ABBYY®

ABBYY Mobile Capture SDK

Cordova Plugin

Table of Contents

Introduction

ABBYY Mobile Capture SDK Cordova Plugin allows to use ABBYY Mobile Capture SDK features in apps based on the Apache Cordova framework. Image Capture, Text Capture and Data Capture features are available as complete scenarios, including user interface implementation. Core API methods for low-level single image processing, such as data extraction, text recognition, images export, etc., can be simply integrated into your projects.

This plugin requires the ABBYY Mobile Capture assets, which differ for Android and iOS, and a license file. You can request ABBYY Mobile Capture trial version on the ABBYY website.

This manual describes the **AbbyyRtrSdk** JavaScript module. More information, including the library usage details, is available in the *ABBYY Mobile Capture SDK Developer's Guide* found in the library packages.

Getting Started

To start developing with the ABBYY Mobile Capture SDK Cordova Plugin, you need to add target platforms and the plugin to your project, and copy the ABBYY Mobile Capture SDK assets and native libraries for Android and iOS, as described below.

1. Create a project:

```
cordova create path/to/ProjectDir com.example.abbyyrtrsdk MyRTRProject
```

2. Change to the project directory to add platforms and the plugin:

```
cd path/to/ProjectDir
cordova platform add ios
cordova platform add android
cordova plugin add cordova-plugin-abbyy-rtr-sdk
```

3. Add ABBYY Mobile Capture SDK files to the following project subdirectories:
   a. Copy Mobile Capture SDK assets and your license file to **www/rtr_assets**.
   b. Copy Android libraries (**abbyy-rtr-sdk-1.0.aar** and **abbyy-ui-1.0.aar**) to **libs/android.**
   c. Copy all the iOS frameworks to **libs/ios**.
4.

The **libs/android** and **libs/ios** paths should be added to the linker search paths. This step is performed automatically during the plugin installation.

5. Use **AbbyyRtrSdk** module methods to add text and data capture functionality to your app.

- Android:

```
cordova build android
cordova run android
```

- iOS (don't forget to specify your Development Team):

```
cordova build ios --buildFlag="-UseModernBuildSystem=0" --
buildFlag="DEVELOPMENT_TEAM=<YOUR_TEAM>"
cordova run ios --buildFlag="-UseModernBuildSystem=0" --
buildFlag="DEVELOPMENT_TEAM=<YOUR_TEAM>"
```

AbbyyRtrSdk module

ABBYY Mobile Capture SDK plugin module.

Methods

Name	Description
User interface	
startTextCapture	Opens a modal dialog for the Text Capture scenario.
startDataCapture	Opens a modal dialog for the Data Capture scenario.
startImageCapture	Opens a modal dialog for the Image Capture scenario.
Core API	
recognizeText	Starts a text capture scenario for a single image.
extractData	Starts a data capture scenario for a single image.
detectDocumentBoundary	Detects a quadrangle representing document boundary on an image.
cropImage	Crops image according to the document boundary and size.
rotateImage	Rotates image by specified angle.
assessQualityForOcr	Estimates if an image quality is suitable for OCR.
exportImage	Exports an image to JPG or PNG format.
exportImagesToPdf	Exports images to PDF format.

User interface methods

This section contains description of methods for simple starting capture scenario with ready-to-use user interface.

startTextCapture method

Opens a modal dialog with controls for the Text Capture scenario.

```
AbbyyRtrSdk.startTextCapture(callback, options)
```

Parameters

callback

The callback function which receives the text capture operation result. Your callback should expect a single JSON object as an argument (see Result).

options

JSON object specifying text capture parameters (see Options).

Options

The table below describes the JSON object that you can pass as the *options* argument to change text capture settings. All keys are optional. Omitting a key means that a default setting will be used.

Key	Value type	Description
licenseFileName	string	The name of the license file. This file must be located in the **www/rtr_assets/** directory in your project. Default: "AbbyyRtrSdk.license".
selectableRecognitionLanguages	string array	Recognition languages which can be selected by the user, for example: ["English", "French", "German"]. Empty array disables language selection. ***Note:*** *For the list of supported languages and their identifiers, see Specifications — Available Languages in the ABBYY Mobile Capture SDK Developer's Guide.* Default: [] (empty array, language selection disabled).
recognitionLanguages	string array	Recognition language selected by default. Default: ["English"].

Key	Value type	Description
areaOfInterest	string	Width and height of the recognition area, separated by a whitespace — for example: "0.8 0.3". The area of interest is centered in the preview frame, its width and height are relative to the preview frame size and should be in the [0.0, 1.0] range. Default: "0.8 0.3" (intended to capture a few lines of text in portrait mode).
orientation	string	The type of image orientation. This key can be set to the following values: • "portrait" • "landscape" • "default" (the orientation will be detected automatically) Default: "default".
stopWhenStable	boolean	Whether to stop the plugin as soon as the result status is "Stable" (see Result status). When enabled (true), the recognition process can be stopped automatically. When disabled (false), recognition can be stopped only manually by user. Default: true (automatic stop enabled).
isFlashlightVisible	boolean	Show (true) or hide (false) the flashlight button in the text capture dialog. Default: true (flashlight available).
isStopButtonVisible	boolean	Show (true) or hide (false) the stop button in the text capture dialog. When the user taps stop, Mobile Capture SDK returns the latest recognition result. Default: true (manual stop available).

Result

This section describes the JSON object that represents text recognition results. The callback you implement should parse this object and show results to user.

Key	Value type	Description

textLines	object array	An array of objects representing recognized lines of text. These objects have the following keys:
		• **text** (string): the recognized text.
		• **quadrangle** (object): quadrangle containing text, represented by an array of its four vertices' coordinates: { **x**: int, **y**: int }. The vertices are indexed clockwise starting from the bottom left.
		• **rect** (object): bounding rectangle for quadrangle. The rectangle is set by the following parameters:
		  • **top** (int): upper left point ordinate
		  • **bottom** (int): bottom left point ordinate
		  • **left** (int): upper left point abscissa
		  • **right** (int): upper right point abscissa
		Note: *The **rect** key exists in the iOS version only.*
		If an error occurs during processing, the **textLines** key is not present in the result.

resultInfo	object	Additional information. This object has the following keys: • **stabilityStatus** (string): result stability status. See Result status for details. • **userAction** (string): the user's action which stopped the plugin, if any. Can be "Manually Stopped" if the stop button has been used, and "Canceled" if the user canceled processing. If the plugin has stopped automatically, the **userAction** key is not present in **resultInfo**. • **frameSize** (string): full size of the preview frame, a string with 2 integers separated with a whitespace ("720 1280"). • **recognitionLanguages** (string array): languages used for recognition, the array contains language identifiers (["English", "French", "German"]).
error	object	Error details. This key is present only if an error occurs. The value is an object which has a single key: • **description** (string): human-readable error description.

