Snapdragon SDK Test App

Release Notes

March 5, 2014

QUALCOMM Incorporated
5775 Morehouse Drive
San Diego, CA 92121-1714
U.S.A.

Contents

1 Revision History

Revision	Date	Description
A	August 2013	Initial release
B	February 2014	Addition of test cases targeting: ■ Facial Recognition ■ Device Info

2 Introduction

The Snapdragon SDK TestApp contains sets of tests that validate the platform implementation of the feature API, supporting software, on a device.

Manufacturers can use the TestApp to validate that their devices support the feature interface prior to release of the device to the ecosystem. Developers can use the TestApp to determine the level of support on the device they are developing to.

The TestApp will be updated to be in sync with each release of the Snapdragon SDK. More information regarding the use of the TestApp may be found in the user guide.

3 New Features

Feature SDK	Test Name	Description	Release
Facial Recognition	fp.add.person.VALID_ENTRY	Base Case: Test checks if the person in the frame is added successfully.	Feb. 2014
Facial Recognition	fp.add.person.NO_FACE	Negative Case: Test checks to see if adding a frame with no faces, fails	Feb. 2014
Facial Recognition	fp.add.person.ADD_SAME_FACE	Negative Case: Test checks to see if adding same person to the album fails.	Feb. 2014
Facial Recognition	fp.add.person.INVALID_FACE_INDEX	Negative Case: Test checks to see if inputing an invalid (integer value less then 0 and greater then or equal the number of faces detected) face index fails.	Feb. 2014
Facial Recognition	fp.add.person.NO_SET_FRAME	Negative Case: Test checks to see if adding a person without doing "setFrame" or "setBitmap" fails	Feb. 2014
Facial Recognition	fp.update.person.VALID_ENTRY	Base Case: Test checks if the person that has been previously added in the album can be updated to its corresponding PersonId	Feb. 2014
Facial Recognition	fp.update.person.NO_FACE	Negative Case: Test checks to see if updating a frame with no faces, fails.	Feb. 2014
Facial Recognition	fp.update.person.INVALID_FACE_INDEX	Negative Case: Test checks to see if inputing an invalid (integer value less then 0 and greater then or equal the number of faces detected) face index fails.	Feb. 2014
Facial Recognition	fp.update.person.INVALID_PERSON_ID(1)	Negative Case: Test checks to see if updating a person with PersonId less then 0, fails.	Feb. 2014
Facial Recognition	fp.update.person.INVALID_PERSON_ID(2)	Negative Case: Test checks to see if updating a person with an non existing PersonId fails.	Feb. 2014
Facial Recognition	fp.update.person.NO_SET_FRAME	Negative Case: Test checks to see if updating a person without doing setFrame/setBitmap, fails.	Feb. 2014
Facial Recognition	fp.delete.person.VALID_PERSON_ID	Base Case: Test checks to see if deleting a valid PersonId is successful	Feb. 2014

Facial Recognition	fp.delete.person.INVALID_PERSON_ID(1)	Negative Case: Test checks to see if deleting a person with PersonId less then 0, fails.	Feb. 2014
Facial Recognition	fp.delete.person.INVALID_PERSON_ID(2)	Negative Case: Test checks to see if deleting a person with a non existing PersonId fails.	Feb. 2014
Facial Recognition	fp.reset.album.VALID_ALBUM	Base Case: Test checks if reseting a valid album is successful.	Feb. 2014
Facial Recognition	fp.reset.album.INVALID_ALBUM	Negative Case: Test checks if reseting a non existent album fails.	Feb. 2014
Facial Recognition	fp.serialize.album.VALID_ALBUM	Base Case: Test checks if the album is successfully converted to a byte array	Feb. 2014
Facial Recognition	fp.serialize.album.INVALID_ALBUM	Negative Case: Test checks if serializing a non existent album fails.	Feb. 2014
Facial Recognition	fp.deserialize.album	Base Case: test checks if deserializing a valid byte array is successful	Feb. 2014
Facial Recognition	fp.deserialize.album.NULL_BUFFER	Negative Case: Test checks if deserializing NULL, fails.	Feb. 2014
Facial Recognition	fp.deserialize.album.CORRUPTED_BUFFER	Negative Case: Test checks if deserializing corrupted buffer, fails.	Feb. 2014
Facial Recognition	fp.get.face.data.IDENTIFY.EXISTING_FACE	Base Case: Test checks if a valid PersonId and a valid Confidence value is returned	Feb. 2014
Facial Recognition	fp.get.face.data.IDENTIFY.NO_EXISTING_FACE	Negative Case: Test checks if invalid PersonId and an invalid Confidence value is returned	Feb. 2014
Facial Recognition	fp.set.confidence	Base Case: Test checks to see if the confidence value of the framework is set to a desired value.	Feb. 2014
Facial Recognition	fp.set.confidence.LESS_THAN_ONE	Negative Case: Test checks to see if the argument for confidence value is not less than or equal to zero	Feb. 2014
Facial Recognition	fp.set.confidence.GREATER_THAN_NINTY_NINE	Negative Case: Test checks to see if the argument for the confidence value is not greater than 99	Feb. 2014
Device Info	is.isCriteriaPositioning	Base Case: Test checks to see if the feature FEATURE_OPTIMAL_CRITERIA_POSITIONING is enabled	Feb. 2014

