

<https://developer.mozilla.org/en-US/docs/Web/API/Touch>

The **Touch** interface represents a single contact point on a touch-sensitive device. The contact point is commonly a finger or stylus and the device may be a touchscreen or trackpad.

The [Touch.radiusX](#), Touch.radiusY, and Touch.rotationAngle describe the area of contact between the user and the screen, the *touch area*. This can be helpful when dealing with imprecise pointing devices such as fingers. These values are set to describe an ellipse that as closely as possible matches the entire area of contact (such as the user's fingertip).

**Note:** Many of the properties' values are hardware-dependent; for example, if the device doesn't have a way to detect the amount of pressure placed on the surface, the force value will always be 0. This may also be the case for radiusX and radiusY; if the hardware reports only a single point, these values will be 1.

## Constructor

---

### [Touch\(\)](#)

Creates a Touch object.

#### **This is an experimental technology**

Because this technology's specification has not stabilized, check the [compatibility table](#) for usage in various browsers. Also note that the syntax and behavior of an experimental technology is subject to change in future versions of browsers as the specification changes.

The **Touch()** constructor creates a new [Touch](#) object.

# Syntax

```
touch = new Touch(touchInit);
```

## Arguments

### *touchInit*

Is a `TouchInit` dictionary, having the following fields:

- "identifier", required, of type `long`, that is the identification number for the touch point.
- "target", required, of type [EventTarget](#), the item at which the touch point started when it was first placed on the surface.
- "clientX", optional and defaulting to 0, of type `double`, that is the horizontal position of the touch on the client window of user's screen, excluding any scroll offset.
- "clientY", optional and defaulting to 0, of type `double`, that is the vertical position of the touch on the client window of the user's screen, excluding any scroll offset.
- "screenX", optional and defaulting to 0, of type `double`, that is the horizontal position of the touch on the user's screen.
- "screenY", optional and defaulting to 0, of type `double`, that is the vertical position of the touch on the user's screen.
- "pageX", optional and defaulting to 0, of type `double`, that is the horizontal position of the touch on the client window of user's screen, including any scroll offset.
- "pageY", optional and defaulting to 0, of type `double`, that is the vertical position of the touch on the client window of the user's screen, including any scroll offset.
- "radiusX", optional and defaulting to 0, of type `float`, that is the radius of the ellipse which most closely circumscribes the touching area (e.g. finger, stylus) along the axis indicated by `rotationAngle`, in CSS pixels of the same scale as `screenX`; 0 if no value is known. The value must not be negative.
- "radiusY", optional and defaulting to 0, of type `float`, that is the radius of the ellipse which most closely circumscribes the touching area (e.g. finger, stylus) along the axis perpendicular to that indicated by `rotationAngle`, in CSS pixels of the same scale as `screenY`; 0 if no value is known. The value must not be negative.
- "rotationAngle", optional and defaulting to 0, of type `float`, that is the angle (in degrees) that the ellipse described by `radiusX` and `radiusY` is rotated clockwise about its center; 0 if no value is known. The value must be greater than or equal to 0 and less than 90. If the ellipse described by `radiusX` and `radiusY` is circular, then `rotationAngle` has no effect. The user agent may use 0 as the value in this case, or it may use any other value in the allowed range. (For example, the user agent may use the `rotationAngle` value from the previous touch event, to avoid sudden changes.)
- "force", optional and defaulting to 0, of type `float`, that is the relative value of pressure applied, in the range 0 to 1, where 0 is no pressure, and 1 is the highest level of pressure the touch device is capable of sensing; 0 if no value is known. In environments where force is known, the absolute pressure represented by the `force` attribute, and the sensitivity in levels of pressure, may vary.

## Specifications

Specification	Status	Comment
<a href="#">Touch Events – Level 2</a> <a href="#">The definition of 'TouchEvent' in that specification.</a>	Editor's Draft	Initial definition.

## Browser compatibility

- Desktop
- Mobile

Feature	Chrome	Firefox (Gecko)	Internet Explorer	Opera	Safari (WebKit)
Basic support	48.0	?	?	35	?

# Properties

---

---

*This interface has no parent, and doesn't inherits or implements any other property.*

# Touch.identifier

---

[Touch.identifier](#) Read only

Returns a unique identifier for this Touch object. A given touch point (say, by a finger) will have the same identifier for the duration of its movement around the surface. This lets you ensure that you're tracking the same touch all the time.

