. (*See, e.g.*, APLNDC-Y0000057563, APLNDC-Y0000058568-569, APLNDC-Y0000060382, APLNDC-Y0000061404, APLNDC-Y0000065325.) In addition, the ordinary use of each Accused Device involves using one-finger scroll and two-finger zoom. Accordingly, it is my opinion that all or virtually all users of the Samsung Accused products would engage in direct infringement of the '915 patent. Because Samsung encouraged and intended this direct infringement by end users, it is my opinion that the Samsung defendants have indirectly infringed the method claims of the '915 patent discussed below.

305. Attached as Exhibits 16 and 17 are exemplary claim charts that illustrate the infringement of the claims below by the Galaxy Tab 10.1 (Exhibit 16) and the Galaxy S II (Exhibit 17). Where source code is cited in the Galaxy S II claim chart (corresponding to Android 2.3), reference is also made to analogous code in Android 2.2 (as exemplified by the Samsung Vibrant) and Android 2.1 (as exemplified by the Samsung Captivate).

306. **Claim 1**. Claim 1 recites:

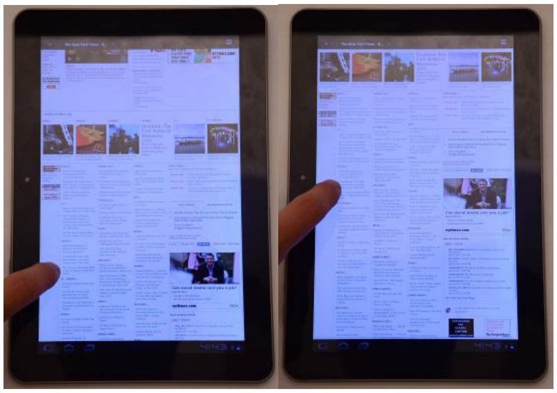
> A machine implemented method for scrolling on a touch-sensitive display of a device comprising:
>
> [a] receiving a user input, the user input is one or more input points applied to the touch-sensitive display that is integrated with the device;
>
> [b] creating an event object in response to the user input;
>
> [c] determining whether the event object invokes a scroll or gesture operation by distinguishing between a single input point applied to the touch-sensitive display that is interpreted as the scroll operation and two or more input points applied to the touch-sensitive display that are interpreted as the gesture operation;
>
> [d] issuing at least one scroll or gesture call based on invoking the scroll or gesture operation;
>
> [e] responding to at least one scroll call, if issued, by scrolling a window having a view associated with the event object based on an amount of a scroll with the scroll stopped at a predetermined position in relation to the user input; and
>
> [f] responding to at least one gesture call, if issued, by scaling the view associated with the event object based on receiving the two or

more input points in the form of the user input.

307. In my opinion, each of the Accused Products meets each and every limitation of claim 1 of the '915 patent literally and, in the alternative, under the doctrine of equivalents, as explained below. Videos of various Accused Products performing the limitations of this claim are included in Exhibit 18 (Galaxy Tab 10.1), Exhibit 19 (Galaxy S II), Exhibit 20 (Vibrant), and Exhibit 21 (Captivate).

308. **Claim 1 – Preamble: "A machine implemented method for scrolling on a touch-sensitive display of a device comprising**." Each of the Accused Products is either a smartphone or tablet running a version of the Android operating system. Each '915 Accused Product, which includes a touch-sensitive display, performs a machine implemented method for scrolling on the touch-sensitive display.

309. For example, the Galaxy Tab 10.1 includes a touch-sensitive display and performs a machine implemented method for scrolling on the touch-sensitive display. Below is an illustration of the Galaxy Tab 10.1 scrolling an image on the touch-sensitive display:

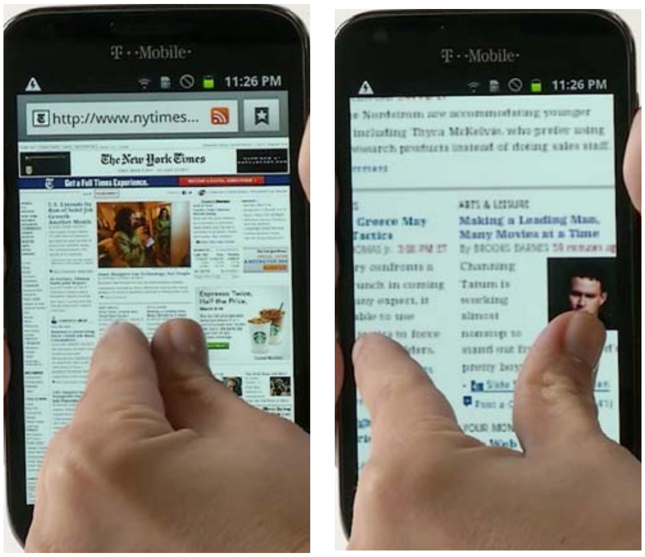


(Scroll operation when one input point is applied.)

(Gesture operation when two or more input points are applied.)

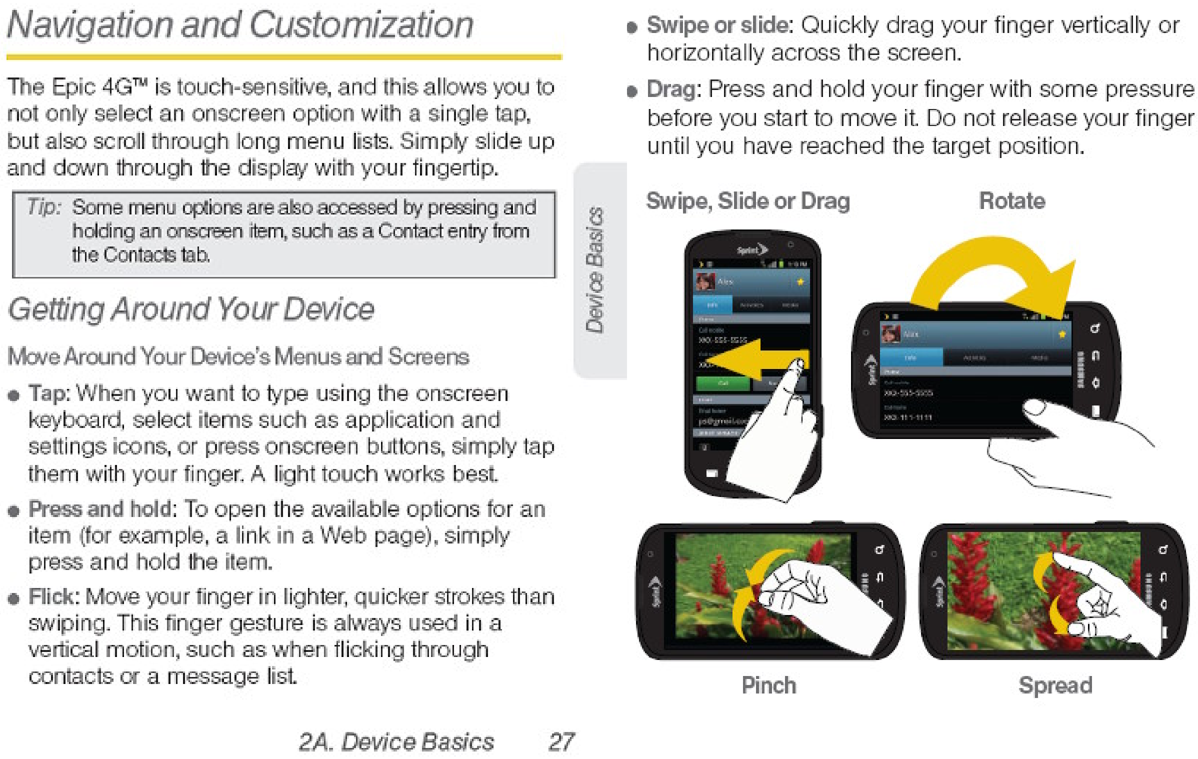
310. For example, the Galaxy S II includes a touch-sensitive display and performs a machine implemented method for scrolling on the touch-sensitive display.

(Scroll operation when one input point is applied.)



(Gesture operation when two or more input points are applied.)

311. User manuals for Samsung products teach users how to scroll. For example, the user manual for the Epic 4G includes the following description:



Navigation and Customization

The Epic 4G™ is touch-sensitive, and this allows you to not only select an onscreen option with a single tap, but also scroll through long menu lists. Simply slide up and down through the display with your fingertip.

> *Tip:* Some menu options are also accessed by pressing and holding an onscreen item, such as a Contact entry from the Contacts tab.

Getting Around Your Device

Move Around Your Device's Menus and Screens

- **Tap:** When you want to type using the onscreen keyboard, select items such as application and settings icons, or press onscreen buttons, simply tap them with your finger. A light touch works best.
- **Press and hold:** To open the available options for an item (for example, a link in a Web page), simply press and hold the item.
- **Flick:** Move your finger in lighter, quicker strokes than swiping. This finger gesture is always used in a vertical motion, such as when flicking through contacts or a message list.
- **Swipe or slide:** Quickly drag your finger vertically or horizontally across the screen.
- **Drag:** Press and hold your finger with some pressure before you start to move it. Do not release your finger until you have reached the target position.

2A. Device Basics 27

312. In the manual displayed above, a Swipe, Slide, or Drag, all of which invoke a scroll operation, are distinguished from a Pinch or Spread, which invoke a gesture operation.

313. To the extent that the preamble is found to be a limitation and is not met literally, in my opinion it is met under the doctrine of equivalents because each of the Accused Products perform steps insubstantially different from scrolling on a touch-sensitive display of a device, and accomplishes the same function in the same way to achieve the same result.

314. **Claim 1 – Element [a] "receiving a user input, the user input is one or more input points applied to the touch-sensitive display that is integrated with the device."** In my opinion, each of the Accused Products performs this step of claim 1.

315. The Accused Products receive a user input. The user input includes one or more input points (one or more fingers) applied to the touch-sensitive display that is integrated with the Samsung device.

316. For example, the Galaxy Tab 10.1 receives user a user input with one input point (one finger) applied to the touch-sensitive display as illustrated above. I also note that the touch-sensitive display is integrated into the Galaxy Tab 10.1.

317. For example, the Galaxy S II receives a user input with one input point (one finger) applied to the touch-sensitive display as shown above. The touch-sensitive display is integrated into the Galaxy S II.

318. Based on my observations of the Accused Products, as well as my analysis of the source code for each major release of Android running on the Accused Products (Android 2.1, 2.2, 2.3, and 3.1), I have determined that each Accused Product receives a user input, where the user input is one or more input points applied to the touch-sensitive display that is integrated with the device. The claim chart in Exhibit 17 identifies analogous code that satisfies this element in Android 2.1, 2.2, and 2.3.

319. To the extent that this limitation is not met literally, in my opinion it is met under the doctrine of equivalents because each of the Accused Products perform steps insubstantially different from machines receiving a user input, the user input is one or more input points applied to the touch-sensitive display that is integrated with the device, and accomplishes the same function in the same way to achieve the same result.

320. **Claim 1 – Element [b] "creating an event object in response to the user input."** In my opinion, each of the Accused Products performs this step of claim 1.

321. Each of the Accused Products, via the Android platform on which they operate, creates an event object in response to the user input.

322. Under the public Android platform, a MotionEvent object is created in response to a touch on the touch screen. (http://developer.android.com/reference/android/view/MotionEvent.html.)

323. I have confirmed the public Android code also appears in the Accused Products. For example, in the Galaxy Tab 10.1 tablet, which runs a version of Android 3.1, the user input is processed by the device driver, which passes the input into user space and parses it into an event object referred to as the "MotionEvent" object. This object is an event object created by the

method InputConsumer::populateMotionEvent(). (*See* frameworks/base/libs/ui/inputTransport.cpp:683-712 [SAMNDCA-C000002822]; *see also* frameworks/base/libs/ui/input.cpp:351-382 [SAMNDCA-C000002830 to -C000002831] (MotionEvent::initialize() method)).

