EXHIBIT I

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

C O N F I D E N T I A L
A T T O R N E Y S' E Y E S O N L Y
O U T S I D E C O U N S E L

VIDEOTAPED DEPOSITION
BENJAMIN B. BEDERSON, Ph.D.
Washington, D.C.
Saturday, September 17, 2011
9:30 a.m.

Job No. 41965

Reporter: Linda S. Kinkade, RDR, CRR, RMR, CSR
Videographer: Conway Barker

The following is the videotaped deposition of BENJAMIN B. BEDERSON, Ph.D. held at the offices of:

Morrison & Foerster
2000 Pennsylvania Avenue, N.W.
Washington, DC 20005

Taken pursuant to applicable Rules of Civil Procedure, before Linda S. Kinkade, Registered Diplomate Reporter, Certified Realtime Reporter, Registered Professional Reporter, Registered Merit Reporter, Certified Shorthand Reporter (CA), and Notary Public, in and for the District of Columbia.

APPEARANCES:

On Behalf of Plaintiff APPLE INC., a California corporation:

MICHAEL A. JACOBS, ESQUIRE
DEOK KEUN AHN, ESQUIRE
Morrison & Foerster
425 Market Street
San Francisco, California 94105

On Behalf of Defendant SAMSUNG ELECTRONICS CO.:

ERIC HUANG, ESQUIRE
AARON KAUFMAN, ESQUIRE
Quinn Emanuel Urquhart & Sullivan
51 Madison Avenue
22nd Floor
New York, New York 10010

me to explain what it does or how it does it?

Q. Let's start with what it does.

A. I believe that this finds the -- this determines the bounds of an email header that most overlaps with the specified cursor rectangle, which is a -- in the application I believe is a blue highlight.

Q. So let's -- these will be concepts I think we'll be using as we go through the rest of the code, so let's make sure we understand them clearly. We talk about an email header. We're talking about the actual set of characters that is associated with an email on a list of -- on a list display of emails, correct?

A. Is it all right if I just describe what it does in my words?

Q. Sure.

A. So when you are zoomed into the email tile application in X-node, one of the parts of that display is a list of email headers where each header is actually an image where the image visually looks like the kind of information commonly included in an email header, such as, I believe, it includes who it's from and what the subject is.

So there is a list of these email headers and there is also a partially transparent highlight -- I'll just -- the code refers to it as a cursor. I'll just call it a highlight. I think that's -- that highlight can be moved up and down among the different email headers, and the email headers can be moved up and down under the highlight in interaction of this application tile.

And so this code is given a particular position of the list and a particular position of the highlight. It determines the rectangle -- the rectangular bounds of a header that is the header that is most under the highlight of which there is only one.

Q. And at this stage of the code all it's going to do is answer that question, which header is closest; it's not necessarily going to take any action based on that decision. Correct?

MR. HUANG: Objection to the form.

THE WITNESS: I don't believe that this code has any what are sometimes called side effects. I believe all it does is return those bounds, the calculated bounds.

BY MR. JACOBS:

their finger beyond a certain number of pixels, the email list will then follow the finger up and down. As they move their finger up and down, when the finger lets -- lifts off the screen, the dragging will stop and specialized method application specific method SnapObjectToHighlight will be called.

Q. And SnapObjectToHighlight is where?

A. SnapObjectToHighlight is defined in the class EmailAppNode.

Q. So it's not in the code that we've reviewed so far.

A. Correct.

Q. Let me see if I can sum this up. We have seen a determination made in this code, in Exhibit 211, EmailListNode cs, we saw a determination made of which header image is closest to the cursor. That was one of the steps that early on in the code we saw, correct?

MR. HUANG: Objection to form.

THE WITNESS: We found a mechanism to determine -- well, to compute the bounds and to determine which email header node -- the language I would use would be most overlaps the highlight cursor.

BY MR. JACOBS:

Q. Most overlaps is the key there, right?

A. Yes.

MR. HUANG: Objection.

THE WITNESS: That's the actual calculation that gets computed.

BY MR. JACOBS:

Q. And then another function we saw was the function of determining whether the finger or other input device has moved far enough such that, depending on the constants that have been input, the displayed item should itself move in response to that input.

MR. HUANG: Objection to the form.

THE WITNESS: Since -- that way you said was long. Can I just try and restate the same thing in my words?

BY MR. JACOBS:

Q. Please.

A. When the user moves their finger, the email header list will follow their finger after the finger has moved a certain definable number of pixels.

Q. And then when the finger -- when the user lifts his finger, a function called

SnapObjectToHighlight is called.

A. Correct.

Q. And that's defined in, I believe it's defined in Email.cs, which is Exhibit 212, if you want to take a quick look at 212. I promise you we'll get a break in a second.

MR. HUANG: Objection to the form. Go ahead.

THE WITNESS: I think there was a question that I was going to answer. Can you read back the question?

BY MR. JACOBS:

Q. Yes. Sure. SnapObjectToHighlight defined in Email.cs.

A. So to clarify, assuming that Exhibit 212 is -- represents the contents of the file Email.cs and that this is the file that I gave to my counsel --

Q. Yes.

A. -- then, yes, that method is defined inside of this source code. And in looking at this source code very briefly, I need to make one further clarification to something I had said.

I said that EmailAppNode appears to be the

then. Can you describe what public SnapObjectToHighlight at the bottom of page 29 does?

A. Yes. So this is the method that gets called in the OnMouseUp method that I described previously when the user lifts their finger off the device after they finish dragging.

It takes two parameters, Node and Animate. The first parameter that is passed in from the caller OnMouseUp is DraggedNode, which I believe is the node that was being dragged up and down, which I believe is slidingNode -- refers back to slidingNode in the original code.