The **Touch.identifier** returns a value uniquely identifying this point of contact with the touch surface. This value remains consistent for every event involving this finger's (or stylus's) movement on the surface until it is lifted off the surface.

## Syntax

```
touchItem.identifier;
```

## Return value

A [DOMString](#) that represents the unique ID of the Touch object.

## Example

```
someElement.addEventListener('touchmove', function(e) {  
  // Iterate through the list of touch points that changed  
  // since the last event and print each touch point's identifier.  
  for (var i=0; i < e.changedTouches.length; i++) {  
    console.log("changedTouches[" + i + "].identifier = " + e.changedTouches[i].identifier);  
  }  
}, false);
```

## Specifications

Specification	Status	Comment
<a href="#">Touch Events – Level 2</a>	Editor's Draft	No change.
<a href="#">Touch Events</a>	Recommendation	Initial definition.

## Browser compatibility

- Desktop
- Mobile

Feature	Chrome	Firefox (Gecko)	Internet Explorer	Opera	Safari (WebKit)
Basic support	22.0	<a href="#">18.0</a> (18.0)	Not supported	Not supported	Not supported

# Touch.screenX

---

[Touch.screenX](#) Read only

Returns the X coordinate of the touch point relative to the left edge of the screen.

## Summary

Returns the X coordinate of the touch point relative to the screen, not including any scroll offset.

## Syntax

```
var x = touchItem.screenX;
```

## Return value

x

The X coordinate of the touch point relative to the screen, not including any scroll offset.

## Example

This example illustrates how to access the [Touch](#) object's Touch.screenX and Touch.screenY properties. The Touch.screenX property is the horizontal (x) coordinate of a touch point relative to the screen in CSS pixels. The Touch.screenY property is the vertical coordinate of a touch point relative to the screen in CSS pixels.

In following simple code snippet, we assume the user initiates multiple touch contacts on an element with an id of source and then releases contacts with the surface. When the [touchstart](#) event handler is invoked, each touch point's Touch.screenX and Touch.screenY coordinates are accessed.

```
// Register a touchstart listeners for the 'source' element
var src = document.getElementById("source");

src.addEventListener('touchstart', function(e) {
  // Iterate through the touch points and log each screenX/Y coordinate.
  // The unit of each coordinate is CSS pixels.
  var i;
  for (i=0; i < e.touches.length; i++) {
    console.log("touchpoint[" + i + "].screenX = " + e.touches[i].screenX);
    console.log("touchpoint[" + i + "].screenY = " + e.touches[i].screenY);
  }
}, false);
```

## Specifications

Specification	Status	Comment
<a href="#">Touch Events – Level 2</a>	Editor's Draft	Non-stable version.
<a href="#">Touch Events</a>	Recommendation	Initial definition.

## Browser compatibility

- Desktop
- Mobile

Feature	Chrome	Firefox (Gecko)	Internet Explorer	Opera	Safari (WebKit)
Basic support	22.0	<a href="#">18.0</a> (18.0)	Not supported	Not supported	Not supported

# Touch.screenY

---

[Touch.screenY](#) Read only

Returns the Y coordinate of the touch point relative to the top edge of the screen.

## Summary

Returns the Y coordinate of the touch point relative to the screen, not including any scroll offset.

## Syntax

```
var y = touchItem.screenY;
```

## Return value

y

The Y coordinate of the touch point relative to the screen, not including any scroll offset.

## Example

The [Touch.screenX example](#) includes an example of this property's usage.

## Specifications

Specification	Status	Comment
<a href="#">Touch Events – Level 2</a>	Editor's Draft	Non-stable version.
<a href="#">Touch Events</a>	Recommendation	Initial definition.

## Browser compatibility

- Desktop
- Mobile

Feature	Chrome	Firefox (Gecko)	Internet Explorer	Opera	Safari (WebKit)
Basic support	22.0	<a href="#">18.0</a> (18.0)	Not supported	Not supported	Not supported

# Touch.clientX

---

[Touch.clientX](#) Read only

Returns the X coordinate of the touch point relative to the left edge of the browser viewport, not including any scroll offset.

The Touch.clientX read-only property returns the X coordinate of the touch point relative to the viewport, not including any scroll offset.

## Syntax

```
touchItem.clientX;
```

## Return value

A long representing the X coordinate of the touch point relative to the viewport, not including any scroll offset.