324. Based on my observations of the Accused Products, as well as my analysis of the source code for each major release of Android running on the Accused Products (Android 2.1, 2.2, 2.3, and 3.1), I have determined that each Accused Product practices includes similar computer code that creates an event object in response to user input. The claim chart in Exhibit 17 identifies analogous code that satisfies this element in Android 2.1, 2.2, and 2.3.

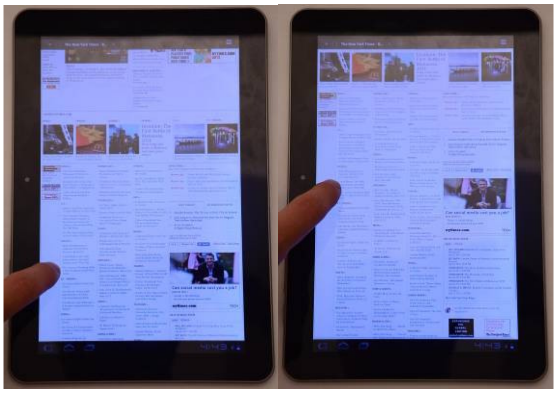
325. Furthermore, Ioi Lam confirmed at his 30(b)(6) deposition that the Android Platform has "event objects." See Ioi Lam Depo. Tr., Mar. 8, 2012 (75:17-76:23).

326. To the extent that this limitation is not met literally, in my opinion it is met under the doctrine of equivalents because each of the Accused Products perform steps insubstantially different from creating an event object in response to the user input, and accomplishes the same function in the same way to achieve the same result.

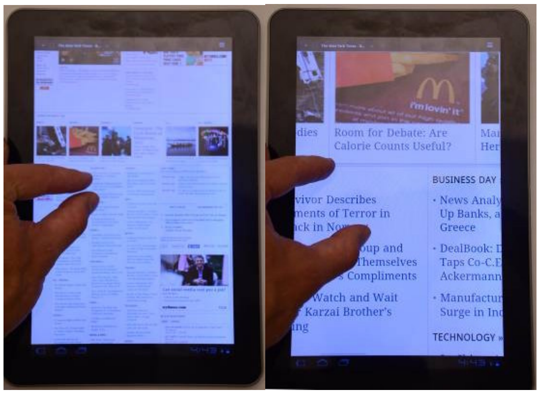
327. **Claim 1 – Element [c]: "determining whether the event object invokes a scroll or gesture operation by distinguishing between a single input point applied to the touch-sensitive display that is interpreted as the scroll operation and two or more input points applied to the touch-sensitive display that are interpreted as the gesture operation"** In my opinion, each of the Accused Products performs this step of claim 1.

328. The Accused Products determine whether an event object invokes a scroll or gesture operation by distinguishing between a single input point (one finger) applied to the touch-sensitive display that is interpreted as the scroll operation and two or more input points (more than one finger) applied to the touch-sensitive display that are interpreted as the gesture operation.

329. For example, the Galaxy Tab 10.1 tablet distinguishes between a scroll operation when one finger is applied to the touch-sensitive display and a gesture operation when two or more fingers are applied to the touch-sensitive display.

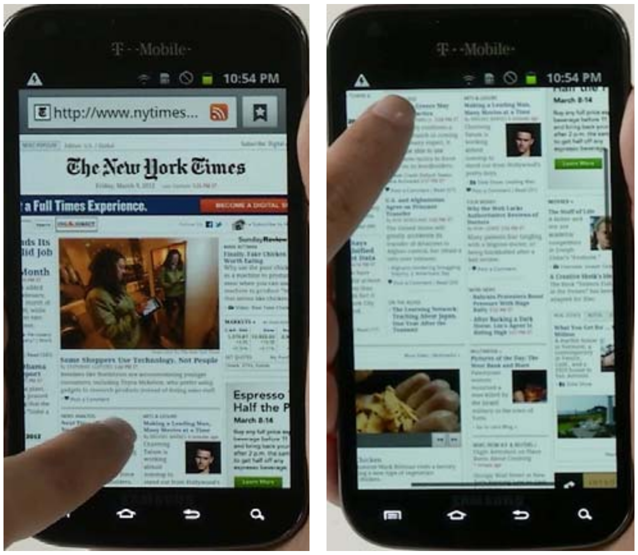


(Scroll operation when one input point is applied.)

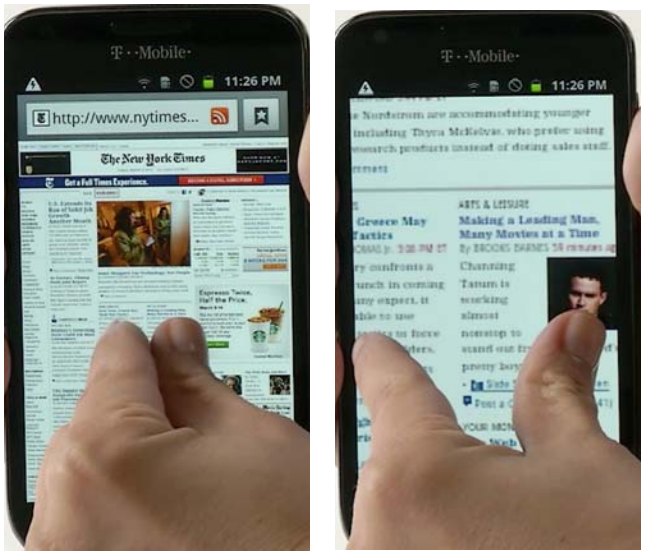


(Gesture operation when two or more input points are applied.)

330. For example, the Galaxy S II phone distinguishes between a scroll operation when one finger is applied to the touch-sensitive display and a gesture operation when two or more fingers are applied to the touch-sensitive display, as illustrated below:



(Scroll operation when one input point is applied.)



(Gesture operation when two or more input points are applied.)

331. For example, in the Galaxy Tab 10.1 tablet, which runs Android 3.1, the WebView class's handleQueuedMotionEvent() method interprets the input points associated with the MotionEvent object it processes. The handleQueueMotionEvent() method distinguishes between a single input point (ev.getPointerCount == 1) and two or more input points (ev.getPointerCount > 1). (*See* WebView.java:10281-10314 [SAMDNCA-C000002857].) If one input point is

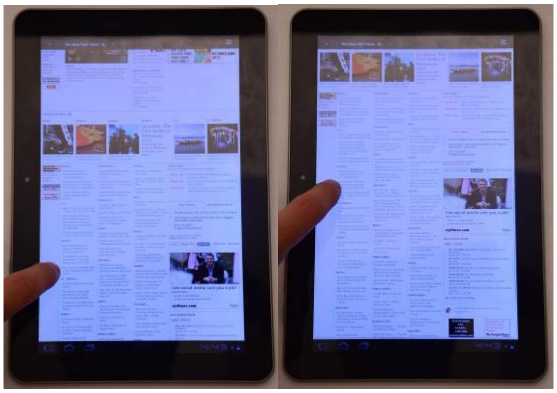
detected, the contact is interpreted as a scroll operation in handleTouchEventCommon(). (*See* WebView.java:10312 [SAMNDCA-C000002857].) If two or more input points are detected, the contact is interpreted as a gesture operation via a call to handleMultiTouchInWebView(). (*See* WebView.java:10302 [SAMNDCA-C000002857]; WebView.java:7887-7944 [SAMNDCA-C000002858].)

332. Based on my inspection of Samsung source code for each major release of Android running on the Accused Products (Android 2.1, 2.2, 2.3, and 3.1), I have determined that each Accused Product includes similar computer code that distinguishes between a single input point (one finger) applied to the touch-sensitive display that is interpreted as the scroll operation and two or more input points (more than one finger) applied to the touch-sensitive display that are interpreted as the gesture operation. The claim chart in Exhibit 17 identifies analogous code that satisfies this element in Android 2.1, 2.2, and 2.3.

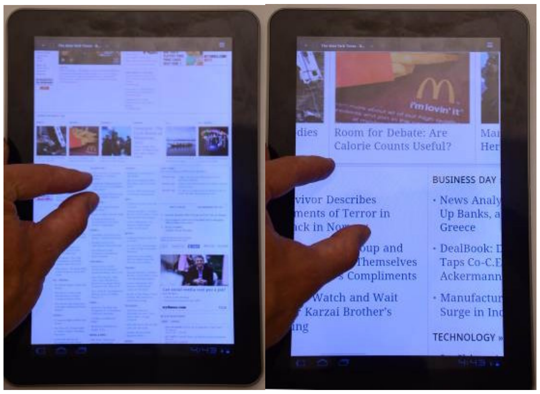
333. To the extent that this limitation is not met literally, in my opinion it is met under the doctrine of equivalents because each of the Accused Products perform steps insubstantially different from determining whether the event object invokes a scroll or gesture operation by distinguishing between a single input point applied to the touch-sensitive display that is interpreted as the scroll operation and two or more input points applied to the touch-sensitive display that are interpreted as the gesture operation, and accomplishes the same function in the same way to achieve the same result.

334. **Claim 1 – Element [d]: "issuing at least one scroll or gesture call based on invoking the scroll or gesture operation."** Each of the Accused Products issues a scroll call or a gesture call based on invoking the scroll or gesture operation.

335. For example, as illustrated below, the Galaxy 10.1 tablet issues a scroll call when the scroll operation is invoked. Alternatively, the tablet issues a gesture call when the gesture operation is invoked.

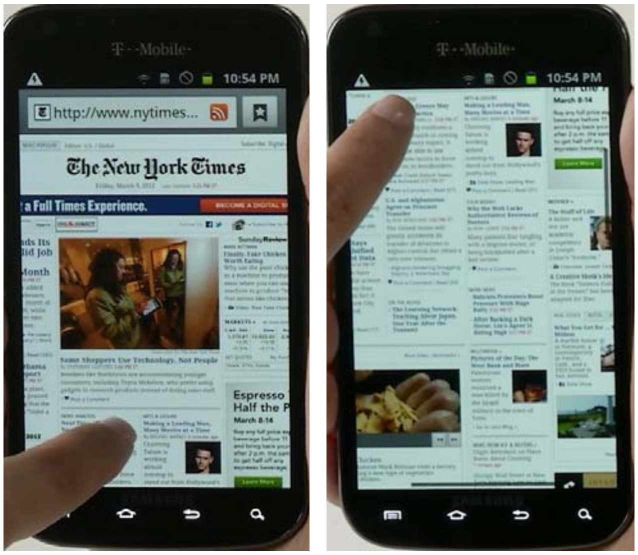


(Scroll call when scroll operation is invoked.)



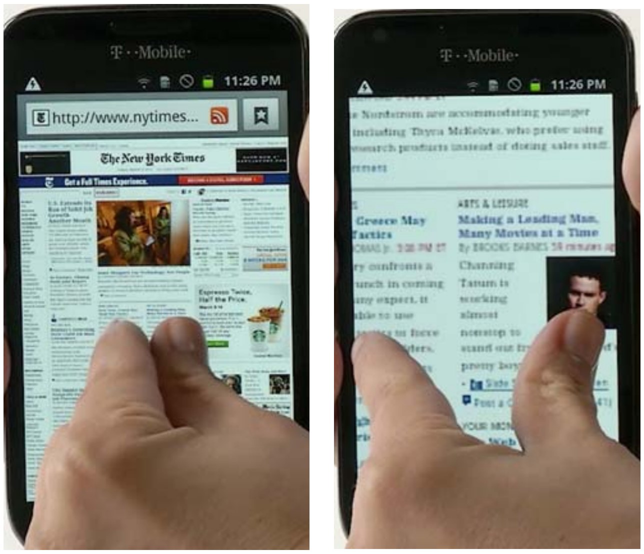
(Gesture call (scaling) when gesture operation is invoked.)

336. For example, the Galaxy S 2 phone issues a scroll call when the scroll operation is invoked.



(Scroll operation when one input point is applied.)

337. The phone issues a gesture call when the gesture operation is invoked.



(Gesture operation when two or more input points are applied.)

338. For example, in the Galaxy 10.1 tablet, if one input point is detected, handleQueuedMotionEvent() will call handleTouchEventCommon() (WebView.java:10312 [SAMNDCA-C000002926]), which issues a scroll call to doDrag() or doFling().