4 Bugs

4.1 Known Bugs

No content for this release

Feature SDK	Test Name	Tracking ID	Description

4.2 Fixed Bugs

No content for this release

Feature SDK	Test Name	Tracking ID	Description

4.3 Withdrawn Bugs

No content for this release

Feature SDK	Test Name	Tracking ID	Description

5 Platform Limitations

No content for this release

CR	Family	Test Name	Description	PL

6 Cumulative Feature Descriptions

Feature SDK	Test Name	Description	Release
Facial Recognition	fp.add.person.VALID_ENTRY	Base Case: Test checks if the person in the frame is added successfully.	Feb. 2014
Facial Recognition	fp.add.person.NO_FACE	Negative Case: Test checks to see if adding a frame with no faces, fails	Feb. 2014
Facial Recognition	fp.add.person.ADD_SAME_FACE	Negative Case: Test checks to see if adding same person to the album fails.	Feb. 2014
Facial Recognition	fp.add.person.INVALID_FACE_INDEX	Negative Case: Test checks to see if inputing an invalid (integer value less then 0 and greater then or equal the number of faces detected) face index fails.	Feb. 2014
Facial Recognition	fp.add.person.NO_SET_FRAME	Negative Case: Test checks to see if adding a person without doing "setFrame" or "setBitmap" fails	Feb. 2014
Facial Recognition	fp.update.person.VALID_ENTRY	Base Case: Test checks if the person that has been previously added in the album can be updated to its corresponding PersonId	Feb. 2014
Facial Recognition	fp.update.person.NO_FACE	Negative Case: Test checks to see if updating a frame with no faces, fails.	Feb. 2014
Facial Recognition	fp.update.person.INVALID_FACE_INDEX	Negative Case: Test checks to see if inputing an invalid (integer value less then 0 and greater then or equal the number of faces detected) face index fails.	Feb. 2014
Facial Recognition	fp.update.person.INVALID_PERSON_ID(1)	Negative Case: Test checks to see if updating a person with PersonId less then 0, fails.	Feb. 2014
Facial Recognition	fp.update.person.INVALID_PERSON_ID(2)	Negative Case: Test checks to see if updating a person with an non existing PersonId fails.	Feb. 2014
Facial Recognition	fp.update.person.NO_SET_FRAME	Negative Case: Test checks to see if updating a person without doing setFrame/setBitmap, fails.	Feb. 2014
Facial Recognition	fp.delete.person.VALID_PERSON_ID	Base Case: Test checks to see if deleting a valid PersonId is successful	Feb. 2014
Facial Recognition	fp.delete.person.INVALID_PERSON_ID(1)	Negative Case: Test checks to see if deleting a person with PersonId less then 0, fails.	Feb. 2014