## Example

This example illustrates using the [Touch](#) object's Touch.clientX and Touch.clientY properties. The Touch.clientX property is the horizontal coordinate of a touch point relative to the browser's viewport excluding any scroll offset. The Touch.clientY property is the vertical coordinate of the touch point relative to the browser's viewport excluding any scroll offset.

In this example, we assume the user initiates a touch on an element with an id of source, moves within the element or out of the element and then releases contact with the surface. When the [touchend](#) event handler is invoked, the changes in the Touch.clientX and Touch.clientY coordinates, from the starting touch point to the ending touch point, are calculated.

```
// Register touchstart and touchend listeners for element 'source'
var src = document.getElementById("source");
var clientX, clientY;

src.addEventListener('touchstart', function(e) {
    // Cache the client X/Y coordinates
    clientX = e.touches[0].clientX;
    clientY = e.touches[0].clientY;
}, false);

src.addEventListener('touchend', function(e) {
    var deltaX, deltaY;

    // Compute the change in X and Y coordinates.
    // The first touch point in the changedTouches
    // list is the touch point that was just removed from the surface.
    deltaX = e.changedTouches[0].clientX - clientX;
    deltaY = e.changedTouches[0].clientY - clientY;

    // Process the data ...
}, false);
```

## Specifications

Specification	Status	Comment
<a href="#">Touch Events – Level 2</a>	Editor's Draft	Non-stable version.
<a href="#">Touch Events</a>	Recommendation	Initial definition.

## Browser compatibility

- Desktop
- Mobile

Feature	Chrome	Firefox (Gecko)	Internet Explorer	Opera	Safari (WebKit)
Basic support	22.0	<a href="#">18.0</a> (18.0)	Not supported	Not supported	Not supported



# Touch.clientY

---

[Touch.clientY](#) Read only

Returns the Y coordinate of the touch point relative to the top edge of the browser viewport, not including any scroll offset.

The **Touch.clientY** read-only property returns the Y coordinate of the touch point relative to the browser's viewport, not including any scroll offset.

## Syntax

```
touchItem.clientY;
```

## Return value

A long value representing the Y coordinate of the touch point relative to the viewport, not including any scroll offset.

## Example

This example illustrates using the [Touch](#) object's Touch.clientX and Touch.clientY properties. The Touch.clientX property is the horizontal coordinate of a touch point relative to the browser's viewport excluding any scroll offset. The Touch.clientY property is the vertical coordinate of the touch point relative to the browser's viewport excluding any scroll offset.

In this example, we assume the user initiates a touch on an element with an id of source, moves within the element or out of the element and then releases contact with the surface. When the [touchend](#) event handler is invoked, the changes in the Touch.clientX and Touch.clientY coordinates, from the starting touch point to the ending touch point, are calculated.

```
// Register touchstart and touchend listeners for element 'source'
var src = document.getElementById("source");
var clientX, clientY;

src.addEventListener('touchstart', function(e) {
    // Cache the client X/Y coordinates
    clientX = e.touches[0].clientX;
    clientY = e.touches[0].clientY;
}, false);

src.addEventListener('touchend', function(e) {
    var deltaX, deltaY;

    // Compute the change in X and Y coordinates.
    // The first touch point in the changedTouches
    // list is the touch point that was just removed from the surface.
    deltaX = e.changedTouches[0].clientX - clientX;
    deltaY = e.changedTouches[0].clientY - clientY;

    // Process the data ...
}, false);
```

## Specifications

Specification	Status	Comment
<a href="#">Touch Events – Level 2</a>	Editor's Draft	No changes since last version.
<a href="#">Touch Events</a>	Recommendation	Initial definition.

## Browser compatibility

- Desktop
- Mobile

Feature	Chrome	Firefox (Gecko)	Internet Explorer	Opera	Safari (WebKit)
Basic support	22.0	<a href="#">18.0</a> (18.0)	Not supported	Not supported	Not supported

# Touch.pageX

---

[Touch.pageX](#) Read only

Returns the X coordinate of the touch point relative to the left edge of the document. Unlike `clientX`, this value includes the horizontal scroll offset, if any.

The **Touch.pageX** read-only property returns the X coordinate of the touch point relative to the viewport, including any scroll offset.

## Syntax

```
touchItem.pageX;
```

## Return value

A long representing the X coordinate of the touch point relative to the viewport, including any scroll offset.