And the second parameter is called Animate, which is passed in as true. So Animate is true.

So the definition of SnapObjectToHighlight takes the overall email header list that is displayed on the screen and moves it with animation over time so that the email header object that most overlaps the highlight cursor ends up completely underneath the highlight cursor.

Q. Can you explain how that occurs?

A. Yes. The first line determines the bounds of the highlight cursor in the appropriate coordinate system.

BY MR. JACOBS:

Q. And is it the very same code that executes that functionality in the following two conditions: Condition 1, the cursor highlight bar is between email header images in the list; condition 2, the email header is in the white display space underneath the last -- below the last of the email headers in the list?

MR. HUANG: Objection to the form of the question.

THE WITNESS: You used the word email header where I think you meant highlight cursor.

BY MR. JACOBS:

Q. I think I did, yes. So let me ask it again. Maybe now that I have stated it orally I can do it more clearly.

There are two possibilities for the email highlight cursor to be out of alignment with email headers. One possibility is it's in between email headers; the other possibility is it's after the last of the email headers. Correct?

A. Yes.

Q. Is it the exact same code that causes the email header to snap into alignment with the

email cursor bar in either of those two cases?

MR. HUANG: Same objection.

THE WITNESS: So there is only one code sequence of flow that performs snapping, and that same sequence is used wherever the email list is positioned vertically, including when the bottom-most email header is above the bottom of the screen.

BY MR. JACOBS:

Q. And the -- it is possible that, when the user lifts -- in the case of the depiction on page 6 of your declaration -- when the user lifts his finger, that the blue cursor bar and email header image are in alignment, correct?

MR. HUANG: Objection to the form.

THE WITNESS: So at the time the user lifts off, it's possible that one of the email headers is already completely aligned underneath the highlight cursor -- highlight cursor.

BY MR. JACOBS:

Q. That's my question.

A. Yes, that's possible.

Q. So I think you did this before, but if you could just again point us to the code that tests whether that condition has been met.

A. Okay.

Q. Or, to state the obvious, or that the condition has not been met and that the snap functionality must be invoked in order to create alignment.

MR. HUANG: Objection to the form.

THE WITNESS: So I believe those -- the code does not distinguish between those two conditions and it does exactly the same thing in the case where the email header is already aligned with the highlight. It performs the same animation over the same period of time, but the calculated ydiff, the amount that it moves, is zero. And so it will actually do the code, it will do the animation, it will render many times with several in-between states, and those in between states will all be the same. So it actually does do a kind of degenerate snapping.

BY MR. JACOBS:

Q. So the calculation that needs to be done in order to decide whether it's -- in order to decide whether there is any kind of visual movement that has to occur is the calculation of ydiff?

A. So ydiff calculates the amount of

movement. If ydiff is zero, then the code executed is the same, but the user will experience that nothing changes on the screen.

Q. Let's just go through again the calculation of ydiff with that case in mind. I think that was on 212 at 28.

A. On pages 28 and 29 there is the method SnapPositionToObject, which we've already discussed, and at the top on page 29, near the top, it calculates ydiff to be sourceRectangle.top minus Rectangle.Top.

So in this case where the top of the email header is the same position as the top of the highlight cursor, then these two top values will be the same, whatever they are, and so when you subtract one from the other, they will have the value of zero. Ydiff will then have a value of zero. Three lines later Matrix -- sorry -- the OffsetY property of the Matrix variable gets decreased by an amount whose value is ydiff. If ydiff is zero, then it won't be decreased.

So then, when it does the actual animation by calling Animate to Matrix and it passes in that Matrix variable, if Matrix has not been changed from its initial value, then the result of calling

Animate to Matrix will be that nothing on the screen changes position.

Q. And then looking -- thinking about this from the other end of the email list, thinking about the top of the email list, is it possible that the user ended his -- the user experience with the blue bar -- by lifting the finger at the point where the blue cursor bar is above the first email header?

MR. HUANG: Objection to the form.

THE WITNESS: I don't recall.

BY MR. JACOBS:

Q. Is there some -- is there something that would stop that from happening?

MR. HUANG: Same objection, objection to the form.

THE WITNESS: I haven't seen anything in the code we looked at today that would stop the highlight cursor from ending up above the top-most email header.

BY MR. JACOBS:

Q. And if there is no obstacle to that occurring, then the same SnapTo functionality should cause the blue bar to go to the top email header, correct?

Do you see that?

A. Yes, I do.

Q. What's the reference to depending on the degree of the over-pan?

MR. HUANG: Objection, form.

THE WITNESS: If in the example that's described here with these images, for a concrete example, if the user has dragged -- moves -- touches the screen, drags their finger up so they are moving the email list up, and the bottom-most email header is above the bottom of the screen, if -- so this is the over-pan position -- if they have over-panned to a degree such that that bottom email header is partially overlapping with the highlight cursor, then it will snap back so that the bottom-most email header is aligned with the bottom of the screen in this situation.

BY MR. JACOBS:

Q. And if -- so in order for the snap back to occur, there must be some partial overlap at the end -- when the user lifts his finger?

MR. HUANG: Objection to the form.

THE WITNESS: If the -- it depends on

how much the bottom email header is above the bottom of the screen. So if it's -- that's why I said, depending on the degree of the over-pan, if the degree is such that there is some overlap, then it will snap back.

BY MR. JACOBS:

Q. And if the degree is such that there is no overlap, what happens?

A. I believe then it does not snap back. Then I believe it just stays in that position.

Q. And that's just a function of the state of the code as -- let me start over again.

That is because the code in its -- in the state in which you've provided it to us doesn't have a case for no overlap or beyond overlap; is that correct?