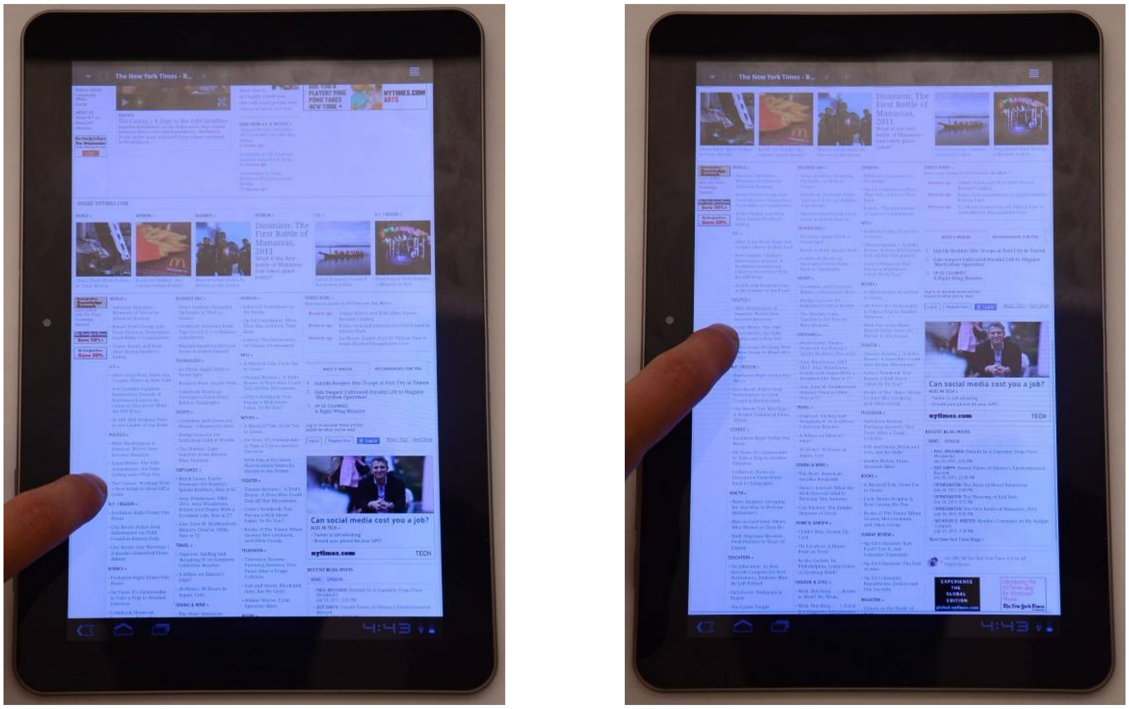
(WebView.java:7617, 7772 [SAMNDCA-C000002926, -C000002930]) If two or more input points are detected, the contact is interpreted as a gesture operation and a call to handleMultiTouchInWebView() is made. (*See* WebView.java:10302 [SAMNDCA-C000002857]; WebView.java:7887-7944 [SAMNDCA-C000002858].)

339. Based on my inspection of Samsung source code for each major release of Android running on the Accused Products (Android 2.1, 2.2, 2.3, and 3.1), I have determined that each Accused Product includes similar computer code that issues at least one scroll or gesture call based on invoking the scroll or gesture operation. The claim chart in Exhibit 17 identifies analogous code that satisfies this element in Android 2.1, 2.2, and 2.3.

340. To the extent that this limitation is not met literally, in my opinion it is met under the doctrine of equivalents because each of the Accused Products perform steps insubstantially different from issuing at least one scroll or gesture call based on invoking the scroll or gesture operation, and accomplishes the same function in the same way to achieve the same result.

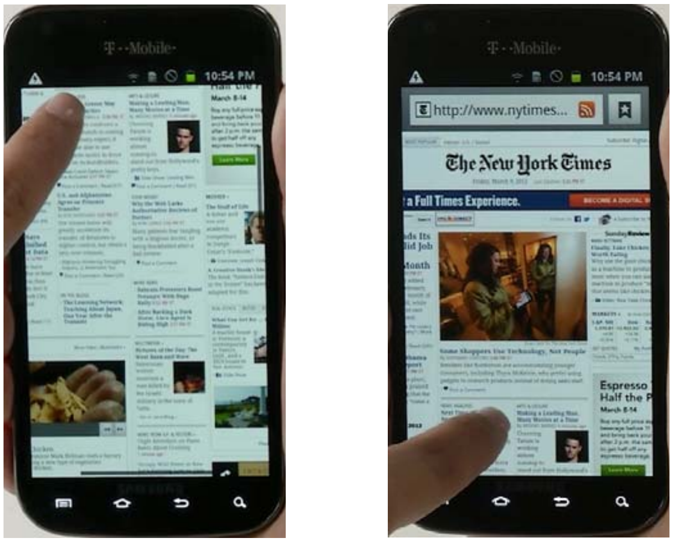
341. **Claim 1 – Element [e] "responding to at least one scroll call, if issued, by scrolling a window having a view associated with the event object based on an amount of a scroll with the scroll stopped at a predetermined position in relation to the user input."** Each of the Accused Products responds to a scroll call, if issued, by scrolling a window having a view associated with the event object based on an amount of a scroll with the scroll stopped at a predetermined position in relation to the user input.

342. For example, the Galaxy 10.1 tablet will respond to at least one scroll call by scrolling a window having a view associated with the MotionEvent object, based on an amount of a scroll with the scroll stopped at a predetermined position in relation to the user input, as illustrated below.


The New York Times
Disunion: The First Battle of Manassas, 2011
Can social media cost you a job?
nytimes.com
4:43

(Screenshot of the Samsung Galaxy Tab 10.1 scrolling an image.)

343. For example, the Galaxy S2 phone will respond to at least one scroll call by scrolling a window having a view associated with the MotionEvent based on an amount of a scroll with the scroll stopped at a predetermined position in relation to the user input, as illustrated below.



344. For example, in the Galaxy 10.1 tablet, the handleTouchEventCommon() method calls doFling() for a scroll operation. (*See* WebView.java:7272-7821 [SAMNDCA-C000002919 to –C000002931] (call done at 7772).) doFling() then calls the Overscroller.fling() method. (*See* WebView.java:9236-9376 [SAMNDCA-C000002932 to –C000002935].) Overscroller.fling() itself calls two instances of the SplineOverScroller class, each of which is responsible for scrolling in one axis (i.e., one scrolls horizontally and the other scrolls vertically). (*See* OverScroller.java:406-448 [SAMNDCA-C000002945].) The SplineOverScroller class thus maintains state information for the fling. (*See id.*)

345. The SplineOverScroller class tracks the start points, start time, duration, total distance, and the final position for the fling. (OverScroller.java:748-782 [SAMNDCA-C000002952 to –C000002953].) The SplineOverScroller.fling() function thus determines the final position of the fling before beginning the fling operation begins.

346. The actual rendering of the fling occurs subsequently as part of the drawing cycle. At the end of an event processing cycle, the method computeScroll() is called to compute which part of the view should be rendered to the user. (*See* WebView.java:3568-3654 [SAMNDCA-C000002958 to –C000002959]. The computeScroll() method uses the SplineOverScroller class to extract the state information for the fling. (*See id.*) Afterwards, it calls WebView.overScrollBy() to scroll the content—this method calculates maximums for the distance the user can scroll beyond the edge of the content and whether content should be fixed to a particular axis. (*See id.*; *see also* View.java:11663-11715 [SAMNDCA-C000002960 to –C000002961] (WebView.overScrollBy()).) onOverScrollBy() itself calls onOverScroller() to ensure the intended scroll coordinates are valid and then calls View.scrollTo() to invoke the scroll operation. (*See* View.java:11663-11715 [SAMNDCA-C000002960 to –C000002961]; WebView.java:3130-3162 [SAMDNCA-2962].) View.scrollTo() scrolls the window (setting mScrollX and mScrollY) based on the amount of a scroll with the scroll stopped at a "predetermined position in relation to the user input." (*See* WebView.java:3130-3162 [SAMDNCA-2962].)

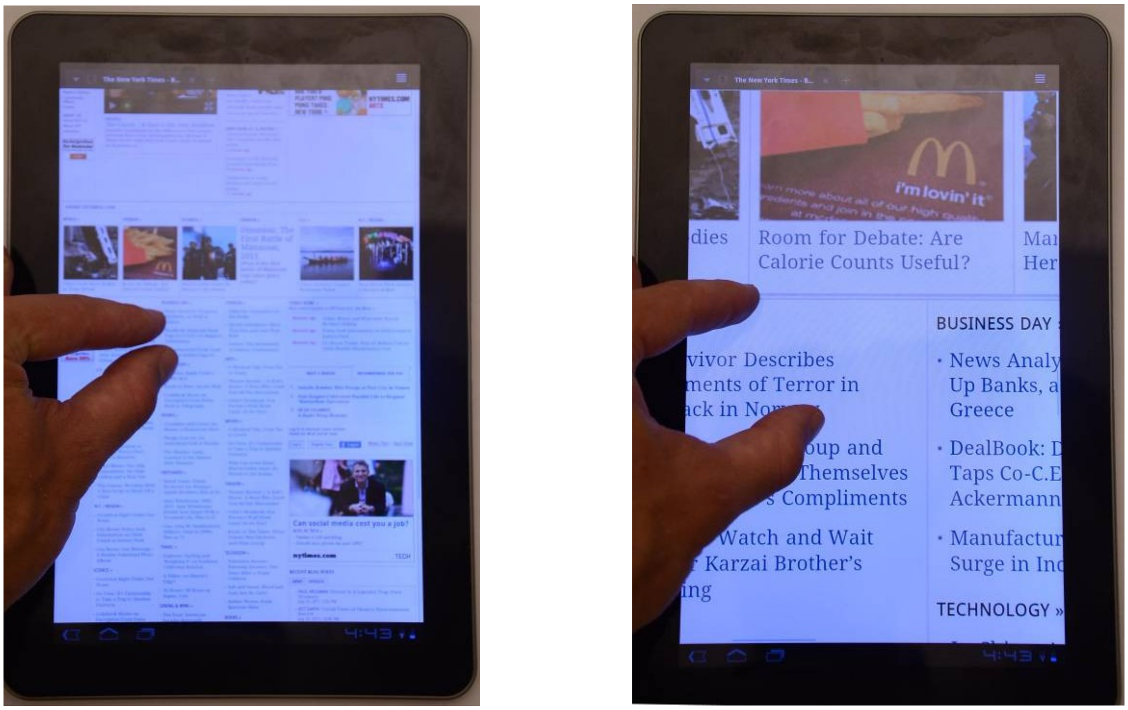
347. Alternatively, it is my opinion that the scroll stops at a "predetermined position in relation to the user input" because after the mScrollX and mScrollY fields are set (or determined), the WebView.onDraw() method is subsequently called to translate and draw the view shown to the user. (*See* WebView.java:4261-4418 [SAMNDCA-C000002965 to –C000002968] (with call to trackFPS() at 4416); WebView.java:8757-8791 [SAMNDCA-C000002964] (trackFPS() translates based on mScrollX and mScrollY then draws).)

348. Based on my inspection of Samsung source code for each major release of Android running on the Accused Products (Android 2.1, 2.2, 2.3, and 3.1), I have determined that each Accused Product includes similar computer code that responds to at least one scroll call by scrolling a window having a view associated with the MotionEvent based on an amount of a scroll with the scroll stopped at a predetermined position in relation to the user input. The claim chart in Exhibit 17 identifies analogous code that satisfies this element in Android 2.1, 2.2, and 2.3.

349. To the extent that this limitation is not met literally, in my opinion it is met under the doctrine of equivalents because each of the Accused Products perform steps insubstantially different from responding to at least one scroll call, if issued, by scrolling a window having a view associated with the event object based on an amount of a scroll with the scroll stopped at a predetermined position in relation to the user input, and accomplishes the same function in the same way to achieve the same result.