## Example

This example illustrates how to access the `Touch` object's `Touch.pageX` and `Touch.pageY` properties. The `Touch.pageX` property is the horizontal coordinate of a touch point relative to the viewport (in CSS pixels), including any scroll offset. The `Touch.pageY` property is the vertical coordinate of a touch point relative to the viewport (in CSS pixels), including any scroll offset.

In following simple code snippet, we assume the user initiates one or more touch contacts on the source element, moves the touch points and then releases all contacts with the surface. When the `touchmove` event handler is invoked, each touch point's `Touch.pageX` and `Touch.pageY` coordinates are accessed via the event's `TouchEvent.changedTouches` list.

```
// Register a touchmove listeners for the 'source' element
var src = document.getElementById("source");

src.addEventListener('touchmove', function(e) {
  // Iterate through the touch points that have moved and log each
  // of the pageX/Y coordinates. The unit of each coordinate is CSS pixels.
  var i;
  for (i=0; i < e.changedTouches.length; i++) {
    console.log("touchpoint[" + i + "].pageX = " + e.changedTouches[i].pageX);
    console.log("touchpoint[" + i + "].pageY = " + e.changedTouches[i].pageY);
  }
}, false);
```

## Specifications

Specification	Status	Comment
<a href="#">Touch Events – Level 2</a>	Editor's Draft	No change from the previous version.
<a href="#">Touch Events</a>	Recommendation	Initial definition.

Browser compatibility

Desktop

Mobile

--

Status

Comment

# Touch.pageY

---

[Touch.pageY](#) Read only

Returns the Y coordinate of the touch point relative to the top of the document. Unlike `clientY`, this value includes the vertical scroll offset, if any.

The **`Touch.pageY`** read-only property returns the Y coordinate of the touch point relative to the viewport, including any scroll offset.

## Syntax

```
touchItem.pageY;
```

## Return value

A long value that represents the Y coordinate of the touch point relative to the viewport, including any scroll offset.

## Example

This example illustrates how to access the `Touch` object's `Touch.pageX` and `Touch.pageY` properties. The `Touch.pageX` property is the horizontal coordinate of a touch point relative to the viewport (in CSS pixels), including any scroll offset. The `Touch.pageY` property is the vertical coordinate of a touch point relative to the viewport (in CSS pixels), including any scroll offset.

In following simple code snippet, we assume the user initiates one or more touch contacts on the source element, moves the touch points and then releases all contacts with the surface.

When the `touchmove` event handler is invoked, each touch point's `Touch.pageX` and `Touch.pageY` coordinates are accessed via the event's `TouchEvent.changedTouches` list.

```
// Register a touchmove
listeners for the 'source' element
var src = document.getElementById("source");

src.addEventListener('touchmove', function(e) {
    // Iterate through the
touch points that have
moved and log each
    // of the pageX/Y coordinates. The unit of each
coordinate is CSS pixels.
    var i;
    for (i=0; i < e.changedTouches.length; i++) {
        console.log("touch-
```

```

point[" + i + "].pageX =
" + e.changedTouches[i].pageX);
    console.log("touch-
point[" + i + "].pageY =
" + e.changedTouches[i].pageY);
    }
}, false);

```

## Specifications

Specification

<del>Browser compatibility</del>	Editor's Draft	No change from last version.
<del>Touch Events - Level 2</del>		
Touch Events	Recommendation	Initial definition.
Desktop		
Mobile		

	Status	Comment
--	--------	---------

## Touch.target

-----

[Touch.target](#) Read only  
Returns the [Element](#) on which the touch point started when it was first placed on the surface, even if the touch point has since moved outside the interactive area of that element or even been removed from the document.

## Summary

Returns the [Element](#) ([EventTarget](#)) on which the touch contact started when it was first placed on the surface, even if the touch point has since moved outside the interactive area of that element or even been removed from the document. Note that if the target element is removed from the document, events will still be targeted at it, and hence won't necessarily bubble up to the window or document anymore. If there is any risk of an element being removed while it is being touched, the best practice is to attach the touch listeners directly to the target.

## Syntax

```

var el = touchPoint.target;

```

## Return value

el

The target element of the [Touch](#) object.

## Example

This example illustrates how to access the [Touch](#) object's [Touch.target](#) property. The [Touch.target](#) property is an [Element](#) ([EventTarget](#)) on which a touch point is started when contact is first placed on the surface.