A. Well, the code -- it does what it does. I mean, it does a very specific set of features and interactions, as we talked about, and that's what it does. So --

Q. Let me ask it this way. Point us where in the code the test is set forth in a way that such that that constraint, that there must be some overlap is present.

A. So in Exhibit 212 --

Q. Email.cs.

A. -- Email.cs, pages 29 to 30, method SnapObjectToHighlight, around the fifth line of code it calls, GetIntersectingEmailItemBounds. As we discussed earlier, this returns the rectangle representing the bounds of the email header that most overlaps the highlight cursor implemented by that GetIntersectingEmailItemBounds method. If we're in this condition where there is no email header that overlaps -- sorry. If we're in the condition where the bounds of every email header does not overlap the bounds of the highlight cursor, then this method will return a sourceRectangle whose value is empty.

The next line of code in SnapObjectToHighlight says, if sourceRectangle is not empty, then it calls SnapPositionToObject. So in the case we're talking about, sourceRectangle would be empty and this SnapPositionToObject method would not get called.

Q. So I may not have been tracking your description. On the bottom of page 29, three -- well, four lines up from the bottom, if you include the brace, there is if sourceRectangle

Microsoft in this time period.

Q. And Amy Karlson was with you at HCIL?

A. Correct, she was.

Q. Same for Aaron Clamage?

A. Correct, yes, he was.

Q. Now, if you turn to paragraph 8, it says, attached as Exhibit F is an executable version of LaunchTile -- line 18, 19, 20 -- which is identical or substantially similar to the version of LaunchTile that we demonstrated in our video and discussed in our paper and presentation at the CHI conference. Do you see that?

A. I do.

Q. The reference to the possibility that it is substantially similar, what are you -- why did you -- why might it not be identical but only substantially similar to the version that you demonstrated?

MR. HUANG: Objection to the form.

THE WITNESS: If you don't mind, I'm just going to read this section of my declaration to get the whole context.

BY MR. JACOBS:

Q. Sure.

A. There is at least two reasons that I said that. One is that I recall that we stopped development of LaunchTile after we submitted the paper, and so -- and the -- we made the video at the time that we submitted the paper. So we made the video with the version of the code that was available right at the time we submitted the paper. We stopped development just about after that time. So I don't think there was any further development. And then, in addition, the executable version that I attached works in the same way that the video does, and so there was -- I couldn't see any difference between the executable that we ran and the demonstration in the video. So it's further corroboration with my memory that these were all done contemporaneously.

Q. But the reason for the qualification of substantially similar is what?

A. That I don't have a precise time stamp of that source code that correlates with the video that shows that it was done exactly the same day and that it was exactly the same code.

Q. The source code control system that we discussed earlier this morning could give you

THE WITNESS: Inasmuch as it refers to snapping and over-panning and under-panning, yes.

BY MR. JACOBS:

Q. Is there functionality in -- I need to understand what that might exclude in your declaration. So if you take a look at paragraphs 10 through 18, are there any paragraphs that are describing LaunchTile that you're unsure of how they apply to XNav?

A. I'm going to have to read those paragraphs.

So the features described in paragraphs 10 through 18 of my declaration LaunchTile and XNav do in fact behave in identical or substantially similar ways.

Q. So, again, that phrase "substantially similar" comes up. Is there any respect in which they are not identical?

A. I'm not aware of any.

Q. The phrase, "over-pan or under-pan command," do you see that?

A. Where are you referring to?

Q. In paragraph 20.

A. Yes, I do.

BY MR. JACOBS:

Q. Let me start over. Can you please demonstrate the case where the highlighting bar is above the email list because you've pulled the email list down and there is no overlap between the highlighting bar and the top-most email header?

MR. HUANG: Same objection.

THE WITNESS: Okay. So I'm dragging the email list down so that there is no overlap between any of the email headers and the highlight bar and then I let go.

MR. JACOBS: Okay. Thank you.

VIDEOGRAPHER: Off the record at 1:34.

(Brief interruption.)

VIDEOGRAPHER: Back on the record at 1:34:44.

BY MR. JACOBS:

Q. Dr. Bederson, in your declaration you refer to some other portions of code as relevant, at paragraph 22 of your declaration. I believe you have ShellForm in front of you, yes, as Exhibit 213. We have located the lines that you referred to at paragraph 22, that being line 746, 760 and 764 at page 19.

BY MR. JACOBS:

Q. Let me start over. Can you please demonstrate the case where the highlighting bar is above the email list because you've pulled the email list down and there is no overlap between the highlighting bar and the top-most email header?

MR. HUANG: Same objection.

THE WITNESS: Okay. So I'm dragging the email list down so that there is no overlap between any of the email headers and the highlight bar and then I let go.

MR. JACOBS: Okay. Thank you.

VIDEOGRAPHER: Off the record at 1:34.

(Brief interruption.)

VIDEOGRAPHER: Back on the record at 1:34:44.

BY MR. JACOBS:

Q. Dr. Bederson, in your declaration you refer to some other portions of code as relevant, at paragraph 22 of your declaration. I believe you have ShellForm in front of you, yes, as Exhibit 213. We have located the lines that you referred to at paragraph 22, that being line 746, 760 and 764 at page 19.

right near the second line from the bottom, and those are the three lines that are referred to in my declaration.

Q. Can you explain what the code is doing on page 20?

MR. HUANG: Objection to the form.

MR. JACOBS: Actually it was a good objection.

BY MR. JACOBS:

Q. Can you explain what the code on page 20 is doing?

MR. HUANG: Same objection.