Below is an example of a result JSON when text capture succeeds.

```
{
  textLines : [
    {
      text : "Welcome to ABBYY Mobile Capture SDK",
      "quadrangle": [
        {
         "x": 82,
         "y": 567
```

```
        },
         {
          "x": 82,
          "y": 567
        },
         {
          "x": 82,
          "y": 567
        },
         {
          "x": 82,
          "y": 567
        }
       ],
     "rect": {
      "top": 515,
      "bottom": 568,
      "left": 82,
      "right": 528
     },
     }  ],
   resultInfo : {
     stabilityStatus : "Available",
     userAction : "Manually Stopped",
     frameSize : "720 1280",
     recognitionLanguages : ["English", "French"]
   }
}
```

On error, you will receive a JSON which does not contain recognized text but provides an error message.

```
{
  error : {
    description : "Something has gone wrong."
  },
  resultInfo : {
    stabilityStatus : "Tentative",
    userAction : "Canceled",
    frameSize : "720 1280",
    recognitionLanguages : ["English", "French"]
  }
}
```

startDataCapture method

Opens a modal dialog with controls for the Data Capture scenario.

```
AbbyyRtrSdk.startDataCapture(callback, options)
```

Parameters

callback

The callback function which receives the data capture operation result. Your callback should expect a single JSON object as an argument (see Result).

options

JSON object specifying data capture parameters (see Options).

Options

The table below describes the JSON object that you can pass as the *options* argument to change data capture settings. All keys are optional. Omitting a key means that a default setting will be used, except the **profile** and **customDataCaptureScenario** keys: you must specify either one of them, but not both at the same time.

Key	Value type	Description
profile	string	The predefined data capture profile to use, for example: "MRZ". ***Note:*** *For the list of supported documents, see Specifications — Data Capture Profiles in the ABBYY Mobile Capture SDK Developer's Guide.*
customDataCaptureScenario	object	Custom data capture settings. This object has the following keys: • **name** (string): the name of your custom data capture scenario, required. • **description** (string): a more detailed description. This key is optional; if not given, it will be assigned the same value as **name**. • **recognitionLanguages** (string array): recognition languages to use. Default is ["English"]. • **fields** (object array): describes data fields to capture. Each object in this array has a single **regEx** key; its value is a string containing the regular expression that should be matched when capturing a field. ***Note:*** *For details, see the How to Capture a Custom Data Field guide in the ABBYY Mobile Capture SDK Developer's Guide.* ***Note:*** *Currently a custom data capture profile supports one recognized field only (**fields** must contain only 1 element).* See below for an example of a custom data capture scenario definition.
licenseFileName	string	The name of the license file. This file must be located in the **www/rtr_assets/** directory in your project.

Key	Value type	Description
		Default: "AbbyyRtrSdk.license".
areaOfInterest	string	Width and height of the recognition area, separated by a whitespace — for example: "0.8 0.3". The area of interest is centered in the preview frame, its width and height are relative to the preview frame size and should be in the [0.0, 1.0] range. Default: "0.8 0.3" (intended to capture a few lines of text in portrait mode).
orientation	string	The type of image orientation. This key can be set to the following values: • "portrait" • "landscape" • "default" (the orientation will be detected automatically) Default: "default".
stopWhenStable	boolean	Whether to stop the plugin as soon as the result status is "Stable" (see Result status). When enabled (true), the recognition process can be stopped automatically. When disabled (false), recognition can be stopped only manually by user. Default: true (automatic stop enabled).
isFlashlightVisible	boolean	Show (true) or hide (false) the flashlight button in the text capture dialog. Default: true (flashlight available).
isStopButtonVisible	boolean	Show (true) or hide (false) the stop button in the text capture dialog. When the user taps stop, Mobile Capture SDK returns the latest recognition result. Default: true (manual stop available).
recognitionLanguages	string	Recognition languages to use. If any of defined languages is not supported for current profile, an error will occur. ***Note:*** *For the list of supported languages for documents, see Specifications — Supported ID Documents in the ABBYY Mobile Capture SDK Developer's Guide.*

Key	Value type	Description
		Default: ["English"].

An example of a custom data capture scenario definition in the options JSON.

```
{
  ...

  customDataCaptureScenario : {
    name : "Code",
    description : "Alphanumeric code, for example: X6YZ64 32VPA zyy777.",
    recognitionLanguages : ["English"],
    fields : [ {
      regEx : "([a-zA-Z]+[0-9]+|[0-9]+[a-zA-Z]+)[0-9a-zA-Z]*"
    } ]
  },

  ...

}
```

Result

This section describes the JSON object that represents data recognition results. The callback you implement should parse this object and show results to user.

Key	Value type	Description
dataScheme	object	The data scheme which was applied during data capture. The value is an object which has two keys: • **id** (string): the internal scheme identifier. • **name** (string): the scheme name. If you had defined a custom data capture scenario in options, both the **id** and **name** will be the same as the scenario name you specified. If you selected a predefined profile, the **id** and **name** are specified by the profile. If an error occurs during processing, the **dataScheme** key is not present in the result.
dataFields	object array	Recognized data fields. Each object in the array represents a separate data field. The data field objects have the following keys: • **id** (string): the internal identifier of the field. • **name** (string): the field name. Similarly to **dataScheme**, in custom scenarios both **id**

Key	Value type	Description

		and **name** are the same as the scenario name you specified (currently custom scenarios allow only 1 recognized field).
		• **text** (string): full text of the field.
		• **quadrangle** (object): quadrangle containing text, represented by an array of its four vertices' coordinates: { **x**: int, **y**: int }. The vertices are indexed clockwise starting from the bottom left.
		• **components** (object array): an array of objects representing field components, that is, the text fragments found on the image, which constitute the field.
		
		In the **components** array each element is an object with the following keys:
		
		• **text** (string): text of this fragment.
		• **quadrangle** (object): quadrangle containing text, represented by an array of its four vertices' coordinates: { **x**: int, **y**: int }. The vertices are indexed clockwise starting from the bottom left.
		• **rect** (object): bounding rectangle for quadrangle. The rectangle is set by the following parameters:
		  • **top** (int): upper left point ordinate
		  • **bottom** (int): bottom left point ordinate
		  • **left** (int): upper left point abscissa
		  • **right** (int): upper right point abscissa
		
		Note: *The **rect** key exists in the iOS version only.*
		
		If an error occurs during processing, the **dataFields** key is not present in the result.

resultInfo	object	Additional information. This object has the following keys: • **stabilityStatus** (string): result stability status. See Result status for details. • **userAction** (string): the user's action which stopped the plugin, if any. Can be "Manually Stopped" if the stop button has been used, and "Canceled" if the user canceled processing. If the plugin has stopped automatically, the **userAction** key is not present in **resultInfo**. • **frameSize** (string): full size of the preview frame, a string with 2 integers separated with a whitespace ("720 1280").
error	object	Error details. This key is present only if an error occurs. The value is an object which has a single key: • **description** (string): human-readable error description.