Facial Recognition	fp.delete.person.INVALID_PERSON_ID(2)	Negative Case: Test checks to see if deleting a person with a non existing PersonId fails.	Feb. 2014
Facial Recognition	fp.reset.album.VALID_ALBUM	Base Case: Test checks if reseting a valid album is successful.	Feb. 2014
Facial Recognition	fp.reset.album.INVALID_ALBUM	Negative Case: Test checks if reseting a non existent album fails.	Feb. 2014
Facial Recognition	fp.serialize.album.VALID_ALBUM	Base Case: Test checks if the album is successfully converted to a byte array	Feb. 2014
Facial Recognition	fp.serialize.album.INVALID_ALBUM	Negative Case: Test checks if serializing a non existent album fails.	Feb. 2014
Facial Recognition	fp.deserialize.album	Base Case: test checks if deserializing a valid byte array is successful	Feb. 2014
Facial Recognition	fp.deserialize.album.NULL_BUFFER	Negative Case: Test checks if deserializing NULL, fails.	Feb. 2014
Facial Recognition	fp.deserialize.album.CORRUPTED_BUFFER	Negative Case: Test checks if deserializing corrupted buffer, fails.	Feb. 2014
Facial Recognition	fp.get.face.data.IDENTIFY.EXISTING_FACE	Base Case: Test checks if a valid PersonId and a valid Confidence value is returned	Feb. 2014
Facial Recognition	fp.get.face.data.IDENTIFY.NO_EXISTING_FACE	Negative Case: Test checks if invalid PersonId and an invalid Confidence value is returned	Feb. 2014
Facial Recognition	fp.set.confidence	Base Case: Test checks to see if the confidence value of the framework is set to a desired value.	Feb. 2014
Facial Recognition	fp.set.confidence.LESS_THAN_ONE	Negative Case: Test checks to see if the argument for confidence value is not less than or equal to zero	Feb. 2014
Facial Recognition	fp.set.confidence.GREATER_THAN_NINTY_NINE	Negative Case: Test checks to see if the argument for the confidence value is not greater than 99	Feb. 2014
Touch-Free Gestures	cg.overlay.preferences.6	Test calls OverlayPreferences(boolean enabled, int flags) constructor with enabled set to true and flags set to "UP | DOWN | LEFT | RIGHT", test succeeds if the overlay object is instantiated	Aug. 2013
Touch-Free Gestures	cg.update.preferences.1	Test sets up an overlay preferences by calling the constructor OverlayPreference(false, 0) which doesn't enable the overlay or set any flags. Then the test will call the update(boolean enabled, int flags) method to enable the overlay (true) and set the flag bit mask (LEFT | RIGHT)	Aug. 2013
Touch-Free Gestures	cg.update.preferences.2	Test sets up an overlay preferences by calling the constructor OverlayPreference(false, 0) which doesn't enable the overlay or set any flags. Then the test will call the update(boolean enabled, int flags) method to enable the overlay (true) and set	Aug. 2013

		the flag bit mask (UP | DOWN | LEFT | RIGHT)	
Touch-Free Gestures	cg.pointer	Tests to make sure a pointer event can be detected. If pointer is detected then the test passes	Aug. 2013
Touch-Free Gestures	cg.near.left_swipe	Tests to make sure a near left swipe can be detected. If a near left swipe event is detected then the test passes	Aug. 2013
Touch-Free Gestures	cg.near.right_swipe	Tests to make sure a near right swipe is detected. If a near right swipe event is detected then the test passes	Aug. 2013
Touch-Free Gestures	cg.engagement.left_swipe	Tests to make sure a left swipe engagement event occurs. If the event occurs then the test passes.	Aug. 2013
Touch-Free Gestures	cg.engagement.right_swipe	Tests to make sure a right swipe engagement event occurs. If the event occurs then the test passes.	Aug. 2013
Touch-Free Gestures	cg.engagement.up_swipe	Tests to make sure an up swipe engagement event occurs. If the event occurs then the test passes.	Aug. 2013
Touch-Free Gestures	cg.engagement.down_swipe	Tests to make sure a down swipe engagement event occurs. If the event occurs then the test passes.	Aug. 2013
Touch-Free Gestures	cg.engagement.engagement_detection	Tests to make sure a hand engagement event is detected. If the event is detected then the test passes.	Aug. 2013
Facial Processing	fp.is.feature.supported.FACAL_PROCESSING	Base case: test checks to see if facial processing functionality is available on the device	Aug. 2013
Facial Processing	fp.object.getInstance.instance_1	Base case: test checks if a valid instance of a facial processing object can be created on the device	Aug. 2013
Facial Processing	fp.object.getInstance.instance_2_pre_release	Test checks that a second call for getInstance() before release() of first returns null.	Aug. 2013
Facial Processing	fp.object.getInstance.instance_2_post_release	Test checks that a second call for getInstance() after release() of first returns valid object.	Aug. 2013
Facial Processing	fp.set.processing.mode.video	Base case: test checks if the framework can be configured for video mode	Aug. 2013
Facial Processing	fp.set.processing.mode.still	Base case: test checks if the framework can be configured for still mode	Aug. 2013
Facial Processing	fp.set.frame.invalidimg	Negative case: test checks if an image can be successfully processed and return detected faces from a invalid static image	Aug. 2013
Facial Processing	fp.set.frame.validImgInvalidDimension	Negative case: test checks if an image can be successfully processed and return detected faces from a valid static image with invalid dimensions	Aug. 2013