THE WITNESS: So the code on pages 19 through 21 are -- all implement a method called OnMouseUp. OnMouseUp is part of a class called LaunchPointListener, which is a subclass of Ppan EventHandler. The class in general supports dragging and snapping in the Zone view. The method in question, OnMouseUp, gets called when the user lifts their finger off the screen after they have finished dragging.

And the particular part of the code we're looking at examines what kind of dragging was being done, in particular, there is a condition around line 7 on page 20 that says:

If direction equals horizontal -- I'm just paraphrasing this code to make it easier to understand -- then it executes one body of code.

So this means, if the user was dragging the zones horizontally, that is, left to right, then there is another conditional that I interpret as meaning, if the user has dragged more than one-sixth of a screen width, then it does one thing, and the thing that it does is it will snap to the next Zone in the direction that they were moving, and that is partially displayed on the screen. If that condition was false, that is, if they dragged less or equal to one-sixth of a screen width, then it would call AdjustToCurrent, which I believe will end up resulting in the Zone snapping back to the Zone that the user started out in.

I'm still answering this question. That was the case for if the user was dragging horizontally.

The next conditional is if the user was dragging vertically, and it does parallel code for snapping forward or backward. And then there is a condition if they were dragging neither horizontally nor vertically, then it would snap back to the Zone

they started out at.

Q. Where is that last case coded?

A. It's the last three lines of page 20 where it says, else, open brace, landscape.AdjustToCurrent (true) and then closed brace.

Q. So that's if neither the width nor the height condition are satisfied, then it leaves that possibility; is that the way that works?

A. Right. If the user had dragged neither horizontally nor vertically.

Q. So camera.ViewBounds.width or camera.ViewBounds.height, those are variables?

A. Not quite.

Q. What are they?

A. Those are compound statements. Camera refers to a variable which represents the camera of the system. ViewBounds is the bounds of what the camera sees and the width -- sorry. Let me see if I can say that more clearly.

ViewBounds is the bounds of the portion of the abstract space that is displayed on the screen.

Q. In the -- when you and I were discussing the email application of the snap

A. I believe that OnMouseUp gets called for each -- not for each -- for many different positions of the finger while the finger is being dragged that include arbitrary positions on the screen, so the finger -- it can report positions or positioning includes an X and a Y position that can be any position on the screen.

Q. And how does it -- can you explain how the code is treating the second phase of the three-step interaction -- interaction sequence -- that you discussed?

A. This code we're talking about is OnMouseUp, so this actually refers to the third part of the interaction sequence when the user lets go of the mouse -- or lets go of the screen. Excuse me.

Q. So the calculation of one-sixth of this screen width I understood your testimony to be that that's the -- I think I misunderstood your testimony. Now I see what's going on.

This one-sixth of the screen width that's being referred to is a comparison of one-sixth of the screen width with the amount of the -- the width of the tile, correct?

A. So in this long statement that we're

talking about on page 20, inside the horizontal condition, if Math.Abs, MousePressedCanvasPoint --

THE REPORTER: If you would read slower.

THE WITNESS: Sorry. Let me restate that. Just to make sure we're -- to be clear, in this statement that begins if Math.Abs (MousePressedCanvasPoint.X minus e.CanvasPosition -- sorry -- CanvasPosition.X, right parenthesis, greater-than sign, left parenthesis, camera.ViewBounds.width/ 6, right parenthesis, in this line that we're talking about, that last part, camera.ViewBounds.width/6, refers to one-sixth of the total display of the width of the screen.

BY MR. JACOBS:

Q. And that -- and the comparison is of one-sixth of the total display of the screen to what?

A. So the other side of the condition is the difference between the point the user touched down on the screen and the current point, which is at this point in the code sequence is when the user lifted off on the

screen.

So this is saying between the time the user has touched down and the time the user has lifted up, this looks at how much -- what the difference is in horizontal position on the screen of the finger's -- of the finger, and it compares that to one-sixth of the width of the screen.

Q. So this algorithm depends on the first step and the last step of the interaction sequence and not on the intermediate phase of the interaction sequence, correct?

A. Not quite. And the reason is because this condition only gets executed if the direction is horizontal that was computed in a previous step. So this is -- this code gets executed only for the particular user interaction sequences where the application has determined that they are moving horizontally.

Q. If the -- step one of the interaction sequence is finger is -- touches on the screen at location X/Y and step -- and then in between, in the middle phase of the interaction sequence, the finger moves to the left and then back -- right to X/Y, is the amount of movement that is

think I quite understood.
BY MR. JACOBS:
Q. So the landing position of the finger and the finger-up position are identical, but in between the user went back and forth diagonally.
MR. HUANG: Same objection.
THE WITNESS: So the code we're talking about right now only executes if the direction variable is horizontal. So we haven't yet talked about how that might come to be. In the case that the direction is horizontal, if they press -- the user touched down, they moved in such a way that direction ended up being horizontal, and then they also had some diagonal or other movement and then came back to the ending point, the direction was still horizontal, then this calculation would still be zero.
BY MR. JACOBS:
Q. So I may need to ask you how the horizontal condition is measured.
A. All right. So I'm going to have to refresh my recollection by looking at some code to answer that question.
Okay. So going back to page 15 in this same

document, Exhibit 213, ShellForm.cs, we've been talking about the class LaunchPointListener, which is a subclass of PPanEventHandler. And I've already described this code as generally supporting the dragging and snapping features in the Zone view.

The second method in this class, at the bottom of page 15, is a method called ShouldStartDragInteraction. I believe this method gets called to determine when the user is at the beginning of the interaction sequence -- sorry -- when the user has first started dragging during the second part of the interaction sequence, whether the visual display on the screen should start tracking, should start dragging underneath the figure.