Below is an example of a result JSON when data capture succeeds.

```
{
  dataScheme : {
    id : "Hello",
    name : "Hello"
  },
  dataFields : [
    {
      id : "Hello",
      name : "Hello",
      text : "Hello world!",
      "quadrangle": [
        {
         "x": 82,
         "y": 567
       },
        {
         "x": 82,
         "y": 567
       },
        {
         "x": 82,
         "y": 567
       },
        {
         "x": 82,
         "y": 567
       }
      ],
      components : [
        {
          text : "Hello",
         "quadrangle": [
          {
           "x": 100,
           "y": 780
         },
          {
           "x": 100,
           "y": 600
         },
          {
           "x": 350,
          "y": 600
        },
         {
          "x": 350,
          "y": 780
        }
       ],
          "rect": {
           "top": 515,
           "bottom": 568,
           "left": 82,
           "right": 528
         },
       },
```

```
...


        }
      ]
    }
  ],
  resultInfo : {
    stabilityStatus : "Available",
    userAction : "Manually Stopped",
    frameSize : "720 1280"
  }
}
```

On error, you will receive a JSON which does not contain recognized data but provides an error message.

```
{
  error : {
    description : "Something unexpected have happened."
  }
  resultInfo : {
    stabilityStatus : "Tentative",
    userAction : "Canceled",
    frameSize : "720 1280"
  }
}
```

startImageCapture method of AbbyyRtrSdk module

Opens a modal dialog with controls for the Image Capture scenario.

```
AbbyyRtrSdk.startImageCapture(callback, options)
```

Parameters

callback

The callback function which receives the image capture operation result. Your callback should expect a single JSON object as an argument (see Result).

options

JSON object specifying image capture parameters (see Options).

Options

The table below describes the JSON object that you can pass as the *options* argument to change image capture settings. All keys are optional. Omitting a key means that a default setting will be used.

Key	Value type	Description
licenseFileName	string	The name of the license file. This file must be located in the **www/rtr_assets/** directory in your project. Default: "AbbyyRtrSdk.license".
cameraResolution	string	The resolution of the images captured from the camera preview. This key can be set to the following values: • "HD" • "FullHD" • "4K" (for iOS only) Default: "FullHD".
isFlashlightButtonVisible	boolean	Show (true) or hide (false) the flashlight button in the image capture dialog. Default: true (flashlight visible).
isCaptureButtonVisible	boolean	Show (true) or hide (false) the button for taking photo manually. Default: false (button is hidden).
isGalleryButtonVisible	boolean	Show (true) or hide (false) the button for choosing an image from photo gallery. Default: true (flashlight visible).
orientation	string	The type of image orientation. This key can be set to the following values: • "portrait" • "landscape" • "default" (the orientation will be detected automatically) Default: "default".
destination	string	Captured image will be saved to corresponding file ("file") or returned as encode base64 image string ("base64"). ***Note:*** *If more than 1 image is captured, this option value will be ignored and the images will be saved to files anyway.* Default: "file".

Key	Value type	Description
exportType	string	Captured image will be saved to this format. This key can be set to the following values: • "jpg" • "png" • "pdf" (all images will be saved to one pdf file) Default: "jpg".
showPreview	boolean	Show image preview after capture (true) or wait until the number of captured images is equal to the **requiredPageCount** (false). Default: false.
requiredPageCount	int	Total number of pages to be captured. Set the page-limitation mode of the image capture as following: • 0 to allow unlimited image capture. The set of the result images can be saved or edited at any time; • a positive value to set the exact number of images that should be captured. Images saving is enabled only when this number of images are been captured. Default: 0.
compressionLevel	string	The uniform image compression scale for lossy formats. This key can be set to the following values: • "Low" • "Normal" • "High" • "ExtraHigh" Default: "Low".
defaultImageSettings	object	Custom image capture settings. This object has the following keys: • **aspectRatioMin** (float): Lower limit of the document's aspect ratio. This property is used in pair with the **aspectRatioMax**, defining an interval of acceptable aspect ratio values of the document to be captured. Setting aspect ratio will help to improve boundary detection accuracy. If only **aspectRatioMax** is set, **aspectRatioMin** will be set to 1. Default: 0 (aspect ratio is not set). • **aspectRatioMax** (float): Upper limit of document's aspect ratio.

Key	Value type	Description
		This property is used in pair with the **aspectRatioMin**, defining an interval of acceptable aspect ratio values of the document to be captured. Setting aspect ratio will help to improve boundary detection accuracy. If only **aspectRatioMin** is set, **aspectRatioMax** will be set to infinity. Default: 0 (aspect ratio is not set). • **imageFromGalleryMaxSize** (int): Maximum available size of an image, loaded from the gallery. The size is defined as the length of the largest side of an image (in pixels). Default: 4096.**minimumDocumentToViewRatio** (float): the minimum document area relative to the whole frame area, required for capture. If the document area is less than this ratio, the image will not be captured. The value can be from 0 to 1. Default: 0.15. • **documentSize** (string): expected size of the original document in millimeters. This key can be set to the following constant values: ○ "A4" ○ "BusinessCard" ○ "Letter" ○ "Any" or defined as a string with 2 integers separated with a whitespace: "54 86". All keys are optional.

Result

This section describes the JSON object that represents image capture results. The callback you implement should parse this object and show results to user.

Key	Value type	Description
images	object array	Captured images. This parameter is returned only if the **exportType** is set to "jpg" or "png". Each object in the array represents a single image with corresponding information. The image objects have the following parameters: • **resultInfo** (object): this object has the following parameters: ○ **exportType** (string): the format of the captured image. ○ **imageSize** (object): full size of the preview frame, an object with int parameters **width** and **height**. • **filePath** (string): full path to the exported file. This parameter is present if the destination is 'file'.

Key	Value type	Description
		• **base64** (string): the original image returned as base64 image string. This parameter is present if the **destination** parameter was set to "base64".
pdfInfo	object	This parameter is returned if the **exportType** is set to "pdf". The value is an object which has the following parameters: • **filePath** (string): full path to the exported PDF file. • **pagesCount** (integer): number of pages in the exported PDF file. Normally fits the number of captured images.
resultInfo	object	Additional information. This object has the following parameters: • **userAction** (string): the user's action which stopped the module, if any. Can be "Canceled" if the user canceled processing. If the module has stopped automatically, the **userAction** parameter is not present in **resultInfo**. • **uriPrefix** (string): add this string to the beginning of the **filePath** or **base64** parameter to get a URI. This parameter can be 'file://' if the destination is 'File' and '' or '' if the destination is 'Base64'.
error	object	Error details. This key is present only if an error occurs. The value is an object which has a single key: • **description** (string): human-readable error description.

Below is an example of a result JSON when image capture succeeds.

```
{
   "images":[
      {
         "resultInfo":{
            "exportType":"jpg"
         },
         "filePath":"/private/var/mobile/Containers/Data/Application/XXXXXX
XX-XXXX-XXXX-XXXX-XXXXXXXXXXXX/tmp/1562917370501.jpg"
      }
   ],
   "resultInfo":{
      "userAction":"Manually Stopped"
   },
   "pdfInfo":{
      "compressionLevel":"Normal",
      "pdfCompressionType":"jpg",
      "pagesCount":1,
      "filePath":"/private/var/mobile/Containers/Data/Application/XXXXXXXX-
XXXX-XXXX-XXXX-XXXXXXXXXXXX/tmp/ImageCapture - 2019-07-12 10_42_57.pdf"
```

```
    }
}
```

On error, you will receive a JSON which does not contain image capture result but provides an error message.

```
{
   "error":
   {
      "description":"Unacceptable license information is used or the
functionality is not available under the license."
   }
}
```

Core API methods

This section contains description of the low-level methods for single image processing.

recognizeText method

Starts text capture and recognition for a single image. Captured image will be processed according to the passed settings and recognized text will be returned with some technical information such as warnings, if any, and text orientation on the image.

```
AbbyyRtrSdk.recognizeText(callback, options)
```

Parameters

options

Object specifying parameters of the text capture scenario for a single image (see Options).