Facial Processing	fp.get.num.faces.01	Base case: test to check that the framework can identify the correct number of faces in a known image	Aug. 2013
Facial Processing	fp.get.num.faces.02	Negative case: test to check that the framework returns zero in an image that has no faces	Aug. 2013
Facial Processing	fp.get.all.face.data.01	Base case: test to check that the framework can identify and return face objects for a known image with all facial feature data	Aug. 2013
Facial Processing	fp.get.all.face.data.02	Negative case: test to check that the framework will return an empty array if getFaceData() is called on an image containing no faces	Aug. 2013
Facial Processing	fp.get.subset.face.data.01	Base case: test to check that the framework can identify and return face objects for a known image with a subset of the facial feature data using the getFaceData(int) variant	Aug. 2013
Facial Processing	fp.get.subset.face.data.02	Negative case: test to check that the framework will return an empty array if getFaceData(int) is called on an image containing no faces	Aug. 2013
Facial Processing	fp.clone.object	Test to make sure that the facial processing object is not clonable	Aug. 2013
Facial Processing	fd.get.left.eye.blink	Base case: check that left eye blink value is between 0 to 100	Aug. 2013
Facial Processing	fd.get.left.eye.blink.01	Base case: checks that blink in with an extreme open value can be detected	Aug. 2013
Facial Processing	fd.get.left.eye.blink.02	Base case: checks that blink in with an extreme closed value can be detected	Aug. 2013
Facial Processing	fd.get.left.eye.blink.03	Negative case: checks that an image that had getFaceData called without the bit set for blink detection, returns a value of NOT_PROCESSED (-999) when inspected for data	Aug. 2013
Facial Processing	fd.get.right.eye.blink	Base case: check that right eye blink value is between 0 to 100	Aug. 2013
Facial Processing	fd.get.right.eye.blink.01	Base case: checks that blink in with an extreme open value can be detected	Aug. 2013
Facial Processing	fd.get.right.eye.blink.02	Base case: checks that blink in with an extreme closed value can be detected	Aug. 2013
Facial Processing	fd.get.right.eye.blink.03	Negative case: checks that an image that had getFaceData called without the bit set for blink detection, returns a value of NOT_PROCESSED (-999) when inspected for data	Aug. 2013
Facial Processing	fd.get.eye.horizontal.gaze.angle	Base case: checks that hotizontal gaze angle is	Aug. 2013