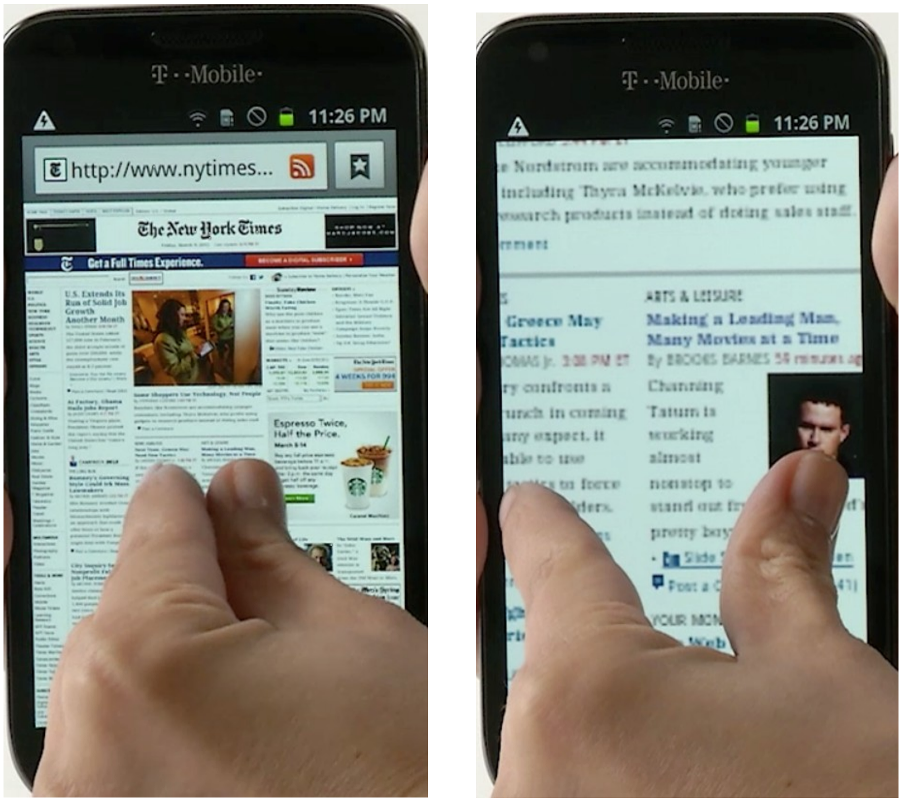
350. **Claim 1 – Element [f] "responding to at least one gesture call, if issued, by scaling the view associated with the event object based on receiving the two or more input points in the form of the user input."** Each of the Accused Products responds to a gesture call, if issued, by calling the view associated with the event object based on receiving the two or more input points in the form of the user input.

351. For example, the Galaxy 10.1 tablet will respond to at least one gesture call by scaling the view (zooming) associated with the MotionEvent object based on receiving two or more input points in the form of the user input, as illustrated below.



(Screenshot of the Samsung Galaxy Tab 10.1 scaling an image.)

352. For example, the Galaxy S 2 phone will respond to at least one gesture call by scaling the view (zooming) by scaling the view associated with the MotionEvent object based on receiving two or more input points in the form of the user input.



353. For example, in the Galaxy 10.1 tablet, the handleMultiTouchInWebView() method calls the WebViewScaleGestureDecetor.onTouchEvent() method to perform the scaling (zoom) operation using the MotionEvent object information, which includes the two or more input points touching the screen. (*See* WebViewScaleGestureDetector.java:189 [SAMNDCA-C000002905].) onTouchEvent() calls setContext(), which records information about the position of the two input points corresponding, for example, to the user's fingers on the screen (WebviewScaleGestureDetector.java:581-630 [SAMNDCA-C000002524 to -C000002525]). As the user moves his fingers relative to one another—as in, for example, a pinching or de-pinching gesture—the handleScale() method of the ZoomManager class calls the WebviewScaleGestureDetector's getScaleFactor() method to calculate the scale factor based on the ratio of the current distance between the fingers and the previous distance between them (as of the last time the touch screen was polled for input). (ZoomManager.java:1323 [SAMNDCA-C000002410]; WebScaleGestureDetector.java:763-768 [SAMNDCA-C000002528].)

handleScale() then calls setZoomScale(), which uses the calculated scale factor to scale the WebView and all of its child views. ZoomManager.java:1372 [SAMNDCA-C000002411]; ZoomManager.java:851-949 [SAMNDCA-C000002399 to -C000002402].)

354. Based on my inspection of Samsung source code for each major release of Android running on the Accused Products (Android 2.1, 2.2, 2.3, and 3.1), I have determined that each Accused Product includes similar computer code that responds to at least one gesture call, if issued, by scaling the view associated with the event object based on receiving the two or more input points in the form of the user input. The claim chart in Exhibit 17 identifies analogous code that satisfies this element in Android 2.1, 2.2, and 2.3.

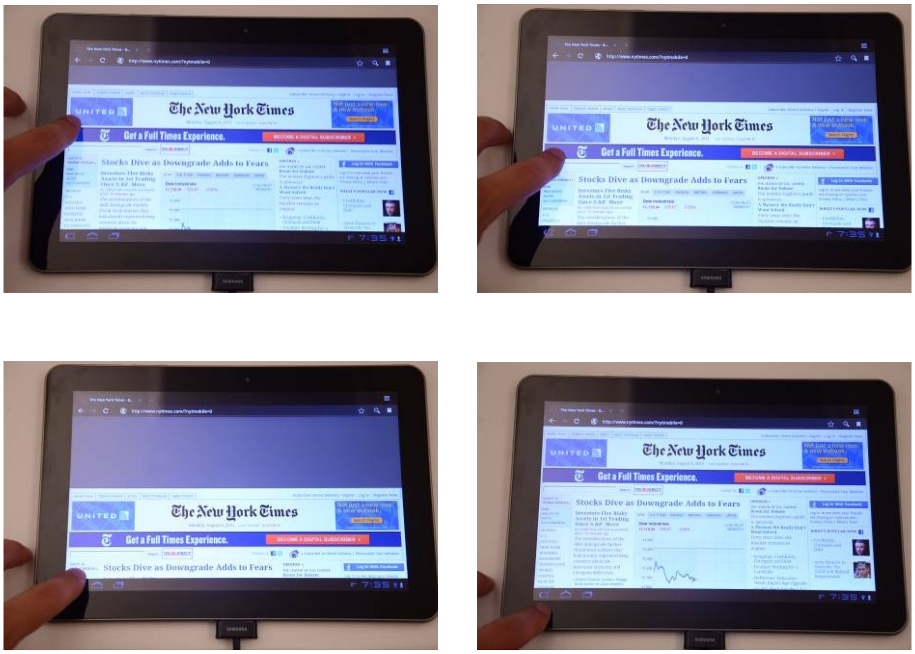
355. To the extent that this limitation is not met literally, in my opinion it is met under the doctrine of equivalents because each of the Accused Products perform steps insubstantially different from responding to at least one gesture call, if issued, by scaling the view associated with the event object based on receiving the two or more input points in the form of the user input, and accomplishes the same function in the same way to achieve the same result.

356. **Claim 2.** Claim 2 recites:

> The method as in claim 1, further comprising:
>
> rubberbanding a scrolling region displayed within the window by a predetermined maximum displacement when the scrolling region exceeds a window edge based on the scroll.

357. The following Accused Products infringe claim 1 and also rubberband a scrolling region displayed within the window by a predetermined maximum displacement when the scrolling region exceeds a window edge based on the scroll: Exhibit 4G; Galaxy Ace; Galaxy S II (i9100, AT&T, and Epic 4G Touch variants); Galaxy Tab 7.0; Galaxy Tab 10.1; and Gravity Smart.

358. For example, the Samsung Galaxy Tab 10.1 rubberbands a scrolling region displayed within the window by a predetermined maximum displacement when the scrolling region exceeds a window edge based on the scroll, as illustrated below.



(Screenshots of the Samsung Galaxy Tab 10.1 rubberbanding upon dragging an image.)

359. For example, the predetermined maximum displacement is defined in the Galaxy Tab 10.1 tablet source code to be 1/6 the height and 1/6 the width of the screen for a fling (i.e., a quick, flicking motion of the user's finger on the screen that causes the view to scroll a predetermined distance without further user input). The handleTouchEventCommon() method calls doFling(). (*See* WebView.java:7272-7821 [SAMNDCA-C000002919 to -C000002931] (call done at 7772).) In the doFling() method, if the isElasticEffectEnabled() method returns a true value (i.e., if the device is configured to "rubberband") the variables "overX" and "overY" are set to 1/6 the screen width and 1/6 the screen height, respectively. (*See* WebView.java:9236-9376 [SAMNDCA-C000002932-2935] (particularly lines 9350-9361).) The overX and overY variables are then passed to the Overscroller.fling() method, and they set the maximum amount for rubberbanding displacement. (*See id.*)

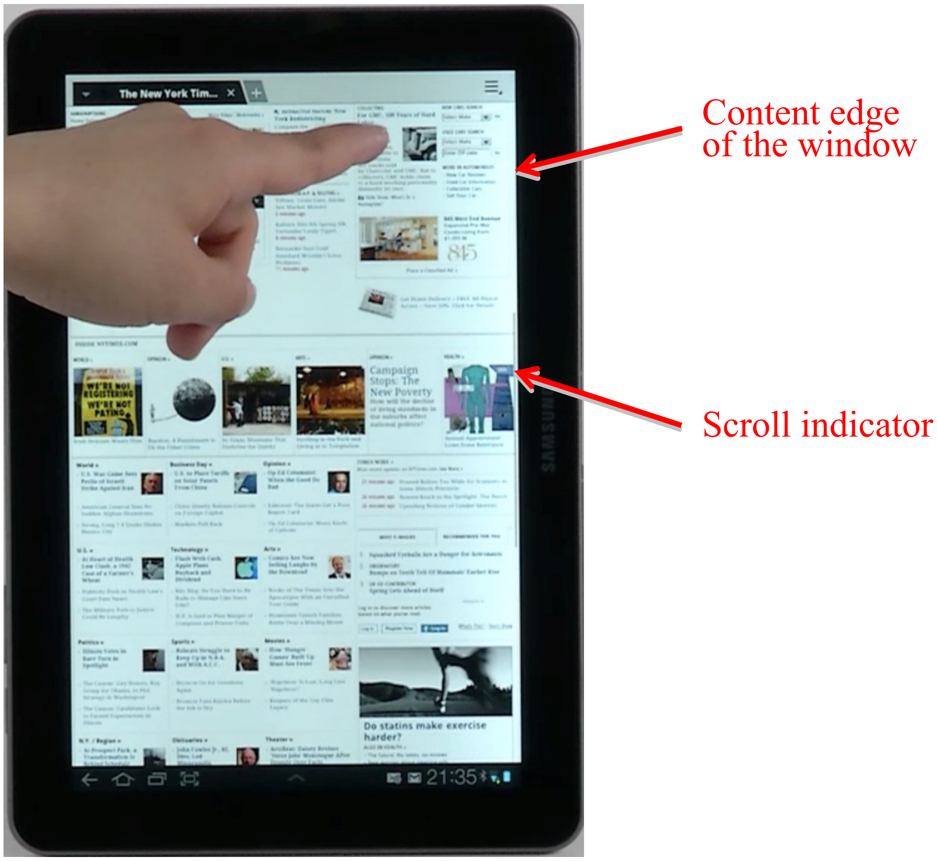
360. To the extent that this limitation is not met literally, in my opinion it is met under the doctrine of equivalents because each of the Accused Products perform steps insubstantially different from rubberbanding a scrolling region displayed within the window by a predetermined maximum displacement when the scrolling region exceeds a window edge based on the scroll, and accomplishes the same function in the same way to achieve the same result.