Return values

The method returns result depending on how the scenario finished (see Result).

Options

The table below describes parameters that you can pass as the *options* argument to change text capture settings. Omitting a parameter means that a default setting will be used.

Parameter	Value type	Description
licenseFileName	string	The name of the license file. This file must be located in the **/assets/** directory in your project.

Parameter	Value type	Description
		Default: "MobileCapture.License".
imageUri	string	**imageUri** (string): Image source for the operation passed as URI. This parameter can be set to the following values: • "Base64": you can pass an encode base64 image string, i.e. "*<base64 encoded image data>*" • "File": image file address, i.e. "*file:///data/user/0/com.abbyy.rtr.reactnative.sample.coreapi/files/pages/page_848b121d-5a7a-4ead-94e6-dd91bc6bfead.jpg*". '*content://*' schema is also supported for Android. Required parameter.
areaOfInterest	object	A rectangle specifying the area of interest in the coordinates, set by the following parameters: • **top** (int): upper left point ordinate • **bottom** (int): bottom left point ordinate • **left** (int): upper left point abscissa • **right** (int): upper right point abscissa I.e., {top: 100, bottom: 1000, left: 100, right: 1000} ***Note:*** *All parameters should be defined to set the* ***areaOfInterest*** *parameter. If some of them are defined and some are not, an error occurs.* Default: a rectangle, containing the whole image.
isTextOrientationDetectionEnabled	boolean	Enables or disables detection of the image orientation while preprocessing. If the property is set to **true**, the image top is detected and correct orientation can be used for image rotation. You can set this property to **false** for speeding the process up. ***Note:*** *Disable the image detection only if you can be sure that the captured image has correct orientation. Otherwise the text on image will not be detected and recognized.* The default value of this property is **true** (enabled).

Parameter	Value type	Description
recognitionLanguages	string[]	List of languages, supported for the text recognition, i.e. ['English', 'Russian']. See the full list of supported languages here. Default: ['English'].

Result

This section describes the object that represents text capture results. Returned parameters depend on the captured and passed image.

Parameter	Value type	Description
orientation	int	An angle on which the image was rotated to get normal orientation. Possible values are: 0, 90, 180, 270.
warnings	string[]	Warnings that occurred during processing, if any.
text	string	The whole recognized text as a single string. Text lines are divided with "\n". Text blocks are divided with "\n\n".
textBlocks	object[]	An array of text blocks. Each text block is represented as an array of **textLines** parameter.

textBlocks.textLines	object[]	An array of text lines. Each text line has the following parameters:
		• **text** (string): recognized text.
		• **quadrangle** (object): quadrangle containing text, represented by an array of its four vertices' coordinates: { **x**: int, **y**: int }. The vertices are indexed clockwise starting from the bottom left.
		• **rect** (object): bounding rectangle for quadrangle. The rectangle is set by the following parameters:
		  • **top** (int): upper left point ordinate
		  • **bottom** (int): bottom left point ordinate
		  • **left** (int): upper left point abscissa
		  • **right** (int): upper right point abscissa
		• **charInfo** (object[]): an array, which elements contain information concerning concrete recognized symbol.

Parameter	Value type	Description

textBlocks.textLines.charInfo	object[]	An array, which elements contain information concerning concrete recognized symbol. This object contains parameters, describing the symbol's place at an image, and its recognition confidence.
		• **quadrangle** (object): quadrangle containing text, represented by an array of its four vertices' coordinates: { **x**: int, **y**: int }
		• **rect** (object): bounding rectangle for quadrangle. The rectangle is set by the following parameters:
		  • **top** (int): upper left point ordinate
		  • **bottom** (int): bottom left point ordinate
		  • **left** (int): upper left point abscissa
		  • **right** (int): upper right point abscissa
		• **isUncertain** (boolean): returned only in case the symbol is uncertain.

Example of a result JSON.

```
{"textBlocks": [{
  "textLines": [
   {
     "text": "",
     "quadrangle": [
       {
        "x": 82,
        "y": 567
      },
       {
        "x": 82,
        "y": 567
      },
       {
        "x": 82,
        "y": 567
      },
       {
        "x": 82,
        "y": 567
      }
     ],
     "rect": {
       "top": 515,
       "bottom": 568,
       "left": 82,
       "right": 528
     },
     "charInfo": [
       {
        "quadrangle": [
         {
```

```
            "x": 82,
            "y": 567
          },
          {
            "x": 82,
            "y": 567
          },
          {
            "x": 82,
            "y": 567
          },
          {
            "x": 82,
            "y": 567
          }
        ],
        "rect": {
         "top": 515,
         "bottom": 568,
         "left": 82,
         "right": 528
        },

        "isUncertain": true,
        "isItalic": true
       }
      ],
      "text": "",
      "orientation": 0
     }]
   }]
}
```


extractData method

Starts data capture for a single image. Data fields will be detected and recognized on the captured image according to the passed settings. One of the settings is a date capture profile, representing a type of a document to be recognized, i.e. a business card. The profile defines a data schema, describing document fields. The profile schema is applied during the capture process.

Result will be returned with some technical information such as warnings, if any, and text orientation on the image.

Note: *The functionality is currently supported for BusinessCards profile only.*

```
AbbyyRtrSdk.extractData(callback, options)
```

Parameters

options

Object specifying parameters of the data capture scenario for a single image (see Options).

Return values

The method returns result depending on how the scenario finished (see Result).

Options

The table below describes parameters that you can pass as the *options* argument to change data capture settings. Omitting a parameter means that a default setting will be used.

Parameter	Value type	Description
licenseFileName	string	The name of the license file. This file must be located in the **/assets/** directory in your project. Default: "MobileCapture.License".
imageUri	string	**imageUri** (string): Image source for the operation passed as URI. This parameter can be set to the following values: • "Base64": you can pass an encode base64 image string, i.e. "*<base64 encoded image data>*" • "File": image file address, i.e. "*file:///data/user/0/com.abbyy.rtr.reactnative.sample.coreapi/files/pages/page_848b121d-5a7a-4ead-94e6-dd91bc6bfead.jpg*". '*content://*' schema is also supported for Android. Required parameter.
profile	string	The profile name. I.e. 'BusinessCards'. ***Note:*** *The functionality is currently supported for BusinessCards profile only.*
isTextOrientationDetectionEnabled	boolean	Enables or disables detection of the image orientation while preprocessing. If the property is set to **true**, the image top is detected and correct orientation can be used for image rotation. You can set this property to **false** for speeding the process up.