In following simple code snippet, we assume the user initiates one or more touch contacts on the source element. When the [touchstart](#) event handler for this element is invoked, each touch point's [Touch.target](#) property is accessed via the event's [TouchEvent.targetTouches](#) list.

```
// Register a touchmove listener for the 'source' element
var src = document.getElementById("source");

src.addEventListener('touchstart', function(e) {
    // Iterate through the touch points that were activated
    // for this element.
    for (var i=0; i < e.targetTouches.length; i++) {
        console.log("touchpoint[" + i + "].target = " + e.targetTouches[i].target);
    }
}, false);
```

## Specifications

Specification

[Browser compatibility](#)

[Touch Events](#)

Desktop

Mobile

Editor's Draft

Recommendation

Non-stable version.

Initial definition.

--

Status

Comment

# Touch.radiusX

---

[Touch.radiusX](#) Read only

Returns the X radius of the ellipse that most closely circumscribes the area of contact with the screen. The value is in pixels of the same scale as screenX.

## Summary

Returns the X radius of the ellipse that most closely circumscribes the area of contact with the touch surface. The value is in CSS pixels of the same scale as [Touch.screenX](#).

This value, in combination with [Touch.radiusY](#) and Touch.rotationAngle constructs an ellipse that approximates the size and shape of the area of contact between the user and the screen. This may be a relatively large ellipse representing the contact between a fingertip and the screen or a small area representing the tip of a stylus, for example.

**Note:** This attribute has *not* been formally standardized. It is specified in the [Touch Events – Level 2](#) Editor's Draft specification and not in Touch Events Recommendation. This attribute is not widely implemented.

## Syntax

```
var xRadius = touchItem.radiusX;
```

## Return value

xRadius

The X radius of the ellipse that most closely circumscribes the area of contact with the touch surface.

## Example

This example illustrates using the [Touch](#) interface's [Touch.radiusX](#), [Touch.radiusY](#) and [Touch.rotationAngle](#) properties. The [Touch.radiusX](#) property is the radius of the ellipse which most closely circumscribes the touching area (e.g. finger, stylus) along the axis **indicated** by the touch point's [Touch.rotationAngle](#). Likewise, the [Touch.radiusY](#) property is the radius of the ellipse which most closely circumscribes the touching area

(e.g. finger, stylus) along the axis **perpendicular** to that indicated by `Touch.rotationAngle`. The `Touch.rotationAngle` is the angle (in degrees) that the ellipse described by `radiusX` and `radiusY` is rotated clockwise about its center.

The following simple code snippet, registers a single handler for the `touchstart`, `touchmove` and `touchend` events. When the `src` element is touched, the element's width and height will be calculate based on the touch point's `radiusX` and `radiusY` values and the element will then be rotated using the touch point's `rotationAngle`.

```
<div id="src"> ... </div>
var src = document.getElementById("src");
```

```
src.addEventListener('touchstart', rotate);
src.addEventListener('touchmove', rotate);
src.addEventListener('touchend', rotate);
```

```
function rotate (e) {
  var touch = e.changedTouches.item(0);
```

```
  // Turn off default
  event handling
  e.preventDefault();
```

```
  // Rotate element
  'src'.
  src.style.width =
touch.radiusX * 2 + 'px';
  src.style.height =
touch.radiusY * 2 + 'px';
  src.style.transform =
"rotate(" + touch.rotationAngle + "deg)";
};
```

## Specifications

Specification

Browser compatibility  
Touch Events - Level 2

Editor's Draft

Non-stable version.

Desktop  
Mobile





Status

Comment

# Touch.radiusY

-----

[Touch.radiusY](#) Read only

Returns the Y radius of the ellipse that most closely circumscribes the area of contact with the screen. The value is in pixels of the same scale as screenY.

## Summary

Returns the Y radius of the ellipse that most closely circumscribes the area of contact with the touch surface. The value is in CSS pixels of the same scale as [Touch.screenX](#).

This value, in combination with [Touch.radiusX](#) and `Touch.rotationAngle` constructs an ellipse that approximates the size and shape of the area of contact between the user and the screen. This may be a large ellipse representing the contact between a fingertip and the screen or a small one representing the tip of a stylus, for example.

**Note:** This attribute has *not* been formally standardized. It is specified in the [Touch Events – Level 2](#) Editor's Draft specification and not in Touch Events Recommendation. This attribute is not widely implemented.