With the expectation -- the way this code is written is that for the first few -- first some number of pixels of finger movement it does not follow the finger. It just -- nothing happens at all. And then after the finger has moved a certain number of pixels, has -- more than some threshold, after which the dragging interaction starts -- the dragging feature starts. So that's implemented in this ShouldStartDragInteraction. And that particular constant, the number of pixels the finger must move, is determined by this constant click

underscore threshold, which is defined elsewhere.

So until the finger has moved enough, that dragging has started, until that point essentially nothing happens on the screen. And, in particular, there is a method starting on page 16 entitled "Pan." This Pan method gets called whenever the user is in the second phase of interaction and they are dragging their screen and that threshold has been reached, that is, they have moved their fingers more than this threshold.

So once the screen starts following the user's finger, then Pan gets called. The first thing Pan does is it looks at a variable called Direction. Direction was initialized on page 15 in the beginning of the declaration of LaunchPointListener class, around the middle of the screen, to be equal to a constant called none, meaning there was no direction initially specified.

So going back to the Pan method on page 16, the first thing this Pan method does is it says, if direction is none, then it compares how much the current position of the finger has moved in the Y direction, that is, vertically, compared to how much it has moved in the X direction, that is, horizontal. If it has moved more vertically than

display of finger movement from the beginning to the end of the interaction sequence, that movement, what defines the end point of that movement?

A. I'm sorry. I don't think I understand the question.

Q. On page 20, if there is more horizontal movement than one-sixth of the screen width, then the additional movement occurs, correct, beyond the movement that -- beyond the movement that was in place when the finger was lifted off the screen.

A. I'm sorry. I don't think -- what you said doesn't make sense to me.

Q. So what happens on page 20 in the -- with the horizontal condition being satisfied and finger movement of greater than one-sixth of the screen display size?

A. So if we've gone through this three steps of interaction, the user has touched down, they have dragged more than a threshold, this system has determined that they have been dragging in a more horizontal than vertical direction, and then the user lifts their finger off the screen, at the time the finger lifts off

the screen this OnMouseUp method is called. If it is determined that the amount that they have moved their finger horizontally -- sorry -- if the horizontal amount that they had moved their finger in this drag sequence is greater than one-sixth of the screen, then the contents of this "if" condition will get executed. Otherwise, the "else" condition will get executed.

Q. And when the "if" condition is executed, what happens?

A. So in the case that it was positive, so either has moved more than one-sixth of the horizontal -- let me restate that. If the user has moved their finger horizontally more than one-sixth of the screen width, then it looks to see if the user had been -- dragged their finger to the right of the starting point or to the left of the starting point.

In the case that they had dragged their finger to the right of the starting point, it calls the NavigateRight method. If they had dragged their finger to the left of the starting point, it calls the NavigateLeft method.

Q. Is the NavigateRight method present in

the ShellForm.cs code?

A. I don't believe so.

Q. What does the NavigateRight method do?

A. My recollection is that it will Animate the screen to the next Zone to the right if there is one.

Q. And when you say Zone, what is the -- what is the Zone in the code? What defines the beginning and end of a Zone in the code?

MR. HUANG: Objection to the form.

THE WITNESS: There is a correspondence in the data structures of the code between what we refer to in the user experience as a Zone. So there is a data structure in the code that represents each of the nine Zones.

BY MR. JACOBS:

Q. In the case in which the finger movement in the interaction sequence is less than one-sixth of the screen dimension, whether it's horizontal or vertical, what does the code do?

MR. HUANG: Objection to the form.

THE WITNESS: If you don't mind, I'm just going to answer for the horizontal just to

sets a callback method so that when the activity is completed it calls a method called PanToTileFinished. It sets a property to say that ShowThumbs equals true and SlowInSlowOut should be false.

This whole big, long conditional ends, and the last four lines of this long method are executed, which sets some values -- I don't know what they mean -- ActiveQuad equals tile; ActiveApp equals null, commented out line, and then another method tile.XNav.AnimateToMode, and I do not know what this line does.

After this method returns, this was the activate method, and the whole calling sequence of other methods that resulted in this getting called, at that point this activity which had been scheduled will start and the animation will occur on the screen, and there will be the visual snapback that I've referred to.

Q. In the case where the greater-than one-sixth test has been met and it's a -- and the horizontal-movement test has been met, and so it's -- greater than one-sixth of the horizontal-screen width has been met, and you are -- but in the direction in which you have

moved there are no more tiles, what happens?

A. So do you mean from the user experience or from the code?

Q. Let's start from the user experience.

A. So just to clarify, if you -- I think the situation you described cannot happen actually.

Q. So you can't move past -- you can't pull an edge of a Zone -- probably better to create an example.

If you have a Zone on a screen and on the right edge of the screen or the right edge of the Zone there are no more tiles --

A. Can we say Zones? There are no more Zones?

Q. No more Zones, and, hence, no more tiles, right, because tiles fill a Zone?

A. Yes.

Q. So to the right -- to the right of the Zone on the screen there is not another Zone, and the finger movement is more than 20% in a horizontal dimension of the screen width to the left, the Zone image does not move.

MR. HUANG: Objection to the form of the question.

THE WITNESS: I'm not sure -- I understood everything you said except for the 20% part.

BY MR. JACOBS:

Q. Okay. Actually I think it's regardless of whether it's 20%. If there is no -- if the Zone is filling the screen and to the right of the Zone there is not another Zone and the finger movement is to the left, does the -- regardless of whether it's 20% of the screen width or not, does the image of the Zone move to the left with the finger?

A. I believe in the set situation it will not.

Q. And what prevents that from happening?

A. In the code?

Q. Yes.

A. I do not know.

Q. So let's take a look back at ShellForm. There is a functionality called protected override void OnDrag.

MR. HUANG: Do you have a page?