Parameter	Value type	Description
		Note: *Disable the image detection only if you can be sure that the captured image has correct orientation. Otherwise the text on image will not be detected and recognized.* The default value of this property is **true** (enabled).
recognitionLanguages	string[]	List of languages, supported for the text recognition, i.e. ['English', 'Russian']. See the full list of supported languages here. Default: ['English'].

Result

This section describes the object that represents data capture results. Returned parameters depend on the captured and passed image.

Parameter	Value type	Description
orientation	int	An angle on which the image was rotated to get normal orientation. Possible values are: 0, 90, 180, 270.
warnings	string[]	Warnings that occurred during processing, if any.

dataFields	object[]	List of objects representing captured and recognized data fields. Each data field has the following parameters:
		• **id** (int): internal identifier of the data field.
		• **name** (string): human readable name of the data field.
		• **text** (string): recognized text of the data field.
		• **quadrangle** (object): quadrangle containing text, represented by an array of its four vertices' coordinates: { **x**: int, **y**: int }. The vertices are indexed clockwise starting from the bottom left.
		• **rect** (object): bounding rectangle for quadrangle. The rectangle is set by the following parameters:
		  • **top** (int): upper left point ordinate
		  • **bottom** (int): bottom left point ordinate
		  • **left** (int): upper left point abscissa
		  • **right** (int): upper right point abscissa
		• **components** (object[]): field components. If the field has only one component, this array contains one element.

Parameter	Value type	Description
		This property may be 0, if the field is not compound. In this case result does not contain **components**. Each component can have the same parameters as **dataField** including **components** array. • **charInfo** (object[]): an array, which elements contain information concerning concrete recognized symbol.

dataFields.charInfo	object[]	An array, which elements contain information concerning concrete recognized symbol. This object contains parameters, describing the symbol's place at an image, and its recognition confidence.
		• **quadrangle** (object): quadrangle containing text, represented by an array of its four vertices' coordinates: { **x**: int, **y**: int }
		• **rect** (object): bounding rectangle for quadrangle. The rectangle is set by the following parameters:
		  • **top** (int): upper left point ordinate
		  • **bottom** (int): bottom left point ordinate
		  • **left** (int): upper left point abscissa
		  • **right** (int): upper right point abscissa
		• **isUncertain** (boolean): returned only in case the symbol is uncertain.

Example of a result JSON.

```
{"dataFields": [
  {
    "id": "Mobile",
   "name": "Mobile",
   "text": "",
    "quadrangle": [
     {
         "x": 82,
       "y": 567
     },
     {
       "x": 82,
      "y": 567
     },
     {
       "x": 82,
       "y": 567
     },
     {
       "x": 82,
       "y": 567
     }
    ],
    "rect": {
```

```
      "top": 515,
      "bottom": 568,
      "left": 82,
      "right": 528
     },
     "charInfo": [
      {
        "quadrangle": [
         {
           "x": 82,
           "y": 567
         },
         {
           "x": 82,
           "y": 567
         },
         {
           "x": 82,
           "y": 567
         },
         {
           "x": 82,
           "y": 567
         }
        ],
        "rect": {
         "top": 515,
         "bottom": 568,
         "left": 82,
         "right": 528
        },
        "isUncertain": true
      }
     ],
    "components": [
      {
       "id": "PhoneCode",
       "name": "PhoneCode",
       "text": "",
       "quadrangle": [
         {
          "x": 82,
          "y": 567
         },
         {
          "x": 82,
          "y": 567
         },
         {
          "x": 82,
          "y": 567
         },
         {
          "x": 82,
          "y": 567
         }
```

```
        ],
        "rect": {
          "top": 515,
          "bottom": 568,
          "left": 82,
          "right": 528
        },
        "charInfo": [
          {
           "quadrangle": [
             {
              "x": 82,
              "y": 567
             },
             {
              "x": 82,
              "y": 567
             },
             {
              "x": 82,
              "y": 567
             },
             {
              "x": 82,
              "y": 567
             }
           ],
           "rect": {
             "top": 515,
             "bottom": 568,
             "left": 82,
             "right": 528
           },
           "isUncertain": true
          }
        ]
       }
    ]
   }]
}
```


detectDocumentBoundary method

Detects a quadrangle representing document boundary on an image.

```
AbbyyRtrSdk.detectDocumentBoundary(callback, options)
```

Parameters

options

Object specifying parameters of the document boundary detection (see Options).

Return values

The method returns result depending on how the document boundary detection finished (see Result).

Options

The table below describes parameters that you can pass as the *options* argument to change document boundary detection settings. Omitting a parameter means that a default setting will be used.

Parameter	Value type	Description
licenseFileName	string	The name of the license file. This file must be located in the **/assets/** directory in your project. Default: "MobileCapture.License".
imageUri	string	**imageUri** (string): Image source for the operation passed as URI. This parameter can be set to the following values: • "Base64": you can pass an encode base64 image string, i.e. "*<base64 encoded image data>*" • "File": image file address, i.e. "*file:///data/user/0/com.abbyy.rtr.reactnative.sample.coreapi/files/pages/page_848b121d-5a7a-4ead-94e6-dd91bc6bfead.jpg*". '*content://*' schema is also supported for Android. Required parameter.
areaOfInterest	object	A rectangle specifying the area of interest in the coordinates, set by the following parameters: • **top** (int): upper left point ordinate • **bottom** (int): bottom left point ordinate • **left** (int): upper left point abscissa • **right** (int): upper right point abscissa I.e., {top: 100, bottom: 1000, left: 100, right: 1000} ***Note:*** *All parameters should be defined to set the* ***areaOfInterest*** *parameter. If some of them are defined and some are not, an error occurs.* Default: a rectangle, containing the whole image.
detectionMode	string	Document boundary detection mode. The mode influences the crop speed and accuracy. This parameter can be set to the following values:

Parameter	Value type	Description
		○ 'Fast' - this mode signifies processing speed ○ 'Default' - balanced mode, that combines optimal processing speed and high quality. Default: 'Default'.
documentSize	object	Document size represented by its width and height in millimeters. The values of the width and height are set to the **documentSize** parameters: • **width** (float) • **height** (float) I.e., {x: 210; y: 297} for A4 document size. To leave the document size undefined, set both width and height to 0. In this case document of any size will be detected.

Result

This section describes the object that represents image capture results. Returned parameters depend on the scenario.

Parameter	Value type	Description
documentSize	object	Detected document size represented by its width and height in millimeters. The values of the width and height are stored in the **documentSize** parameters: • **width** (int) • **height** (int)
documentBoundary	object	Quadrangle containing the whole document. It is represented by an array of its four vertices' coordinates: { **x**: int, **y**: int }. The vertices are indexed clockwise starting from the bottom left. Default: coordinates of a quadrangle, containing the whole image. When document boundary is not detected, this parameter is not returned.