## Syntax

```
var yRadius = touchItem.radiusY;
```

## Return value

yRadius

The Y radius of the ellipse that most closely circumscribes the area of contact with the screen.

## Example

The [Touch.radiusX example](#) includes an example of this property's usage.

## Specifications

Specification

[Touch Events – Level 2](#)

Editor's Draft

Non-stable version.

Browser compatibility

--

Status

Comment

## Touch.rotationAngle

-----  
-

[Touch.rotationAngle](#) Read only  
Returns the angle (in degrees) that the ellipse described by `radiusX` and `radiusY` must be rotated, clockwise, to most accurately cover the area of contact between the user and the surface.

### Summary

Returns the rotation angle, in degrees, of the contact area ellipse defined by `Touch.radiusX` and `Touch.radiusY`. The value may be between 0 and 90. Together, these three values describe an ellipse that approximates the size and shape of the area of contact between the user and the screen. This may be a relatively large ellipse representing the contact between a fingertip and the screen or a small area representing the tip of a stylus, for example.

**Note:** This attribute has *not* been formally standardized. It is specified in the [Touch Events – Level 2](#) Editor's Draft specification and not in Touch Events Recommendation. This attribute is not widely implemented.

### Syntax

```
var angle = touchItem.rotationAngle;
```

### Return value

angle  
The number of degrees of rotation to apply to the described ellipse to align with the contact area between the user and the touch surface.

### Example

The [Touch.radiusX](#) example includes an example of this property's usage.

# Specifications

Specification

Browser Compatibility  
~~Touch Events - Level 2~~

Editor's Draft

Non-stable version.

Desktop  
Mobile

--

Status

Comment

## Touch.force

-----

[Touch.force](#) Read only

Returns the amount of pressure being applied to the surface by the user, as a float between 0.0 (no pressure) and 1.0 (maximum pressure).

The **Touch.force** read-only property returns the amount of pressure the user is applying to the touch surface for a [Touch](#) point.

## Syntax

```
touchItem.force;
```

## Return value

A float that represents the amount of pressure the user is applying to the touch surface. This is a value between 0.0 (no pressure) and 1.0 (the maximum amount of pressure the hardware can recognize). A value of 0.0 is returned if no value is known (for example the touch device does not support this property). In environments where force is known, the absolute pressure represented by the force attribute, and the sensitivity in levels of pressure, may vary.

## Example

This example illustrates using the [Touch](#) interface's [Touch.force](#) property. This property is a relative value of pressure applied, in the range 0.0 to 1.0, where 0.0 is no pressure, and 1.0 is the highest level of pressure the touch device is capable of sensing. In following code snippet, the [touchstart](#) event handler iterates through the targetTouches list and logs the

force value of each touch point but the code could invoke different processing depending on the value.

```
someElement.addEventListener('touchstart', function(e) {  
    // Iterate through the list of touch points and log each touch  
    // point's force.  
    for (var i=0; i < e.targetTouches.length; i++) {  
        // Add code to "switch" based on the force value. For example  
        // minimum pressure versus maximum pressure could result in  
        // different handling of the user's input.  
        console.log("targetTouches[" + i + "].force = " + e.targetTouches[i].force);  
    }  
}, false);
```

## Specifications

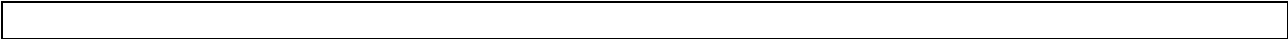
Specification

Browser compatibility  
Touch Events - Level 2

Editor's Draft

Initial definition.

Desktop  
Mobile



Chrome

Firefox (Gecko)

Internet Explorer

Opera

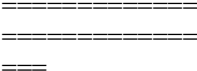
Safari (WebKit)

## Methods



*This interface has no method and no parent, and doesn't inherit or implements any method.*

## Browser compatibility



- Desktop
- Mobile

Feature

Basic support	22.0	<a href="#">18.0</a> (18.0)	Not supported	Not supported	Not supported
radiusX, radiusY, rotationAngle, force	<span style="color: orange;">?</span>	(Yes)	<span style="color: orange;">?</span>	<span style="color: orange;">?</span>	<span style="color: orange;">?</span>
Touch() constructor	<span style="color: orange;">?</span>	Not supported	<span style="color: orange;">?</span>		