361. **Claim 3.** Claim 3 recites:

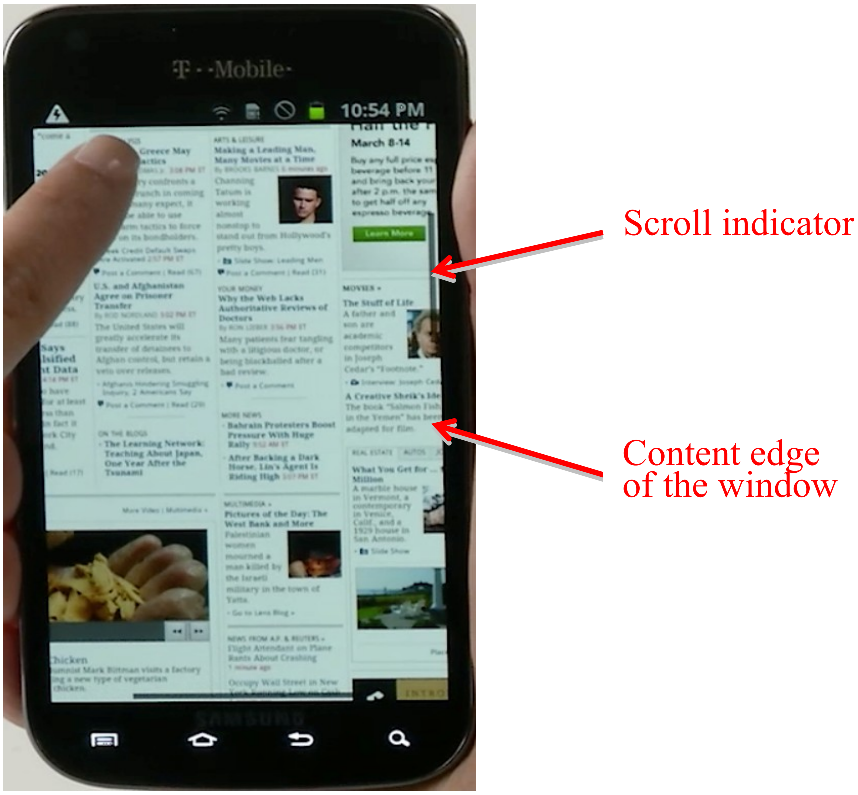
> The method as in claim 1, further comprising:
>
> attaching scroll indicators to a content edge of the window.

362. The following Accused Products attach scroll indicators to a content edge of the window: Acclaim, Captivate, Continuum, Droid Charge, Epic 4G, Exhibit 4G, Fascinate, Galaxy Ace, Galaxy Prevail, Galaxy S (i9000), Galaxy S 4G, Galaxy S II (including its T-Mobile, AT&T, Epic 4G Touch and AT&T Skyrocket versions), Galaxy S Showcase (i500), Galaxy Tab 7.0, Galaxy Tab 10.1, Gem, Gravity Smart, Indulge, Infuse 4G, Intercept, Mesmerize, Nexus S, Nexus S 4G, Replenish, Sidekick, Transform, and Vibrant. The videos in Exhibits 18 through 21 show the Galaxy Tab 10.1, the Galaxy S II, the Vibrant, and the Captivate attaching scroll indicators to a content edge of the window.

363. For example, the Galaxy Tab 10.1 attaches scroll indicators to the content edge of the window, as illustrated below.



364. For example, the Galaxy S II attaches scroll indicators to the content edge of the window, as illustrated below.



365. To the extent that this limitation is not met literally, in my opinion it is met under the doctrine of equivalents because each of the Accused Products perform steps insubstantially different from attaching scroll indicators to a content edge of the window, and accomplishes the same function in the same way to achieve the same result.

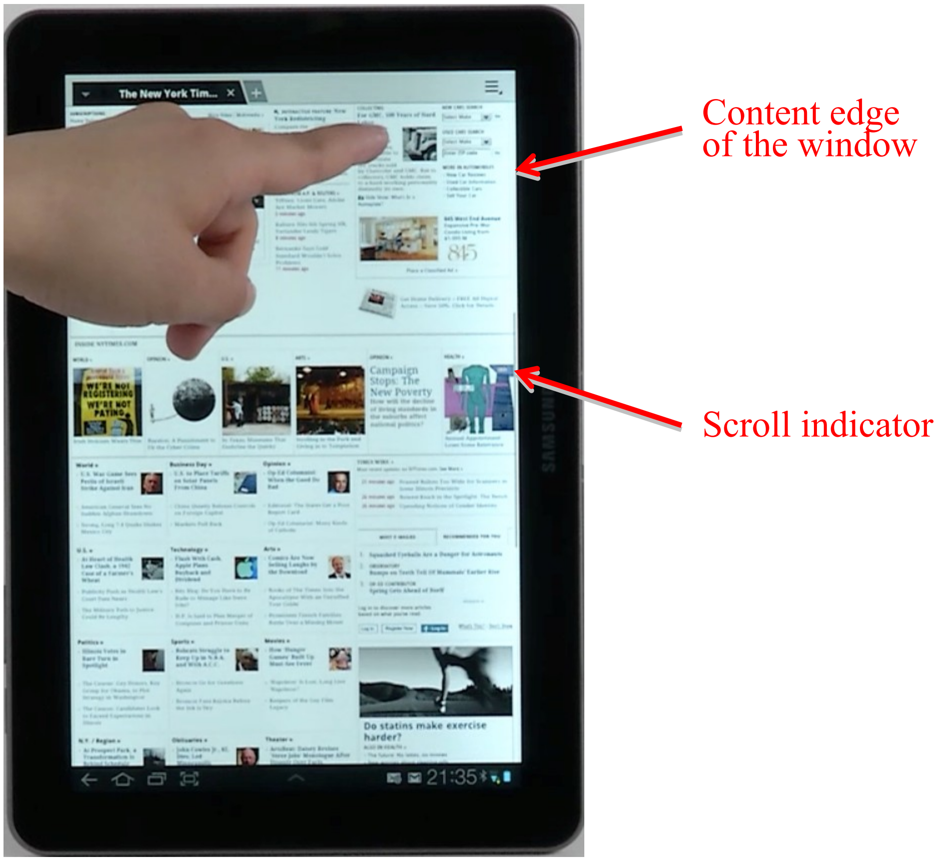
366. **Claim 4.** Claim 4 of the '915 Patent recites:

> The method as in claim 1, further comprising:
>
> attaching scroll indicators to the window edge.

367. The following Accused Products attach scroll indicators to the window edge: Acclaim, Captivate, Continuum, Droid Charge, Epic 4G, Exhibit 4G, Fascinate, Galaxy Ace, Galaxy Prevail, Galaxy S (i9000), Galaxy S 4G, Galaxy S II, (including its T-Mobile, AT&T,

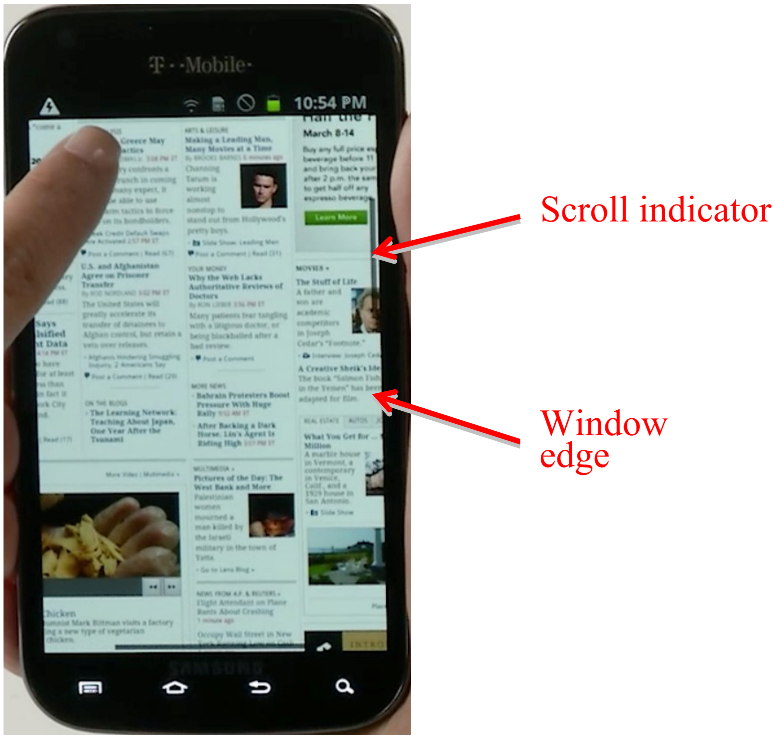
Epic 4G Touch and AT&T Skyrocket versions), Galaxy S Showcase (i500), Galaxy Tab 7.0, Galaxy Tab 10.1, Gem, Gravity Smart, Indulge, Infuse 4G, Intercept, Mesmerize, Nexus S, Nexus S 4G, Replenish, Sidekick, Transform, and Vibrant. The videos in Exhibits 18 through 21 show the Galaxy Tab 10.1, the Galaxy S II, the Vibrant, and the Captivate attaching scroll indicators to the window edge.

368. For example, the Galaxy Tab 10.1 attaches scroll indicators to the window edge, as illustrated below:



(Screenshot of the Samsung Galaxy Tab 10.1 attaching a scroll indicator to the window edge.)

369. For example, the Galaxy S II attaches scroll indicators to the window edge, as illustrated below.



370. To the extent that this limitation is not met literally, in my opinion it is met under the doctrine of equivalents because each of the Accused Products perform steps insubstantially different from attaching scroll indicators to the window edge, and accomplishes the same function in the same way to achieve the same result.

371. **Claim 5.** Claim 5 of the ’915 Patent recites:

> The method as in claim 1, wherein determining whether the event object invokes a scroll or gesture operation is based on receiving a drag user input for a certain time period.

372. Each of the Accused Products determines whether the event object invokes a scroll or gesture operation based on receiving a drag user input for a certain time period.

373. For example, the Galaxy Tab 10.1 tablet determines whether the event object invokes the scroll operation based on receiving a drag user input for a certain time period. The handleTouchEventCommon() invokes the fling operation based on the user scrolling within a certain period of time. (*See* WebView.java:7758 [SAMDNCA00002919 to –C000002931].)

374. Based on my inspection of Samsung source code for each major release of Android running on the Accused Products (Android 2.1, 2.2, 2.3, and 3.1), I have determined that each Accused Product includes similar computer code that determines whether the event object invokes a scroll or gesture operation is based on receiving a drag user input for a certain time period. The claim chart in Exhibit 17 identifies analogous code that satisfies this element in Android 2.1, 2.2, and 2.3.

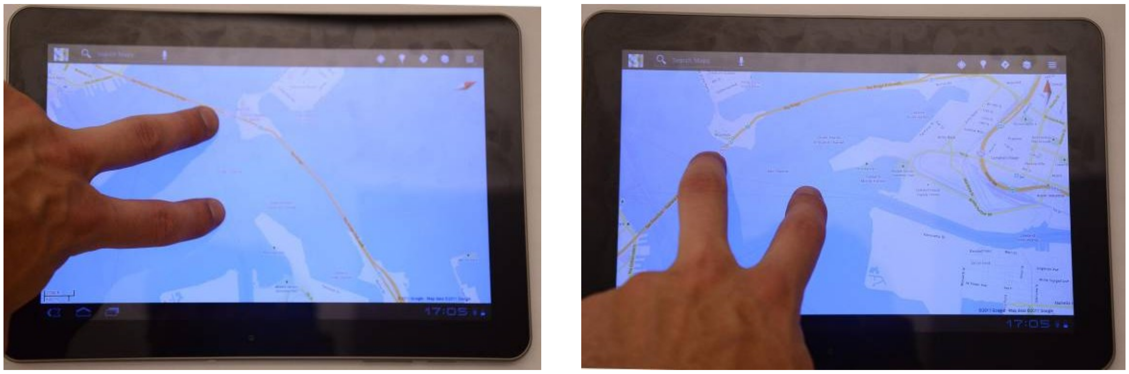
375. To the extent that this limitation is not met literally, in my opinion it is met under the doctrine of equivalents because each of the Accused Products perform steps insubstantially different from invoking a scroll or gesture operation is based on receiving a drag user input for a certain time period, and accomplishes the same function in the same way to achieve the same result.