Example of a result JSON.

```
{
  "documentSize": {
    "width": 0,
   "height": 0
  },
  "documentBoundary": [
  {
   "x": 82,
   "y": 567
  },
  {
   "x": 82,
   "y": 567
  },
  {
   "x": 82,
   "y": 567
  },
  {
   "x": 82,
   "y": 567
  }
  ]
}
```

cropImage method

Crops image according to the document boundary and size. Applying this operation removes empty fields around a detected document the and corrects perspective distortion if needed.

```
AbbyyRtrSdk.cropImage(callback, options)
```

Parameters

options

Object specifying parameters of the image crop operation (see Options).

Return values

The method returns result depending on how the image crop finished (see Result).

Options

The table below describes parameters that you can pass as the *options* argument to change image crop settings. Omitting a parameter means that a default setting will be used.

Parameter	Value type	Description
licenseFileName	string	The name of the license file. This file must be located in the **/assets/** directory in your project. Default: "MobileCapture.License".
imageUri	string	**imageUri** (string): Image source for the operation passed as URI. This parameter can be set to the following values: • "Base64": you can pass an encode base64 image string, i.e. "*<base64 encoded image data>*" • "File": image file address, i.e. "*file:///data/user/0/com.abbyy.rtr.reactnative.sample.coreapi/files/pages/page_848b121d-5a7a-4ead-94e6-dd91bc6bfead.jpg*". '*content://*' schema is also supported for Android. Required parameter.
result	object	Result image description, represented by the following parameters: • **compressionLevel** (string): the uniform image compression scale for lossy formats. Valid for export to JPG only. This parameter can be set to the following values: • 'Low' • 'Normal' • 'High' • 'ExtraHigh' Default: 'Low'. • **exportType** (string): result image will be saved to this format. This parameter can be set to the following values: ○ 'Jpg' ○ 'Png' Default: "Jpg". • **destination** (string): result image will be saved to corresponding file ("File") or returned as encode base64 image string ("Base64"). Default: "Base64". • **filePath** (string): full path to the exported file. This parameter is used only if the **destination** is 'File'. Otherwise the value will be ignored. In case the **destination** is 'File' and this parameter is not set, the path will be generated automatically.
documentSize	string	Document size represented by its width and height in

Parameter	Value type	Description
		millimeters. The values of the width and height are set to the **documentSize** parameters: • **width** (float) • **height** (float) I.e., {x: 210; y: 297} for A4 document size. To leave the document size undefined, set both width and height to 0. In this case document of any size will be detected.
documentBoundary	object	Quadrangle containing the whole document. It is represented by an array of its four vertices' coordinates: { **x**: int, **y**: int }. The vertices are indexed clockwise starting from the bottom left. Default: coordinates of a quadrangle, containing the whole image.

Result

This section describes the object that represents image capture results. Returned parameters depend on the scenario.

Parameter	Value type	Description
imageUri	string	Cropped image returned as URI. This image corresponds to the description in the passed **result** parameter.
resolution	object	The image resolution as calculated from image size and physical page size.
imageSize	object	Cropped image size represented by its width and height in pixels. The values of the width and height are stored in the following parameters: • **width** (int) • **height** (int)

Example of a result JSON.

```
{
  "imageUri": "file://data/user/0/com.abbyy.rtr.cordova.sample/files/
{guid}.jpg",
```

```
  "resolution": {
   "x": 0,
   "y": 0
  },
  "imageSize": {
    "width": 1966,
   "height": 3474
  }
}
```

rotateImage method

Rotates image by specified angle.

```
AbbyyRtrSdk.rotateImage(callback, options)
```

Parameters

options

Object specifying parameters of the image rotation (see Options).

Return values

The method returns result depending on how the scenario finished (see Result).

Options

The table below describes parameters that you can pass as the *options* argument to change image rotation settings. Omitting a parameter means that a default setting will be used.

Parameter	Value type	Description
licenseFileName	string	The name of the license file. This file must be located in the **/assets/** directory in your project. Default: "MobileCapture.License".
imageUri	string	**imageUri** (string): Image source for the operation passed as URI. This parameter can be set to the following values: • "Base64": you can pass an encode base64 image string, i.e. "*<base64 encoded image data>*" • "File": image file address, i.e. "*file:///data/user/0/com.abbyy.rtr.reactnative.sample.coreapi/files/pages/page_848b121d-5a7a-4ead-94e6-*

Parameter	Value type	Description
		dd91bc6bfead.jpg". '*content://*' schema is also supported for Android. Required parameter.
angle	int	The angle in degrees. Available values of the angle: 0, 90, 180, 270.
result	object	Result image description, represented by the following parameters: • **compressionLevel** (string): the uniform image compression scale for lossy formats. Valid for export to JPG only. This parameter can be set to the following values: • 'Low' • 'Normal' • 'High' • 'ExtraHigh' Default: 'Low'. • **exportType** (string): result image will be saved to this format. This parameter can be set to the following values: ○ 'Jpg' ○ 'Png' Default: "Jpg". • **destination** (string): result image will be saved to corresponding file ("File") or returned as encode base64 image string ("Base64"). Default: "File". • **filePath** (string): full path to the exported file. This parameter is present if the destination is 'File'.

Result

This section describes the object that represents image capture results. Returned parameters depend on the scenario.

Parameter	Value type	Description
imageUri	string	Rotatedimage returned as URI. This image corresponds to the description in the passed **result** parameter.
size	object	Rotated image size represented by its width and height in pixels. The values of the width and height are stored in the

Parameter	Value type	Description
		following parameters: • **width** (int) • **height** (int)

Example of a result JSON.

```
{
  "imageUri": "file://data/user/0/com.abbyy.rtr.cordova.sample/files/
{guid}.jpg",
  "imageSize": {
    "width": 1966,
   "height": 3474
  }
}
```

assessQualityForOcr method

Estimates if an image quality is suitable for OCR. The whole image is represented as a set of rectangles. A type of rectangle can be either text or unknown. The quality assessment of the image for OCR is calculated based on the rectangles collection.

Note: *This is a technology preview feature. The functionality will be improved and completed in future versions.*

```
AbbyyRtrSdk.assessQualityForOcr(callback, options)
```

Parameters

options

Object specifying parameters of the image quality assessment (see Options).

Return values

The method returns result depending on how the scenario finished (see Result).

Options

The table below describes parameters that you can pass as the *options* argument to change image quality assessment settings. Omitting a parameter means that a default setting will be used.

Parameter	Value type	Description
licenseFileName	string	The name of the license file. This file must be located in the **/assets/** directory in your project. Default: "MobileCapture.License".
imageUri	string	Image source for the operation passed as URI. This parameter can be set to the following values: • "Base64": you can pass an encode base64 image string, i.e. "*<base64 encoded image data>*" • "File": image file address, i.e. "*file:///data/user/0/com.abbyy.rtr.reactnative.sample.coreapi/files/pages/page_848b121d-5a7a-4ead-94e6-dd91bc6bfead.jpg*". '*content://*' schema is also supported for Android. Required parameter.
documentBoundary	object	Quadrangle containing the whole document. It is represented by an array of its four vertices' coordinates: { **x**: int, **y**: int }. The vertices are indexed clockwise starting from the bottom left. Default: coordinates of a quadrangle, containing the whole image.

Result

This section describes the object that represents image capture results. Returned parameters depend on the scenario.