		between -30 to 30	
Facial Processing	fd.get.eye.horizontal.gaze.angle.01	Base case: checks that hotizontal gaze in the extreme left direction can be detected	Aug. 2013
Facial Processing	fd.get.eye.horizontal.gaze.angle.02	Base case: checks that a horizontal gaze value close to zero (center) can be detected	Aug. 2013
Facial Processing	fd.get.eye.horizontal.gaze.angle.03	Base case: checks that horizontal gaze in the extreme right direction can be detected	Aug. 2013
Facial Processing	fd.get.eye.horizontal.gaze.angle.04	Negative case: checks that an image that had getFaceData called without the bit set for gaze angle, returns a value of NOT_PROCESSED (-999) when inspected for horizontal gaze data	Aug. 2013
Facial Processing	fd.get.eye.vertical.gaze.angle	Base case: check that vertical gaze value is between -20 to 20	Aug. 2013
Facial Processing	fd.get.eye.vertical.gaze.angle.01	Base case: checks that vertical gaze in the extreme up direction can be detected	Aug. 2013
Facial Processing	fd.get.eye.vertical.gaze.angle.02	Base case: checks that a vertical gaze value close to zero (center) can be detected	Aug. 2013
Facial Processing	fd.get.eye.vertical.gaze.angle.03	Base case: checks that vertical gaze in the extreme down direction can be detected	Aug. 2013
Facial Processing	fd.get.eye.vertical.gaze.angle.04	Negative case: checks that an image that had getFaceData called without the bit set for gaze angle, returns a value of NOT_PROCESSED (-999) when inspected for vertical gaze data	Aug. 2013
Facial Processing	fd.get.eye.gaze.point	Base case: checks that gaze point is returned between -1.0 and 1.0	Aug. 2013
Facial Processing	fd.get.eye.gaze.point.01	Base case: checks that gazing at a point in the extreme upper-left direction can be detected	Aug. 2013
Facial Processing	fd.get.eye.gaze.point.02	Base case: checks that gazing at a point in the extreme upper-right direction can be detected	Aug. 2013
Facial Processing	fd.get.eye.gaze.point.03	Base case: checks that gazing at a point in the extreme lower-left direction can be detected	Aug. 2013
Facial Processing	fd.get.eye.gaze.point.04	Base case: checks that gazing at a point in the extreme lower-right direction can be detected	Aug. 2013
Facial Processing	fd.get.eye.gaze.point.05	Base case: checks that gazing at a point in the center direction can be detected	Aug. 2013
Facial Processing	fd.get.eye.gaze.point.06	Negative case: checks that an image that had getFaceData called without the bit set for gaze angle, returns a value of NOT_PROCESSED (NULL) when inspected for gaze point data	Aug. 2013
Facial Processing	fd.get.yaw	Base case: check to see if yaw returned is between -180 and 179	Aug. 2013

Facial Processing	fd.get.yaw.01	Base case: checks that yaw in the extreme left direction can be detected	Aug. 2013
Facial Processing	fd.get.yaw.02	Base case: checks that a yaw value close to zero can be detected	Aug. 2013
Facial Processing	fd.get.yaw.03	Base case: checks that yaw in the extreme right direction can be detected	Aug. 2013
Facial Processing	fd.get.yaw.04	Negative case: checks that an image that had getFaceData called without the bit set for yaw, returns a value of NOT_PROCESSED (-999) when inspected for data	Aug. 2013
Facial Processing	fd.get.roll	Base case: check to see if roll returned is between -180 and 179	Aug. 2013
Facial Processing	fd.get.roll.01	Base case: checks that roll in the extreme left direction can be detected	Aug. 2013
Facial Processing	fd.get.roll.02	Base case: checks that a roll value close to zero can be detected	Aug. 2013
Facial Processing	fd.get.roll.03	Base case: checks that roll in the extreme right direction can be detected	Aug. 2013
Facial Processing	fd.get.roll.04	Negative case: checks that an image that had getFaceData called without the bit set for roll, returns a value of NOT_PROCESSED (-999) when inspected for data	Aug. 2013
Facial Processing	fd.get.pitch	Base case: check to see if pitch returned is between -180 and 179	Aug. 2013
Facial Processing	fd.get.pitch.01	Base case: checks that pitch in the extreme upward direction can be detected	Aug. 2013
Facial Processing	fd.get.pitch.02	Base case: checks that a pitch value close to zero can be detected (head not tilted up or down)	Aug. 2013
Facial Processing	fd.get.pitch.03	Base case: checks that pitch in the extreme downward direction can be detected	Aug. 2013
Facial Processing	fd.get.pitch.04	Negative case: checks that an image that had getFaceData called without the bit set for pitch, returns a value of NOT_PROCESSED (-999) when inspected for data	Aug. 2013
Facial Processing	fd.get.smile	Base case: check that the smile value returned is between 0 and 100	Aug. 2013
Facial Processing	fd.get.smile.01	Base case: checks that an image containing a highly probable smile (person grinning) can be detected	Aug. 2013
Facial Processing	fd.get.smile.02	Base case: checks that an image containing a highly improbable smile (person frowning) can be	Aug. 2013