376. **Claim 6.** Claim 6 recites:

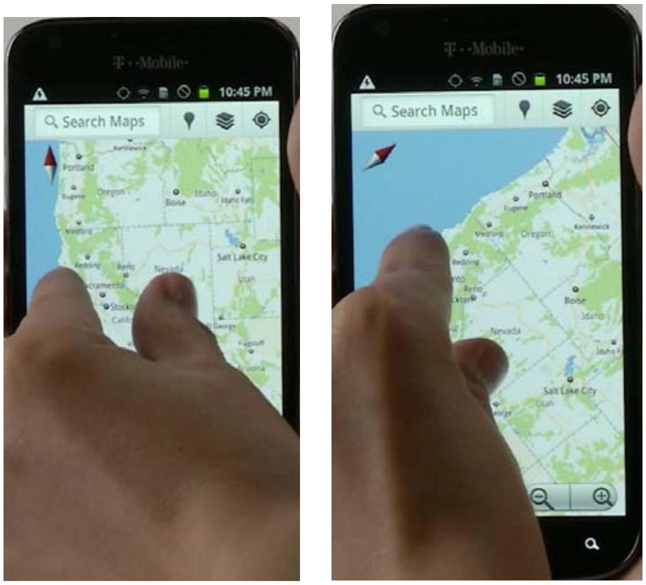
> The method as in claim 1, further comprising:
>
> responding to at least one gesture call, if issued, by rotating a view associated with the event object based on receiving a plurality of input points in the form of the user input.

377. The following Accused Products respond to at least one gesture call, if issued, by rotating a view associated with the event object based on receiving a plurality of input points in the form of the user input: Galaxy S II (including its Epic 4G Touch and AT&T Skyrocket versions), Galaxy Tab 10.1, Nexus S, and Nexus S 4G. A video of the Galaxy Tab 10.1 performing the limitations of this claim is attached as Exhibit 22, and a video of the Galaxy S II performing the limitations of this claim is attached as Exhibit 23.

378. For example, the Galaxy Tab 10.1 responds to at least one gesture call, if issued, by rotating a view associated with the event object based on receiving a plurality of input points (plurality of fingers) in the form of the user input.



379. For example, the Galaxy S II responds to at least one gesture call, if issued, by rotating a view associated with the event object based on receiving a plurality of input points (plurality of fingers) in the form of the user input.



380. To the extent that this limitation is not met literally, in my opinion it is met under the doctrine of equivalents because each of the Accused Products perform steps insubstantially different from responding to at least one gesture call, if issued, by rotating a view associated with the event object based on receiving a plurality of input points in the form of the user input, and accomplishes the same function in the same way to achieve the same result.

381. **Claim 7.** Claim 7 recites:

> The method as in claim 1, wherein the device is one of: a data processing device, a portable device, a portable data processing device, a multi touch device, a multi touch portable device, a wireless device, and a cell phone.

382. Each of the Accused Products is a portable data processing device, a multi touch device, a multi touch portable device, a wireless device, and a cell phone.

383. To the extent that this limitation is not met literally, in my opinion it is met under the doctrine of equivalents because each of the Accused Products is insubstantially different from a multi touch portable device, and accomplishes the same function in the same way to achieve the same result.

384. **Claim 8.** Claim 8 recites:

> A machine readable storage medium storing executable program instructions which when executed cause a data processing system to perform a method comprising:
>
> [a] receiving a user input, the user input is one or more input points applied to a touch-sensitive display that is integrated with the data processing system;
>
> [b] creating an event object in response to the user input;
>
> [c] determining whether the event object invokes a scroll or gesture operation by distinguishing between a single input point applied to the touch-sensitive display that is interpreted as the scroll operation and two or more input points applied to the touch-sensitive display that are interpreted as the gesture operation
>
> [d] issuing at least one scroll or gesture call based on invoking the scroll or gesture operation;
>
> [e] responding to at least one scroll call, if issued, by scrolling a window having a view associated with the event object;
>
> [f] responding to at least one gesture call, if issued, by scaling the view associated with the event object based on receiving the two or more input points in the form of the user input.

385. **Claim 8 – Preamble "A machine readable storage medium storing executable program instructions which when executed cause a data processing system to perform a method comprising."** Each of the Accused Products is either a smartphone or tablet running a version of the Android operating system, which includes a data processing system. Each '915 Accused Product includes a computer readable storage medium storing executable program

instructions which when executed cause the data processing system to perform the method described in claim 8.

386. **Claim 8 – Element [a] "receiving a user input, the user input is one or more input points applied to a touch-sensitive display that is integrated with the data processing system."** In my opinion, each of the Accused Products includes a machine readable storage medium storing executable program instructions which when executed cause a data processing system to receive a user input, where the user input is one or more input points applied to a touch-sensitive display that is integrated with the data processing system, for the same reasons as explained with respect to claim 1, above.

387. **Claim 8 – Element [b] "creating an event object in response to the user input."** In my opinion, each of the Accused Products includes a machine readable storage medium storing executable program instructions which when executed cause a data processing system to create an event object in response to the user input, for the same reasons as explained with respect to claim 1.

388. **Claim 8 – Element [c] "determining whether the event object invokes a scroll or gesture operation by distinguishing between a single input point applied to the touch-sensitive display that is interpreted as the scroll operation and two or more input points applied to the touch-sensitive display that are interpreted as the gesture operation."** In my opinion, each of the Accused Products includes a machine readable storage medium storing executable program instructions which when executed cause a data processing system to determine whether the event object invokes a scroll or gesture operation by distinguishing between a single input point applied to the touch-sensitive display that is interpreted as the scroll operation and two or more input points applied to the touch-sensitive display that are interpreted as the gesture operation, for the same reasons as explained with respect to claim 1.

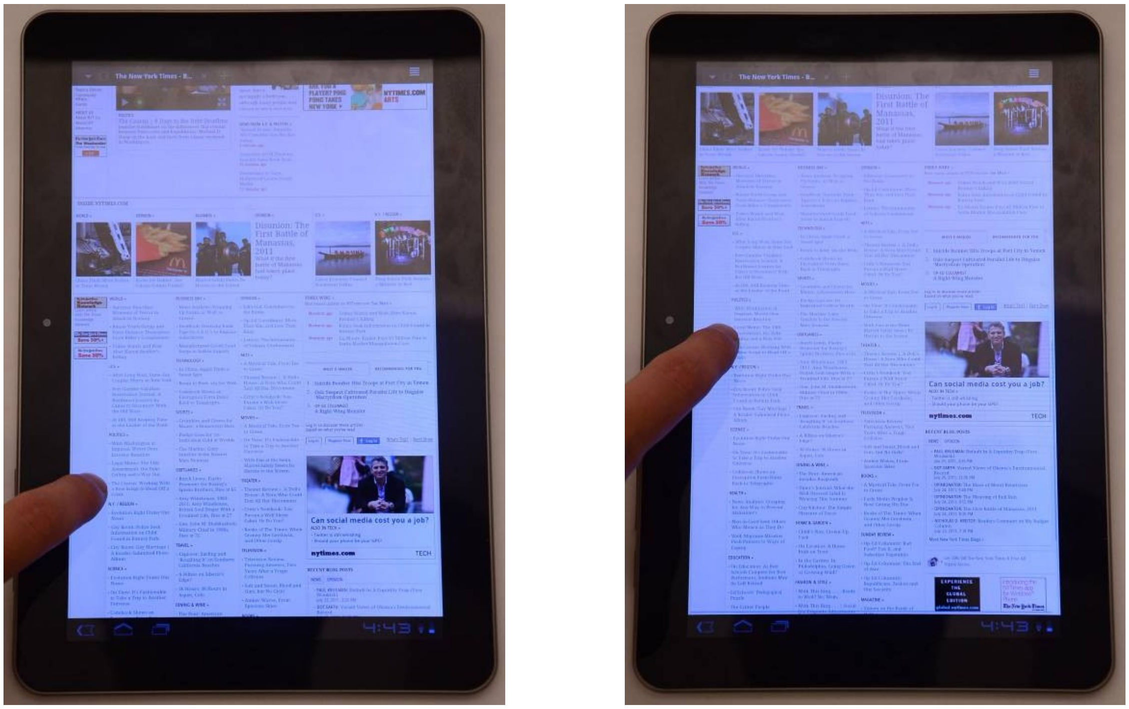
389. **Claim 8 – Element [d] "issuing at least one scroll or gesture call based on invoking the scroll or gesture operation."** In my opinion, each of the Accused Products includes a machine readable storage medium storing executable program instructions which when

executed cause a data processing system to issue at least one scroll or gesture call based on invoking the scroll or gesture operation, for the same reasons as explained with respect to claim 1.

390. **Claim 8 – Element [e] "responding to at least one scroll call, if issued, by scrolling a window having a view associated with the event object."** In my opinion, each of the Accused Products includes a machine readable storage medium storing executable program instructions which when executed cause a data processing system to respond to at least one scroll call, if issued, by scrolling a window having a view associated with the event object.

391. Each of the Accused Products responds to a scroll call, if issued, by scrolling a window having a view associated with the event object based on an amount of a scroll with the scroll stopped at a predetermined position in relation to the user input.

392. For example, the Galaxy 10.1 tablet will respond to at least one scroll call by scrolling a window having a view associated with the MotionEvent object, as illustrated below.


The New York Times
Dissension: The First Battle of Manassas, 2011
Can social media cost you a job?
nytimes.com
TECH
4:43
The New York Times
Dissension: The First Battle of Manassas, 2011
Can social media cost you a job?
nytimes.com
TECH
4:43

(Screenshot of the Samsung Galaxy Tab 10.1 scrolling an image.)

393. For example, the Galaxy S2 phone will respond to at least one scroll call by scrolling a window having a view associated with the MotionEvent object, as illustrated below.

394. For example, in the Galaxy 10.1 tablet, the handleTouchEventCommon() method calls doFling() for a scroll operation. (*See* WebView.java:7272-7821 [SAMNDCA-C000002919 to –C000002931] (call done at 7772).) doFling() then calls the Overscroller.fling() method. (*See* WebView.java:9236-9376 [SAMNDCA-C000002932 to –C000002935].) Overscroller.fling() itself calls two instances of the SplineOverScroller class, each of which is responsible for scrolling in one axis (i.e., one scrolls horizontally and the other scrolls vertically). (*See* OverScroller.java:406-448 [SAMNDCA-C000002945].) The SplineOverScroller class thus maintains state information for the fling. (*See id.*)

395. The actual rendering of the fling occurs subsequently as part of the drawing cycle. At the end of an event processing cycle, the method computeScroll() is called to compute which part of the view should be rendered to the user. (*See* WebView.java:3568-3654 [SAMNDCA-C000002958 to –C000002959]. The computeScroll() method uses the SplineOverScroller class to extract the state information for the fling. (*See id.*) Afterwards, it calls WebView.overScrollBy() to scroll the content—this method calculates maximums for the distance the user can scroll beyond the edge of the content and whether content should be fixed to a particular axis. (*See id.*; *see also* View.java:11663-11715 [SAMNDCA-C000002960 to –

C000002961] (WebView.overScrollBy()).) onOverScrollBy() itself calls onOverScroller() to ensure the intended scroll coordinates are valid and then calls View.scrollTo() to invoke the scroll operation. (*See* View.java:11663-11715 [SAMNDCA-C000002960 to –C000002961]; WebView.java:3130-3162 [SAMDNCA-2962].) View.scrollTo() scrolls the window (setting mScrollX and mScrollY) based on the amount of a scroll with the scroll stopped at a "predetermined position in relation to the user input." (*See* WebView.java:3130-3162 [SAMDNCA-2962].)

396. Based on my inspection of Samsung source code for each major release of Android running on the Accused Products (Android 2.1, 2.2, 2.3, and 3.1), I have determined that each Accused Product includes similar computer code that responds to at least one scroll call by scrolling a window having a view associated with the MotionEvent object.

397. To the extent that this limitation is not met literally, in my opinion it is met under the doctrine of equivalents because each of the Accused Products perform steps insubstantially different from responding to at least one scroll call, if issued, by scrolling a window having a view associated with the event object, and accomplishes the same function in the same way to achieve the same result.