Parameter	Value type	Description
qualityAssessmentForOcrBlocks	object[]	Collection of rectangles with corresponding quality assessment for each of them. Each rectangle can have the following parameters: • **type** (string): rectangle type for quality for OCR estimating. Can be 'Text' or 'Unknown'. • **quality** (int): for blocks with 'Text' **type** this parameter stores a value from 0 to 100 that indicates suitability of the text for OCR. • **rect** (object): bounding rectangle for quadrangle. The rectangle is set by the following parameters: • **top** (int): upper left point ordinate

Parameter	Value type	Description
		• **bottom** (int): bottom left point ordinate • **left** (int): upper left point abscissa • **right** (int): upper right point abscissa

Example of a result JSON.

```
{
  "qualityAssessmentForOcrBlocks": [
    {
     "type": "Text",
     "quality": 2,
     "rect": {
       "top": 1,
      "bottom": 65,
      "left": 65,
      "right": 129
     }
   }
  ]
}
```

exportImage method

Exports an image to JPG or PNG format. Please note, that this method exports a single image only.

```
AbbyyRtrSdk.exportImage(callback, options)
```

Parameters

options

Object specifying parameters of the image export (see Options).

Return values

The method returns result depending on how the scenario finished (see Result).

Options

The table below describes parameters that you can pass as the *options* argument to change image export settings. Omitting a parameter means that a default setting will be used.

Parameter	Value type	Description
licenseFileName	string	The name of the license file. This file must be located in the **/assets/** directory in your project. Default: "MobileCapture.License".
imageUri	string	**imageUri** (string): Image source for the operation passed as URI. This parameter can be set to the following values: • "Base64": you can pass an encode base64 image string, i.e. "*<base64 encoded image data>*" • "File": image file address, i.e. "*file:///data/user/0/com.abbyy.rtr.reactnative.sample.coreapi/files/pages/page_848b121d-5a7a-4ead-94e6-dd91bc6bfead.jpg*". '*content://*' schema is also supported for Android. Required parameter.
result	object	Result image description, represented by the following parameters: • **compressionLevel** (string): the uniform image compression scale for lossy formats. Valid for export to JPG only. This parameter can be set to the following values: • 'Low' • 'Normal' • 'High' • 'ExtraHigh' Default: 'Low'. • **exportType** (string): result image will be saved to this format. This parameter can be set to the following values: ○ 'Jpg' ○ 'Png' Default: "Jpg". • **destination** (string): result image will be saved to corresponding file ("File") or returned as encode base64 image string ("Base64"). Default: "Base64". • **filePath** (string): full path to the exported file. This parameter is present if the destination is 'File'.

Result

This section describes the object that represents image capture results. Returned parameters depend on the scenario.

Parameter	Value type	Description
imageUri	string	Exported image returned as URI. This image corresponds to the description in the passed **result** parameter.
imageSize	object	Exported image size represented by its width and height in pixels. The values of the width and height are stored in the following parameters: • **width** (int) • **height** (int)

Example of a result JSON.

```
{
  "imageUri": "file://data/user/0/com.abbyy.rtr.cordova.sample/files/
{guid}.jpg",
  "imageSize": {
    "width": 1966,
   "height": 3474
  }
}
```

exportImagesToPdf method

Exports images to PDF format. An array of images can be exported, see options description for details.

```
AbbyyRtrSdk.exportImagesToPdf(callback, options)
```

Parameters

options

Object specifying parameters of the image export to PDF (see Options).

Return values

The method returns result depending on how the scenario finished (see Result).

Options

The table below describes parameters that you can pass as the *options* argument to change image export to PDF settings. Omitting a parameter means that a default setting will be used.

Parameter	Value type	Description
licenseFileName	string	The name of the license file. This file must be located in the **/assets/** directory in your project. Default: "MobileCapture.License".

images	object[]	Array of images for export with corresponding information. Each image can have the following parameters:
		• **pageSize** (object): image size represented by its width and height in points (1/72 of an inch). I.e., a page size of A4 is 595x842.
		If both width and height are set to 0, image size will be detected in pixels.
		The values of the width and height are stored in corresponding width and height parameters:
		  • **width** (int)
		  • **height** (int)
		• **compressionLevel** (string): the uniform image compression scale for lossy formats. Valid for export to JPG only. This parameter can be set to the following values:
		• 'Low'
		• 'Normal'
		• 'High'
		• 'ExtraHigh'
		
		Default: 'Low'.
		• **imageUri** (string): Image source for the operation passed as URI. This parameter can be set to the following values:
		• "Base64": you can pass an encode base64 image string, i.e. "*<base64 encoded image data>*"
		• "File": image file address, i.e. "*file:///data/user/0/com.abbyy.rtr.reactnative.sample.coreapi/files/pages/page_848b121d-5a7a-4ead-94e6-dd91bc6bfead.jpg*".
		'*content://*' schema is also supported for Android.
		
		Required parameter.

result	object	Result PDF file description, represented by the following parameters: • **destination** (string): result image will be saved to corresponding file ("File") or returned as encode base64 image string ("Base64"). Default: "Base64". • **filePath** (string): full path to the exported file. This parameter is used only if the **destination** is 'File'. Otherwise the value will be ignored. In case the

Parameter	Value type	Description
		destination is 'File' and this parameter is not set, the path will be generated automatically.
pdfInfo	object	Extra information corresponding to the PDF file. This object can have the following parameters: • **title** (string) • **subject** (string) • **keywords** (string) • **author** (string) • **company** (string) • **creator** (string) • **producer** (string)

Result

This section describes the object that represents image capture results. Returned parameters depend on the scenario.

Parameter	Value type	Description
pdfUri	string	Result URI, corresponding to the description in the passed **result** parameter. If the PDF file destination was 'File', this parameter value corresponds to the **filePath** parameter value with '*file://*' prefix.

Example of a result JSON.

```
{
  "pdfUri": "file://data/user/0/com.abbyy.rtr.cordova.sample/files/
{guid}.pdf"
}
```

Result status

Result status in ABBYY Mobile Capture SDK is an estimate of how stable the result is, and whether it is likely to be improved by recognizing more frames. It is not recommended to use the recognition result in any way if its status is below "Available".

Name	Description
NotReady	No content available.
Tentative	Content detected on a single frame.
Verified	Content verified: matching content found in at least two frames.
Available	Matching content found in three or more frames. The content is recognized and the result is available, though the result can still vary with the addition of new frames.
TentativelyStable	The result has been stable in the last two frames.
Stable	The result has been stable in the last three or more frames.

Troubleshooting

This section contains ready decisions for troublesome cases.

If your iOS application build fails ...

If your iOS application build fails, it may mean that some build phases were not added correctly. In your Xcode project in the **Build Phases**, there are two separate phases, running the following scripts:

- copy_assets.py
- copy_frameworks.sh

In case there are no such phases, from the command line add the Cordova Plugin once more:

```
cordova plugin add cordova-plugin-abbyy-rtr-sdk
```

Now your XCode project is setup correctly.

Copyright and Trademark Notices

Cordova iOS

Apache License
Version 2.0, January 2004
http://www.apache.org/licenses/

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

1. Definitions.

"License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

"Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

"Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

"Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

"Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.