		detected	
Facial Processing	fd.get.smile.03	Negative case: checks that an image that had getFaceData called without the bit set for smile detection, returns a value of NOT_PROCESSED (-999) when inspected for data	Aug. 2013
Facial Processing	fd.left.eye.point.01	Base case: checks that the left eye location in an image containing a face can be detected	Aug. 2013
Facial Processing	fd.left.eye.point.02	Negative case: checks that an image that had getFaceData called without the bit set for face coordinates, returns a value of NOT_PROCESSED (NULL) when inspected for data	Aug. 2013
Facial Processing	fd.right.eye.point.01	Base case: checks that the right eye location in an image containing a face can be detected	Aug. 2013
Facial Processing	fd.right.eye.point.02	Negative case: checks that an image that had getFaceData called without the bit set for face coordinates, returns a value of NOT_PROCESSED (NULL) when inspected for data	Aug. 2013
Facial Processing	fd.right.mouth.point.01	Base case: checks that the mouth location in an image containing a face can be detected	Aug. 2013
Facial Processing	fd.right.mouth.point.02	Negative case: checks that an image that had getFaceData called without the bit set for face coordinates, returns a value of NOT_PROCESSED (NULL) when inspected for data	Aug. 2013
Facial Processing	fd.face.rect.01	Base case: checks that a face rect in an image containing a face can be detected	Aug. 2013
Facial Processing	fd.face.rect.02	Base case: checks that a face rect in an image containing a face can be detected	Aug. 2013
Facial Processing	fp.setBitmap.01	checks that sending a bitmap with known number of faces returns the correct number of faces	Aug. 2013
Facial Processing	fp.setBitmap.02	checks that passing a bitmap with known number of faces returns a non-null non-empty array of facedata.	Aug. 2013
Device Info	di.is.snapdragon.01	Base case: test checks to see if isSnapdragon() returns true in the event that both possible files exist on the device	Aug. 2013
Device Info	di.is.snapdragon.02	Base case: test checks to see if isSnapdragon() returns true in the event that only one of the files exist on the device	Aug. 2013
Device Info	di.is.snapdragon.03	Negative case: test checks that isSnapdragon() returns false on devices that don't have any of the validation files	Aug. 2013
Device Info	di.is.feature.supported.indoor.position.01	Base case: test if isFeatureSupported returns true	Aug. 2013

		for indoor positioning if the QUIPC_ENABLED key is set to true in gps.conf	
Device Info	di.is.feature.supported.indoor.position.02	Base case: test that isFeatureSupported returns true for indoor positioning if the QUIPC_ENABLED key is set to true in izat.conf (implies it didn't exist in gps.conf)	Aug. 2013
Device Info	di.is.feature.supported.indoor.position.03	Negative case: test that isFeatureSupported returns false for indoor positioning if the QUIPC_ENABLED key is set to false in gps.conf	Aug. 2013
Device Info	di.is.feature.supported.indoor.position.04	Base case: test that isFeatureSupported returns false for indoor positioning the QUIPC_ENABLED key is set to false in izat.conf (implies it didn't exist in gps.conf)	Aug. 2013
Device Info	di.is.feature.supported.low.power.geofence.01	Base case: test that isFeatureSupported returns true for low-power geofencing if the Qualcomm location service is installed and accessible through PackageManager	Aug. 2013
Device Info	di.is.feature.supported.low.power.geofence.02	Negative case: test that isFeatureSupported returns false for low-power geofencing if the Qualcomm location service is not installed and accessible through PackageManager	Aug. 2013
Device Info	di.is.feature.supported.surround.sound.recording.01	Base case: test that isFeatureSupported returns true for Surround Sound Recording on a device that has Fluence enabled	Aug. 2013
Device Info	di.is.feature.supported.surround.sound.recording.02	Base case: test that isFeatureSupported returns true for Surround Sound Recording on a device that has Fluence Pro enabled	Aug. 2013
Device Info	di.is.feature.supported.surround.sound.recording.03	Negative case: test that isFeatureSupported returns false for Surround Sound Recording on a device that has neither Fluence nor Fluence Pro enabled	Aug. 2013
Device Info	di.is.feature.supported.echo.cancellation.01	Base case: test isFeatureSupported returns true for Echo Cancellation on a device that has Fluence enabled	Aug. 2013
Device Info	di.is.feature.supported.echo.cancellation.02	Base case: test isFeatureSupported returns true forEcho Cancellation on a device that has Fluence Pro enabled	Aug. 2013
Device Info	di.is.feature.supported.echo.cancellation.03	Negative case: test isFeatureSupported returns true for Echo Cancellation on a device that has neither Fluence nor Fluence Pro enabled	Aug. 2013
Device Info	is.isCriteriaPositioning	Base Case: Test checks to see if the feature FEATURE_OPTIMAL_CRITERIA_POSITIONING is enabled	Feb. 2014

1