MR. JACOBS: Yes. I believe it's page 17.

THE WITNESS: I see the OnDrag method

panning or attempting to pan the screen.

Q. And how does it -- what happens when there is no, in our case, where there is no Zone to the right?

A. I'm just going to look at this code for a moment. I'll figure this out in another minute so that it will settle.

Q. Not a problem.

A. To be honest, I'm having a little trouble understanding exactly how this code works. My interpretation of the code is not consistent with how it behaved, so I'm likely misunderstanding something here.

Q. Let me come at the topic this way. It is the case, as we discussed, that you can't cause the Zone to move to the left when the right-most boundary of the Zone does not have adjacent to it on the right another Zone, correct?

MR. HUANG: Objection to the form.

THE WITNESS: I think -- let me just --

BY MR. JACOBS:

Q. Say it your way.

A. There is a grid of 3x3 Zones that's

fixed in the code. So if you were in the right-most column, that is, any of the three right Zones, and you try and drag to the left, you will not be able to drag to the left.

Q. Why? Why did you design it that way?

MR. HUANG: Objection to the form.

THE WITNESS: I don't recall our thinking in that specific design decision.

BY MR. JACOBS:

Q. Was it a design decision?

MR. HUANG: Objection to the form.

THE WITNESS: Or possibly a lack of a design decision. I don't remember what our thinking was for that particular interaction detail.

BY MR. JACOBS:

Q. The source code for LaunchTile, were you able to locate that?

A. No, I was not.

Q. Any idea what happened to it?

A. Well, I know that Amy Karlson was primarily responsible for writing it. I believe she managed source code, and I don't think I probably followed it in that much detail. So that's why I -- when I looked, I didn't have it,

MR. HUANG: Objection to the form.

THE WITNESS: I wouldn't characterize it that way, but LaunchTile runs on the pocket PC platform. And we had a number of different devices from different manufacturers over the years, so I don't remember which specific one we showed on that day.

BY MR. JACOBS:

Q. Paragraph 20 says, XNav was meant to run on different mobile touchscreen devices and on different operating systems than LaunchTile. Do you see that?

A. Yes, I see that.

Q. So which operating system was LaunchTile intended for?

A. LaunchTile was intended for pocket PC, and, I believe, and Windows mobile platforms, and XNav was intended for the Windows XP and desktop operating systems. I think we tested it on Windows XP.

Q. Do you recall any specific demonstrations that you did of LaunchTile at the HCIL symposium in 2005 May?

A. Do you mean do I specifically have a recollection of me, sort of mind's eye, showing

it to an individual?

Q. Exactly.

A. No, I cannot think of a specific one.

Q. On paragraph 20 -- sorry -- on paragraph 25 you -- the declaration says that your team presented our work, which included LaunchTile and its user interface features including the snap and panning features discussed above. Do you see that?

A. Yes, I do.

Q. And what do you recall specifically of demonstrations or discussions of the snap and panning features discussed above in your declaration?

A. I think I -- I got confused in your question, if you don't mind repeating it.

(Record read.)

THE WITNESS: So I recall in the presentation we showed the video, and the video included the panning features and at least some of the snap features, and I recall generally demonstrating the software -- I recall both me and Amy generally demonstrating the software -- at least me and Amy. There may have been others as well. Aaron Clamage probably was there as

well demonstrating it. So that's what I recall.
BY MR. JACOBS:
Q. Were you demonstrating the software in a live basis in conformance with what was demonstrated on the video? Were you trying to map what was on the video to your live demonstration?
MR. HUANG: Objection to the form.
THE WITNESS: No. The video was a short, you know, narrow summary, and when we gave live demos it was much more casual. We would typically hand the device over to whoever we were showing it to, let them do whatever they want, ask us any questions. They had already seen the video, so they typically would want to go beyond that.
BY MR. JACOBS:
Q. Do you recall anything specifically being demonstrated in May 2005 that wasn't in the video?
A. I don't recall the specific details of what was or was not shown to any specific individual.
Q. Let me show you an email that you produced to us.

A. I don't recall having any disagreement with that.

(Exhibit No. 217 marked for identification.)

BY MR. JACOBS:

Q. 217 is another email that may fall into the category of a response, although it's not in a string with John's email. It may refresh your recollection. So 217 is an email from you to John SanGiovanni dated August 25th, 2005, subject, LaunchTile source, and in this email you give him a location from which he can download the source code for LaunchTile for the desktop. Do you see that?

A. Yes, I see that.

Q. And then you must have sent him an email so that he could decrypt an encrypted file, correct?

A. That's implied by this email.

Q. And the reason it was encrypted was what?

MR. HUANG: Objection to the form.

THE WITNESS: I think I vaguely -- well, I guess I can't remember. I don't have a specific recollection of why I would have

encrypted it.
BY MR. JACOBS:
Q. Do you have a general sense of why you were encrypting files that you were maybe -- the files in particular that you were sending to Microsoft?
A. I guess the reason I hesitate is I don't remember whether this was my request or somebody else's request. If it was somebody else's request, then it was because they requested it, and if it was at my initiation, I guess I thought it was a good idea at the time. I just don't really remember why.
Q. You do recall Microsoft receiving the code. I think you said a few minutes ago that, when I asked you what became of LaunchTile, that at least you transmitted the code, I guess, maybe to be precise about what you said. Did Microsoft receive the code?
A. I guess I assume they received the code and there may have even been follow-up emails or other communications. I don't have a specific recollection of them representing that they received the code.
Q. So to expand from that slightly, you

don't have a recollection of them saying, we got the code, we tried it out, it works really well here, we like it, something like that?