3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.

4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

(a) You must give any other recipients of the Work or

Derivative Works a copy of this License; and

(b) You must cause any modified files to carry prominent notices stating that You changed the files; and

(c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and

(d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the

appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.

8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

END OF TERMS AND CONDITIONS

ADDITIONAL LICENSES:

```
================================================================================
===========
CordovaLib/classes/NSData+Base64.*
================================================================================
===========

//  Created by Matt Gallagher on 2009/06/03.
//  Copyright 2009 Matt Gallagher. All rights reserved.
//
//  This software is provided 'as-is', without any express or implied
//  warranty. In no event will the authors be held liable for any damages
//  arising from the use of this software. Permission is granted to anyone to
//  use this software for any purpose, including commercial applications, and to
//  alter it and redistribute it freely, subject to the following restrictions:
//
//  1. The origin of this software must not be misrepresented; you must not
//     claim that you wrote the original software. If you use this software
//     in a product, an acknowledgment in the product documentation would be
//     appreciated but is not required.
//  2. Altered source versions must be plainly marked as such, and must not be
//     misrepresented as being the original software.
//  3. This notice may not be removed or altered from any source
//     distribution.

================================================================================
```

==========
bin/node_modules/shelljs:
==========
Copyright (c) 2012, Artur Adib <aadib@mozilla.com>
All rights reserved.

You may use this project under the terms of the New BSD license as follows:

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions are met:
* Redistributions of source code must retain the above copyright
notice, this list of conditions and the following disclaimer.
* Redistributions in binary form must reproduce the above copyright
notice, this list of conditions and the following disclaimer in the
documentation and/or other materials provided with the distribution.
* Neither the name of Artur Adib nor the
names of the contributors may be used to endorse or promote products
derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
ARE DISCLAIMED. IN NO EVENT SHALL ARTUR ADIB BE LIABLE FOR ANY
DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES
(INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES;
LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND
ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
(INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF
THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Cordova Android

Apache Cordova
Copyright 2015-2020 The Apache Software Foundation

This product includes software developed at
The Apache Software Foundation (http://www.apache.org/).

Apache License
Version 2.0, January 2004
http://www.apache.org/licenses/

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

1. Definitions.

"License" shall mean the terms and conditions for use, reproduction,
and distribution as defined by Sections 1 through 9 of this document.

"Licensor" shall mean the copyright owner or entity authorized by
the copyright owner that is granting the License.

"Legal Entity" shall mean the union of the acting entity and all
other entities that control, are controlled by, or are under common
control with that entity. For the purposes of this definition,

"control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

"Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

"Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.

3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.

4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

   (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and

   (b) You must cause any modified files to carry prominent notices stating that You changed the files; and

   (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and

   (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

   You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with

the conditions stated in this License.

5. Submission of Contributions. Unless You explicitly state otherwise,
   any Contribution intentionally submitted for inclusion in the Work
   by You to the Licensor shall be under the terms and conditions of
   this License, without any additional terms or conditions.
   Notwithstanding the above, nothing herein shall supersede or modify
   the terms of any separate license agreement you may have executed
   with Licensor regarding such Contributions.

6. Trademarks. This License does not grant permission to use the trade
   names, trademarks, service marks, or product names of the Licensor,
   except as required for reasonable and customary use in describing the
   origin of the Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty. Unless required by applicable law or
   agreed to in writing, Licensor provides the Work (and each
   Contributor provides its Contributions) on an "AS IS" BASIS,
   WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or
   implied, including, without limitation, any warranties or conditions
   of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A
   PARTICULAR PURPOSE. You are solely responsible for determining the
   appropriateness of using or redistributing the Work and assume any
   risks associated with Your exercise of permissions under this License.

8. Limitation of Liability. In no event and under no legal theory,
   whether in tort (including negligence), contract, or otherwise,
   unless required by applicable law (such as deliberate and grossly
   negligent acts) or agreed to in writing, shall any Contributor be
   liable to You for damages, including any direct, indirect, special,
   incidental, or consequential damages of any character arising as a
   result of this License or out of the use or inability to use the
   Work (including but not limited to damages for loss of goodwill,
   work stoppage, computer failure or malfunction, or any and all
   other commercial damages or losses), even if such Contributor
   has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability. While redistributing
   the Work or Derivative Works thereof, You may choose to offer,
   and charge a fee for, acceptance of support, warranty, indemnity,
   or other liability obligations and/or rights consistent with this
   License. However, in accepting such obligations, You may act only
   on Your own behalf and on Your sole responsibility, not on behalf
   of any other Contributor, and only if You agree to indemnify,
   defend, and hold each Contributor harmless for any liability
   incurred by, or claims asserted against, such Contributor by reason
   of your accepting any such warranty or additional liability.

ADDITIONAL LICENSES:

==========
bin/node_modules/q
==========

Copyright 2009–2017 Kristopher Michael Kowal. All rights reserved.
Permission is hereby granted, free of charge, to any person obtaining a copy
of this software and associated documentation files (the "Software"), to
deal in the Software without restriction, including without limitation the
rights to use, copy, modify, merge, publish, distribute, sublicense, and/or
sell copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in
all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING
FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
IN THE SOFTWARE.

==========
bin/node_modules/nopt
==========

The ISC License

Copyright (c) Isaac Z. Schlueter and Contributors

Permission to use, copy, modify, and/or distribute this software for any
purpose with or without fee is hereby granted, provided that the above
copyright notice and this permission notice appear in all copies.

THE SOFTWARE IS PROVIDED "AS IS" AND THE AUTHOR DISCLAIMS ALL WARRANTIES
WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES OF
MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR
ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES
WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN
ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR
IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

==========
bin/node_modules/which
==========

The ISC License

Copyright (c) Isaac Z. Schlueter and Contributors

Permission to use, copy, modify, and/or distribute this software for any
purpose with or without fee is hereby granted, provided that the above
copyright notice and this permission notice appear in all copies.

THE SOFTWARE IS PROVIDED "AS IS" AND THE AUTHOR DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.
© 2020 GitHub, Inc.

Android Jetpack

Apache License
Version 2.0, January 2004
http://www.apache.org/licenses/

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

1. Definitions.

"License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

"Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

"Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

"Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

"Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes

of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.

3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.

4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

   (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and

   (b) You must cause any modified files to carry prominent notices stating that You changed the files; and

(c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and

(d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.

8. Limitation of Liability. In no event and under no legal theory,

whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

END OF TERMS AND CONDITIONS

APPENDIX: How to apply the Apache License to your work.

To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives.

Copyright 2019 The Android Open Source Project

Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at

http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.

Portions of this software are derivative works based on software developed by ABBYY Production LLC, Copyright (c) 2019 ABBYY Production LLC. All rights reserved.

Software used in the Mobile Capture Sample

Cordova Plugin Social Sharing

The MIT License (MIT)

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Cordova Plugin AndroidX Adapter

The MIT License

Copyright (c) 2019 Dave Alden / Working Edge Ltd.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

All other trademarks and copyrights are the property of their respective owners.