398. **Claim 8 – Element [f] "responding to at least one gesture call, if issued, by scaling the view associated with the event object based on receiving the two or more input points in the form of the user input."** In my opinion, each of the Accused Products includes a machine readable storage medium storing executable program instructions which when executed cause a data processing system to respond to at least one gesture call, if issued, by scaling the view associated with the event object based on receiving the two or more input points in the form of the user input, for the same reasons as explained with respect to claim 1.

399. **Claim 9.** Claim 9 recites:

> The medium as in claim 8, further comprising:
>
> rubberbanding a scrolling region displayed within the window by a predetermined maximum displacement when the scrolled region exceeds a window edge based on the scroll.

400. Claim 9 claims the media as in claim 8 and adds a limitation analogous to dependent claim 2 requiring "rubberbanding." Accordingly, the same Accused Products discussed in connection with claim 2 infringe claim 8 for the reasons discussed in connection with claim 2.

401. **Claim 10.** Claim 10 recites:

> The medium as in claim 8, further comprising:
>
> attaching scroll indicators to a content edge of the view.

402. Claim 10 claims the media as in claim 8 and adds a limitation analogous to dependent claim 3 requiring "attaching scroll indicators to a content edge of the view." Accordingly, the same Accused Products discussed in connection with claim 3 infringe claim 9 for the reasons discussed in connection with claim 3.

403. **Claim 11.** Claim 11 recites:

> The medium as in claim 8, further comprising:
>
> attaching scroll indicators to a window edge of the view.

404. Claim 11 claims the media as in claim 8 and adds a limitation analogous to dependent claim 4 requiring "attaching scroll indicators to a window edge of the view." Accordingly, the Accused Products discussed in connection with claim 4 infringe claim 10 for the reasons discussed in connection with claim 4.

405. **Claim 12.** Claim 12 recites:

> The medium as in claim 8, wherein determining whether the event object invokes a scroll or gesture operation is based on receiving a drag user input for a certain time period.

406. Claim 12 claims the media as in claim 8 and adds a limitation analogous to dependent claim 5 wherein "determining whether the event object invokes a scroll or gesture operation is based on receiving a drag user input for a certain time period." Accordingly, the Accused Products discussed in connection with claim 5 infringe claim 12 for the reasons discussed in connection with claim 5.

407. **Claim 13.** Claim 13 recites:

> The medium as in claim 8, further comprising:

> Responding to at least one gesture call, if issued, by rotating a view associated with the event object based on receiving a plurality of input points in the form of the user input.

408. Claim 13 claims the media as in claim 8 and adds a limitation analogous to dependent claim 6 further comprising "responding to at least one gesture call, if issued, by rotating a view associated with the event object based on receiving a plurality of input points in the form of the user input." Accordingly, the Accused Products discussed in connection with claim 6 infringe claim 13 for the reasons discussed in connection with claim 6.

409. **Claim 14.** Claim 14 recites:

> The medium as in claim 8, wherein the data processing system is one of: a data processing device, a portable device, a portable data processing device, a multi touch device, a multi touch portable device, a wireless device, and a cell phone.

410. Claim 14 claims the media as in claim 8 and adds a limitation analogous to dependent claim 7 wherein the data processing system may be a "multi touch portable device." Accordingly, the Accused Products discussed in connection with claim 7 infringe claim 14 for the reasons discussed in connection with claim 7.

411. **Claim 15.** Claim 15 recites:

> An apparatus, comprising:
>
> [a] means for receiving, through a hardware device, a user input on a touch-sensitive display of the apparatus, the user input is one or more input points applied to the touch-sensitive display that is integrated with the apparatus;
>
> [b] means for creating an event object in response to the user input;
>
> [c] means for determining whether the event object invokes a scroll or gesture operation by distinguishing between a single input point applied to the touch-sensitive display that is interpreted as the scroll operation and two or more input points applied to the touch-sensitive display that are interpreted as the gesture operation;
>
> [d] means for issuing at least one scroll or gesture call based on invoking the scroll or gesture operation;
>
> [e] means for responding to at least one scroll call, if issued, by scrolling a window having a view associated with the event object; and
>
> [f] means for responding to at least one gesture call, if issued, by scaling the view associated with the event object based on receiving

the two or more input points in the form of the user input.

412. **Claim 15 – Preamble "An apparatus, comprising:"** Claim 15 is directed to an apparatus. Each of the Accused Products is an apparatus.

413. **Claim 15 – element [a] "means for receiving, through a hardware device, a user input on a touch-sensitive display of the apparatus, the user input is one or more input points applied to the touch-sensitive display that is integrated with the apparatus."** I have been informed that the limitation "means for receiving, through a hardware device, a user input on a touch-sensitive display of the apparatus" is in "means plus function" form and is governed by section 112.6. The function is receiving, through a hardware device, a user input on a touch-sensitive display of the apparatus. The corresponding structure is one or more special or general purpose processors programmed with special-purpose software to execute an algorithm, the special-purpose software including computer instructions for receiving, through a hardware device, a user input on a touch-sensitive display of the apparatus.

414. As discussed above, each of the Accused Products includes a processor programmed to execute an algorithm to receive, through a touch screen, a user input. The Accused Products perform the claimed function in manner equivalent to the manner described in the specification. *See*, *e.g.*, '915 Patent at 1:59-67, 2:37-42, 4:29-6:32, 6:33-36, 12:19-13:40, 21:10-56, 22:5-16, 22:42-48; FIGS. 1, 13, 14, 32, and 33A-C.

415. Claim 15 element [a] also requires that the user input is one or more input points applied to the touch-sensitive display that is integrated with the apparatus. As explained above, each of the Accused Products receives user input in the form of one or more inputs points applied to the touch-sensitive display integrated with the apparatus.

416. **Claim 15 – element [b] "means for creating an event object in response to the user input."** I have been informed that this limitation is in "means plus function" form and is governed by section 112.6. The function is creating an event object in response to the user input. The corresponding structure is one or more special or general purpose processors programmed

with special-purpose software to execute an algorithm, the special-purpose software including computer instructions for creating an event object in response to the user input.

417. As discussed above, each of the Accused Products includes a processor programmed to execute an algorithm for creating an event object in response to the user input. The Accused Products perform the claimed function in manner equivalent to the manner described in the specification. *See*, *e.g.*, '915 Patent at 1:59-67, 2:37-42, 4:29-6:37, 12:30-32, 21:10-56, 22:5-16, 22:42-48; FIGS. 1, 13, 32, and 33A-C.

418. **Claim 15 – element [c] "means for determining whether the event object invokes a scroll or gesture operation by distinguishing between a single input point applied to the touch-sensitive display that is interpreted as the scroll operation and two or more input points applied to the touch-sensitive display that are interpreted as the gesture operation."** I have been informed that this limitation is in "means plus function" form and is governed by section 112.6. The function is determining whether the event object invokes a scroll or gesture operation by distinguishing between a single input point applied to the touch-sensitive display that is interpreted as the scroll operation and two or more input points applied to the touch-sensitive display that are interpreted as the gesture operation. The corresponding structure is one or more special or general purpose processors programmed with special-purpose software to execute an algorithm, the special-purpose software including computer instructions for determining whether the event object invokes a scroll or gesture operation by distinguishing between a single input point applied to the touch-sensitive display that is interpreted as the scroll operation and two or more input points applied to the touch-sensitive display that are interpreted as the gesture operation.

419. As discussed above, each of the Accused Products includes a processor programmed to execute an algorithm for determining whether the event object invokes a scroll or gesture operation by distinguishing between a single input point applied to the touch-sensitive display that is interpreted as the scroll operation and two or more input points applied to the touch-sensitive display that are interpreted as the gesture operation. The Accused Products perform the claimed function in manner equivalent to the manner described in the specification.

See, *e.g.*, '915 Patent at 1:59-67, 2:22-29, 2:37-42, 4:29-6:32, 6:37-48, 6:57-60, 9:61-11:13, 12:19-14:40, 21:10-56, 22:5-16, 22:42-48; FIGS. 1, 7-10, 13, 14, 32, and 33A-C.

420. **Claim 15 – element [d] "means for issuing at least one scroll or gesture call based on invoking the scroll or gesture operation."** I have been informed that this limitation is in "means plus function" form and is governed by section 112.6. The function is issuing at least one scroll or gesture call based on invoking the scroll or gesture operation. The corresponding structure is one or more special or general purpose processors programmed with special-purpose software to execute an algorithm, the special-purpose software including computer instructions for issuing at least one scroll or gesture call based on invoking the scroll or gesture operation.

421. As discussed above, each of the Accused Products includes a processor programmed to execute an algorithm for issuing at least one scroll or gesture call based on invoking the scroll or gesture operation. The Accused Products perform the claimed function in manner equivalent to the manner described in the specification. *See*, *e.g.*, '915 Patent at 1:59-67, 2:22-29, 2:37-42, 4:29-6:32, 6:46-48, 9:61-11:13, 12:19-28, 12:34-37, 13:21-50, 21:10-56, 22:5-16, 22:42-48; FIGS. 1, 7-10, 13, 14, 32, and 33A-C.

422. **Claim 15 – element [e] "means for responding to at least one scroll call, if issued, by scrolling a window having a view associated with the event object."** I have been informed that this limitation is in "means plus function" form and is governed by section 112.6. The function is responding to at least one scroll call, if issued, by scrolling a window having a view associated with the event object. The corresponding structure is a display coupled with one or more special or general purpose processors programmed with special-purpose software to execute an algorithm, the special-purpose software including computer instructions for responding to at least one scroll call, if issued, by scrolling a window having a view associated with the event object.

423. As discussed above, each of the Accused Products includes a display and a processor programmed to execute an algorithm for responding to at least one scroll call, if issued, by scrolling a window having a view associated with the event object. The Accused Products perform the claimed function in manner equivalent to the manner described in the specification.

See, *e.g.*, '915 Patent at 1:59-67, 2:37-42, 4:29-6:32, 6:46-56, 8:4-25, 9:61-11:13, 18:25-19:61, 20:50-21:56, 22:5-16, 22:42-48; FIGS. 1, 4, 7-10, 28, 29, 30A-B, 32, and 33A-C.

424. **Claim 15 – element [f] "means for responding to at least one gesture call, if issued, by scaling the view associated with the event object based on receiving the two or more input points in the form of the user input."** I have been informed that this limitation is in "means plus function" form and is governed by section 112.6. The function is responding to at least one gesture call, if issued, by scaling the view associated with the event object based on receiving the two or more input points in the form of the user input. The corresponding structure is a display coupled with one or more special or general purpose processors programmed with special-purpose software to execute an algorithm, the special-purpose software including computer instructions for responding to at least one gesture call, if issued, by scaling the view associated with the event object based on receiving the two or more input points in the form of the user input.

425. As discussed above, each of the Accused Products includes a display and a processor programmed to execute an algorithm for responding to at least one gesture call, if issued, by scaling the view associated with the event object based on receiving the two or more input points in the form of the user input. The Accused Products perform the claimed function in manner equivalent to the manner described in the specification. *See*, *e.g.*, '915 Patent at 1:59-67, 2:22-29, 2:37-42, 4:29-6:32, 6:57-60, 8:4-25, 12:19-14:40, 18:25-19:61, 20:50-21:56, 22:5-16, 22:42-48; FIGS. 1, 4, 13-15, 16A-C, 28-29, 30A-B, 32, and 33A-C.

426. In summary, in my opinion each of the Accused Products is an apparatus that practices Claim 15. To the extent that this claim is not met literally, in my opinion it is met under the doctrine of equivalents because each of the Accused Products accomplishes the same function in the same way to achieve the same result.

427. **Claim 16.** Claim 16 recites:

> The apparatus as in claim 15, further comprising: means for rubberbanding a scrolling region displayed within the window by a predetermined maximum displacement when the scrolling region exceeds a window edge based on the scroll.