A. I don't have any specific recollection.

Q. How about a general -- any general recollection of their reaction to the product when you -- assuming you delivered it to them?

A. Well, they had already been familiar with it. This was a year later. The conference had already come and gone. So I wouldn't -- so the answer is, I guess, no.

Q. Let me show you a couple more emails. This is going back in time again.

(Exhibit Nos. 218, 219 and 220 marked for identification.)

REDACTED

part of the thread so I can make more sense of that?

Q. Sure.

A. Okay.

Q. Looking at John SanGiovanni's email to you of May 17th, 2005, Re Enhanced LaunchTile PowerPoint, he says, thanks, yes, making great progress on the IP licensing front. I convinced MSR -- I believe that's Microsoft Research, correct?

A. Probably.

Q. -- to let me write up the LaunchTile shell concepts together with some of Daniel Robbins' ZoomZone claims into a patent. I'm anxious to see what the U.S. Patent Office thinks.

Do you see that?

A. I do.

Q. What information do you have beyond this email on that patent application?

MR. HUANG: Objection to the form of the question.

THE WITNESS: I believe they had filed an application that is on the -- I believe that there is -- I believe Microsoft has filed an

application for some of the things related to LaunchTile and this other project that was going on at Microsoft at the time, ZoneZoom.

BY MR. JACOBS:

Q. And what was ZoneZoom?

A. My general recollection is that it included the idea that you could break a cellphone user interface into a 3x3 grid and map it to the 3x3 grid of a typical 1 through 9 NumPad so that you could press one of the numbers on the numerical keypad and have that interact with the grid on the visual display.

MR. JACOBS: Why don't we take a break.

VIDEOGRAPHER: This is the end of tape 4. Off the record at 3:29.

(Proceedings recessed.)

VIDEOGRAPHER: This is the beginning of tape 5 in the deposition of Dr. Bederson. On the record at 3:47.

BY MR. JACOBS:

Q. Dr. Bederson, how are you being compensated for your work on this case?

A. How? At an hourly basis, based on the number of -- the amount of time I have put into

work on this case.

Q. What is your hourly rate?

A. $450 an hour.

Q. And that is the rate that you have in your understanding, your agreement, with Quinn Emanuel on behalf of Samsung?

A. Yes, I believe it is.

Q. How many hours have you worked so far on the case?

A. I don't recall exactly.

Q. Roughly?

A. Maybe 40 or 50.

Q. Your understanding is that your work in this case is as a fact witness, correct?

A. That's my understanding.

Q. You haven't been asked to form any expert opinions on the relationship between what you did and Apple's patents, correct?

A. Correct, I have not.

Q. Have you looked at the Apple patent that Samsung is asserting your work against?

A. I've reviewed it.

Q. But not formed any opinions at Samsung's request on the relationship between your work and the patent, correct?

dragging a Zone, does the blue dot move with the Zone?

A. No. The blue dot stays in the center of the screen.

Q. And that's in fact, if we go back to the demonstration, we would see that, correct?

A. Correct.

Q. So blue remains centered, the Zone moves, and, as it says here in the article, each Zone is depicted with an empty center hub during dragging, correct?

A. Correct.

Q. And then, upon thumb release, the zoom space animates to align, I guess, leaving aside the 20% issue, what it might have said is, upon thumb release, the zoom space animates such that the zoom space is aligned with blue, right? Blue didn't move.

A. Correct. It doesn't actually say that blue moves in this, but I agree that it is written in a slightly awkward way. It says the zoom space animates, so it's pretty clear that the zoom space is the thing that's animating and it results in the zoom space being aligned with blue, but the way it's written is to align with

blue, so it's just very -- perhaps slightly awkward, but I think the meaning is pretty clear.

Q. Okay. So what we are saying here is, upon thumb release, the zoom space animates such that the specified zone's empty hub is aligned with blue, specified by the virtue of the 20% rule.

A. Well, you said 20%. I think dividing by 6 --

Q. Oh, you're right.

A. That's why I was confused. It's not quite 20%. This described the interaction in terms of hubs. The effect is exactly the same as if we had said center the Zone on the screen because of the fact that the blue dot is centered on the screen and because of the empty center hub is centered within the Zone. So it's described in a different way. The effect is the same.

Q. And the next sentence, the visual and animated guidance ensures the user is never caught between zones, what is that sentence driving at?

A. Actually it says automated guidance,

Then at time 1:15 seconds the user again touched the screen, dragged down, let go, which resulted in snapping forward to the Zone above.

And that's the end of the video.

BY MR. JACOBS:

Q. In either of the videos did we see the activity that's described in paragraph 14 of your declaration?

A. Neither video showed the activity described in paragraph 14 of my declaration.

Q. And that's what your declaration calls the under-panning case, correct?

A. I don't think -- paragraph 14 doesn't use that term, but I believe this describes the concept that was described earlier as under-panning.

Q. And to get back to the way the source code works, that's the case where in the three-stage interaction sequence where, after landing on the screen, the finger is moved less than 20% in the relevant direction -- sorry -- yes, less than one-sixth in the relevant direction such that there is what the declaration describes as a snapback.

A. Correct.

LaunchTile is motivated by the idea that, if there are some places that are convenient for the interface to go to, then you should make the interface naturally take you to those places and not let you get stuck in inconvenient places.

Q. So you published an article recently, "The Promise of Zoomable User Interfaces."

Mark this as the next in order.

(Exhibit No. 222 marked for identification.)

BY MR. JACOBS:

Q. The Promise of Zoomable User Interfaces by Benjamin B. Bederson, 2011, Taylor & Francis. What was this published in?

A. This was published in a journal named Behaviour & Information Technology.

Q. In 2011?

A. Yes.

Q. On page 4 you have a discussion of Desert Fog citing Jul and Furnas. Desert Fog labels a phenomena that you describe as allowing users to fly through the space going absolutely anywhere including deep into the spaces between objects. Do you see that?