428. Claim 16 claims the apparatus as in claim 15 and adds a limitation analogous to dependent claim 2 further comprising "means for rubberbanding a scrolling region displayed within the window by a predetermined maximum displacement when the scrolling region exceeds a window edge based on the scroll." Accordingly, the Accused Products discussed in connection with claim 2 infringe claim 16 for the reasons discussed in connection with claim 2.

429. I have been informed that this limitation is in "means plus function" form and is governed by section 112.6. The function is rubberbanding a scrolling region displayed within the window by a predetermined maximum displacement when the scrolling region exceeds a window edge based on the scroll. The corresponding structure is a display coupled with one or more special or general purpose processors programmed with special-purpose software to execute an algorithm, the special-purpose software including computer instructions for rubberbanding a scrolling region displayed within the window by a predetermined maximum displacement when the scrolling region exceeds a window edge based on the scroll.

430. As discussed above, each of the above-listed products includes a display and a processor programmed to execute an algorithm for rubberbanding a scrolling region displayed within the window by a predetermined maximum displacement when the scrolling region exceeds a window edge based on the scroll. The above-listed products perform the claimed function in manner equivalent to the manner described in the specification. *See*, *e.g.*, '915 Patent at 1:59-67, 2:11-21, 2:37-42, 4:29-6:32, 7:46-8:3-25, 8:61-9:60, 18:25-19:61, 20:50-21:56, 22:5-16, 22:21-26, 22:42-48, 22:53-58; FIGS. 1, 3, 4, 6A-D, 28, 29, 30A-B, 32, and 33A-C.

431. In summary, in my opinion each of the above-listed products is an apparatus that practices Claim 16. To the extent that this claim is not met literally, in my opinion it is met under the doctrine of equivalents because each of the above-listed products accomplishes the same function in the same way to achieve the same result.

432. **Claim 17.** Claim 17 recites:

> The apparatus as in claim 15, further comprising: means for attaching scroll indicators to a content edge of the window.

433. Claim 17 claims the apparatus in claim 15 and adds a limitation analogous to dependent claim 3 further comprising "means for attaching scroll in indicators to a content edge of the window." Accordingly, the Accused Products discussed in connection with claim 3 infringe claim 17 for the reasons discussed in connection with claim 3.

434. I have been informed that this limitation is in "means plus function" form and is governed by section 112.6. The function is attaching scroll indicators to a content edge of the window. The corresponding structure is a display coupled with one or more special or general purpose processors programmed with special-purpose software to execute an algorithm, the special-purpose software including computer instructions for attaching scroll indicators to a content edge of the window.

435. As discussed above, each of the above-listed products includes a display and a processor programmed to execute an algorithm for attaching scroll indicators to a content edge of the window. The above-listed products perform the claimed function in manner equivalent to the manner described in the specification. *See*, *e.g.*, '915 Patent at 1:59-67, 2:11-21, 2:37-42, 4:29-6:32, 7:46-8:3-25, 8:61-9:60, 18:25-19:61, 20:50-21:56, 22:5-16, 22:21-26, 22:42-48, 22:53-58; FIGS. 1, 3, 4, 6A-D, 28, 29, 30A-B, 32, and 33A-C.

436. In summary, in my opinion each of the above-listed products is an apparatus that practices Claim 17. To the extent that this claim is not met literally, in my opinion it is met under the doctrine of equivalents because each of the above-listed products accomplishes the same function in the same way to achieve the same result.

437. **Claim 18.** Claim 18 recites:

> The apparatus as in claim 15, further comprising: means for attaching scroll indicators to the window edge.

438. Claim 18 claims the apparatus in claim 15 and adds a limitation analogous to dependent claim 4 further comprising "means for attaching scroll indicators to the window edge." Accordingly, the Accused Products discussed in connection with claim 4 infringe claim 18 for the reasons discussed in connection with claim 4.

439. I have been informed that this limitation is in "means plus function" form and is governed by section 112.6. The function is attaching scroll indicators to the window edge. The corresponding structure is a display coupled with one or more special or general purpose processors programmed with special-purpose software to execute an algorithm, the special-purpose software including computer instructions for attaching scroll indicators to the window edge.

440. As discussed above, each of the above-listed products includes a display and a processor programmed to execute an algorithm for attaching scroll indicators to the window edge. The above-listed products perform the claimed function in manner equivalent to the manner described in the specification. *See*, *e.g.*, '915 Patent at 1:59-67, 2:11-21, 2:37-42, 4:29-6:32, 7:46-8:3-25, 8:61-9:60, 18:25-19:61, 20:50-21:56, 22:5-16, 22:21-26, 22:42-48, 22:53-58; FIGS. 1, 3, 4, 6A-D, 28, 29, 30A-B, 32, and 33A-C.

441. In summary, in my opinion each of the above-listed products is an apparatus that practices Claim 18. To the extent that this claim is not met literally, in my opinion it is met under the doctrine of equivalents because each of the above-listed products accomplishes the same function in the same way to achieve the same result.

442. **Claim 19.** Claim 19 recites:

> The apparatus as in claim 15, wherein determining whether the event object invokes a scroll or gesture operation is based on receiving a drag user input for a certain time period.

443. Claim 19 claims the apparatus in claim 15 and adds a limitation analogous to dependent claim 5 wherein "determining whether the event object invokes a scroll or gesture operation is based on receiving a drag user input for a certain time period." Accordingly, the Accused Products discussed in connection with claim 5 infringe claim 19 for the reasons discussed in connection with claim 5. To the extent that this claim is not met literally, in my opinion it is met under the doctrine of equivalents because each of the Accused Products accomplishes the same function in the same way to achieve the same result.

444. **Claim 20.** Claim 20 recites:

> The apparatus as in claim 15, further comprising: means for responding to at least one gesture call, if issued, by rotating a view associated with the event object based on receiving a plurality of input points in the form of the user input.

445. Claim 20 claims the apparatus in claim 15 and adds a limitation analogous to dependent claim 6 further comprising "means for responding to at least one gesture call, if issued, by rotating a view associated with the event object based on receiving a plurality of input points in the form of the user input." Accordingly, the Accused Products discussed in connection with claim 6 infringe claim 20 for the reasons discussed in connection with claim 6.

446. I have been informed that this limitation is in "means plus function" form and is governed by section 112.6. The function is responding to at least one gesture call, if issued, by rotating a view associated with the event object based on receiving a plurality of input points in the form of the user input. The corresponding structure is a display coupled with one or more special or general purpose processors programmed with special-purpose software to execute an algorithm, the special-purpose software including computer instructions for responding to at least one gesture call, if issued, by rotating a view associated with the event object based on receiving a plurality of input points in the form of the user input.

447. As discussed above with respect to Claim 13, each of the Accused Products discussed in Claim 13 includes a processor programmed to execute an algorithm for responding to at least one gesture call, if issued, by rotating a view associated with the event object based on receiving a plurality of input points in the form of the user input. These Accused Products perform the claimed function in manner equivalent to the manner described in the specification. *See*, *e.g.*, '915 Patent at 1:59-67, 2:37-42, 4:29-6:37, 12:30-32, 21:10-56, 22:5-16, 22:42-48; FIGS. 1, 13, 32, and 33A-C. To the extent that this claim is not met literally, in my opinion it is met under the doctrine of equivalents because each of the above-listed products accomplishes the same function in the same way to achieve the same result.

448. **Claim 21.** Claim 21 recites:

> The apparatus as in claim 15, wherein the apparatus is one of: a data processing device, a portable device, a portable data processing device, a multi touch device, a multi touch portable device, a wireless device, and a cell phone.

449. Claim 21 claims an apparatus in claim 15 and adds a limitation analogous to claim 7, "wherein the apparatus is one of: a data processing device, a portable device, a portable data processing device, a multi touch device, a multi touch portable device, a wireless device, and a cell phone." Accordingly, the Accused Products discussed in connection with claim 7 infringe claim 21 for the reasons discussed in connection with claim 6. To the extent that this claim is not met literally, in my opinion it is met under the doctrine of equivalents because each of the above-listed products accomplishes the same function in the same way to achieve the same result.

E. Samsung's Devices Have Been Modeled on Apple's iOS

450. Based on documents that I have reviewed, Samsung appears to have modeled the scrolling, pinch zoom and rotation features in its products after those in Apple's iOS.

451. In December 2009, Samsung's C.E.O. issued "instruction items" for 2010, stating that "going forward our comparison standard is Apple iPhone. In High End cases, evaluate with iPhone standard." (SAMNDCA10907803.) The then principal engineer of Samsung's Mobile R & D, Dongsub Kim, reiterated this sentiment in an email to several at the company, saying, "Henceforth our standard for comparison is the Apple iPhone." (SAMNDCA1097800 at -801.)

452. In an email from Senior Designer Eunjung Chang in December 2009 to an undisclosed number of recipients, Chang summarized the results of a UX informational meeting with several European subsidiaries. Chang reported that many "strongly request multi-touch (pinch interaction)." (SAMNDCA10015268 at -273.) Furthermore, several at the meeting informed about "the market's need for this [pinch interaction] in a variety of features such as a browser, game, photo. "They feel that whether this is installed in a product is an important factor when customers make purchases because it is convenient and fun." Others went as far as to say the pinch interaction was "absolutely necessary for multimedia contents and Internet browsing." (*Id.*)

453. In February 2011, Tae Woo Rhim stated, "Enabling zoom in all mobile versions is a directive from Head of Verification group." (S-ITC-003401550.)

454. Many Samsung documents show that Samsung measured the implementation of pinch zoom and scrolling on its phones against Apple's products. Usually, these head-to-head

comparisons are in the form of charts measuring smoothness, response time, and feel of these features. (SAMNDCA00229419; SAMNDCA00229399; SAMNDCA00201351; SAMNDCA00201642; SAMNDCA00229449; SAMNDCA00525362; SAMNDCA00525359; S-ITC-003680292 at -299; S-ITC-003409246 at -253; S-ITC-003524055.)

455. Samsung developed patches to improve zoom and scroll functionality in comparison to Apple. After one such U1 browser scrolling patch was applied to a Samsung product, Ioi Lam wrote Jaegwan Shin saying, "initial response for scroll looks good. However, they feel like zoom-in is a little bit heavy compared to iPhone after applying the patch." (SAMNCA00229440.)

F. The '915 Patent Could Not Be Designed Around Without Rendering the Accused Products Much Less Useable

456. I have been asked to consider whether the Accused Products could be re-designed so that they do not infringe the '915 patent. In my opinion, any such re-design would make the Accused Products much less useable, render them inconvenient for users, and deprive them of intuitive functionality that smartphone and tablet users have come to expect.

457. The '915 patent provides functionality that is central to all of the Accused Products: the ability to distinguish automatically between a one-finger scroll call and a two-finger gesture such as a zoom or rotate gesture. This functionality is highly intuitive; indeed, many users who experiment with devices equipped with this functionality immediately understand how to use them without any explanation. Scrolling, zooming and rotating are among the most common actions users take with the Accused Products, and are used in multiple applications.

458. Potential alternative designs that do not practice the '915 patent would be far less useful. A smartphone that required users to press a key in order to zoom or un-zoom, for example, would be much less intuitive and would provide a much less satisfying user experience. than devices that practice the '915 patent.

equivalent to the corresponding structures described in the '891 patent for performing the functions in claim 74. Accordingly, these three Samsung Accused Products infringe claim 74.

VIII. CONCLUSION

593. My opinions are subject to change based on additional opinions that Samsung's experts may present and information I may receive in the future or additional work I may perform. I reserve the right to supplement this Report with new information and/or documents that may be discovered or produced in this case, or to address any new claim constructions offered by Samsung or ordered by the court. With this in mind, based on the analysis I have conducted and for the reasons set forth above, I have preliminarily reached the conclusions and opinions in this Report.

594. In connection with my anticipated testimony in this action, I may use as exhibits various documents produced in this Action that refer or relate to the matters discussed in this Report. I have not yet selected the particular exhibits that might be used. In addition, I may create or assist in the creation of certain demonstrative exhibits to assist in the presentation of my testimony and opinions as described herein or to summarize the same or information cited in this Report. Again, those exhibits have not yet been created.

Dated: March 22, 2012

/s/
Karan Singh