A. No, actually. Sorry. Where are you?

Q. It's on the right-hand column of -- it's such a vivid image I thought it might just jump from the page. The right-hand column of page 4, second paragraph.

A. Yes, I see this.

Q. So just to maybe start a little bit earlier, different zoomable user interfaces have also had various navigation mechanisms, which are ways for users to move through the space. Again, there is a trade-off between flexibility and usability. Some interfaces allow users to fly through the space going absolutely anywhere, including deep into the spaces between objects, resulting in some researchers labeling this phenomenon Desert Fog, Jul and Furnas. Then you say, very few other applications let a user navigate beyond the actual content.

Can you explain the contrast you were drawing there between ZUIs and other applications?

MR. HUANG: Objection to the form.

THE WITNESS: Sure. So if we continue reading this paragraph, it describes this idea of not letting you navigate between the actual content. I believe it says, almost every document browser and editor limits navigation to

the available content with the notable exception of Microsoft Excel's scroll bar arrows, Apple numbers, and Google -- I'm sorry -- with the notable exception of Microsoft Excel's scroll bar arrows. Apple numbers and Google spreadsheet, on the other hand, do limit navigation. On the other hand, some interfaces allow you only to click on objects to zoom into them and click on a zoom out button to zoom out, making it impossible to get lost, but also giving less control over exactly where you look.

So the point of this paragraph was to describe that there are some applications that let the user navigate in space possibly -- navigating can be simple scrolling or it could be this kind of zooming navigation, which is a little bit more uncommon, or it could be 3-D navigation in a 3-D world.

Sorry. I was describing that sometimes you can navigate to a place where there is no content. If there is no content, then you're kind of in a place that essentially -- typically -- represented with an empty screen. And that was a concern because that would make a user feel disoriented since there is nothing on the screen.

And I said that it was more common for applications to stop a user from navigating to a place where there was no content, although it occurred, both in widespread applications like Excel and in many ZUIs, in at least those.

Q. So the basic contrast you were drawing was between those ZUIs that are flexible but haven't addressed this problem of getting lost in Desert Fog, and most applications which do constrain you to the space that's filled by content. Is that -- am I capturing the essence of your paragraph correctly?

MR. HUANG: Objection to the form of the question.

THE WITNESS: The paragraph said -- well, it didn't say "most." It said there were few applications that let you move to a place where there is no content, although I did describe some, and many constrained you to navigating only within available, visible content.

BY MR. JACOBS:

Q. And that -- but you were describing that, as of 2011, there remains this problem in ZUIs of flying through the space going

BY MR. JACOBS:

Q. Just as you look at some of the applications, the ZUIs that are listed on the table, do you regard the Apple iPhone, year 2007, but the iPhone you're referring to, I guess, is the 2010 version of it, do you regard that as having resolved the problem of getting lost in Desert Fog?

A. So in this particular -- in this table I'm to read, the thing that comes closest to the Desert Fog issue is the right-most column that talks about navigation mechanism for zooming, and for the Apple iPhone it says, tap to zoom in, physical button to zoom out.

So I think it's probably referring to the home screen application icons where you tap on one of those application icons and it has a zooming transition. It zooms in to launch the application icon and you can press the hardware button at the bottom of the device to zoom out.

So at least in that case it solves -- that interface solves the Desert Fog problem as described. So this is a case where, with some external knowledge, you could interpret this to understand that, but it was not explicit in this --

not fully explicit in this table.

MR. JACOBS: Could you just give us like three minutes to make sure -- oh, actually, we need to go one more demo, right? We figured out how to move the blue highlighter in XNav. And so if we can get you over by the video to just demonstrate that mode, that would be great.

THE WITNESS: Okay.

VIDEOGRAPHER: Off the record at 4:37.

(Brief interruption.)

VIDEOGRAPHER: Back on the record at 4:39.

BY MR. JACOBS:

Q. Dr. Bederson, we've given you again the Sony Vaio with the XNav on it that we were demonstrating earlier in the deposition, and we've now learned how to move the blue highlighter into the center of the screen and illustrate the case where the blue highlighter is between -- it ends up between email headers when the stylus or the finger is lifted from the screen; is that correct?

A. Yes.

Q. Can you demonstrate that for us?

A. Actually right now that highlight is

A. Correct.

MR. JACOBS: I think we're done. Thank you.

MR. HUANG: Thank you.

THE WITNESS: Thank you.

VIDEOGRAPHER: This concludes the deposition of Dr. Bederson. Off the record at 4:57 and it consists of five tapes.

(Proceedings concluded.)

//

(Signature having not been waived, the deposition of BENJAMIN B. BEDERSON, Ph.D. concluded at 4:57 p.m.)

CERTIFICATE OF SHORTHAND REPORTER

NOTARY PUBLIC

I, Linda S. Kinkade, RDR, CRR, RMR, CSR, the notarial officer before whom the foregoing proceedings were taken, do hereby certify that the foregoing transcript is a true and correct record of the proceedings; that said proceedings were taken by me stenographically, to the best of my ability, and thereafter reduced to typewriting; and that I am neither counsel for or related to, nor employed by any of the parties to this case and have no interest, financial or otherwise, in its outcome.

IN WITNESS WHEREOF, I have hereunto set my hand and affixed my notarial seal this 17th day of September 2011.

_____

Linda S. Kinkade

NOTARY PUBLIC IN AND FOR
THE DISTRICT OF COLUMBIA
My commission expires: July 14, 2012