



User Guide for Read-a-Card

Read-a-Card software from v3.3.0

Revised April 2019 v1.03

If you need help to set up or use Read-a-Card, beyond what is contained in this User Guide and the Developer Guide, then please contact our support team.

Before you get in touch it would be helpful if you could check which version of Read-a-Card you are using. You will find this on the **About tab**.

Website: <https://readacard.com/support>

Email: support@read-a-card.com

Telephone UK and Europe: +44 (0) 1428 685861

Telephone North America and Latin America: +1 888-262-9642 or +1 565-262-9642

If you have any feedback on setting up or using Read-a-Card software or this documentation, then please contact our support team. The product is constantly being reviewed and improved and we value feedback about your experience.

Copyright 2019 Dot Origin Ltd. All rights reserved.

No part of this User Guide may be published or reproduced without the written permission of Dot Origin Ltd except for personal use. This User Guide relates to correct use of the Read-a-Card software only. No liability can be accepted under any circumstances relating to the operation of the user's own PC, network or infrastructure.

Dot Origin Ltd

Unit 7, Coopers Place Business Park, Combe Lane, Wormley

Godalming GU8 5SZ United Kingdom

+44 (0) 1428 685861

Contents

1 Getting started with Read-a-Card	1
1.1 Installing Read-a-Card software	2
1.2 Installing your card reader	2
1.3 Licensing Read-a-Card	3
1.4 Starting Read-a-Card	4
1.5 Start up interface	5
1.6 Tool tray icons	5
1.7 Status tab	6
2 Simple applications	7
2.1 Default - Keyboard wedge functionality	7
2.2 Insert RFID card ID into Word or Excel example	8
2.3 Launch URL when a card is presented example	9
2.4 Log contactless card information example	10
2.5 Run command example	11
2.6 NFC Smart Poster example	12
3 Configuration through the user interface	13
3.1 Card Type tab	13
3.2 Reader Type tab	14
3.3 Action tab	15
3.4 Format tab	16
3.5 Logging tab	17
3.6 Web Server tab	18
3.7 Lock tab	19
3.8 Settings tab	20
3.9 About tab	21
3.10 Obtaining updates	21
4 Advanced features	22
4.1 Using Read-a-Card with other applications	22
4.2 Accessing program settings directly	22
4.3 Logging timestamp format	22
4.4 Run scripts and open URLs	24
4.5 Keyboard buffer templates	24
4.6 Enable web server	24
4.7 Lock settings	24
4.8 Logging structures	24
4.9 Using Read-a-Card plug-ins	25



A	NFC Decoder	A-1
	A.1 Configuration of the NFC plug-in	A-1
	A.2 Set up required actions	A-1
B	Paxton Net2 Compatible Decoder	B-1
C	DESFire Decoder	C-1
	C.1 Configuration of DESFire Decoder plug-in	C-1
	C.2 Modes of operation	C-2
D	MIFARE Sector Decoder	D-1
	D.1 Configuration of MIFARE Sector Decoder plug-in	D-1
	D.2 Sector data configuration per card type	D-2
	D.3 MIFARE access keys	D-4
	D.4 Decoding facility codes and card numbers	D-5
	D.5 Read-a-Card actions	D-8
	D.6 MIFARE Sector Decoder example configuration file	D-9
E	Custom Format Decoder	E-1
	E.1 Configuration of Custom Format Decoder plug-in	E-1
	E.2 Configuration for different card types	E-2



1 Getting started with Read-a-Card

Read-a-Card is a software utility that runs on a Windows PC and reads information from many types of contactless cards and tags. Card numbers and unique IDs can be read and logged to file, or automatically sent to other software applications. This includes typing the card ID, known as a keyboard wedge.

Read-a-Card is compatible with a wide range of readers from different manufacturers, and can also be used to identify cards, test reader functionality and manage the use of multiple readers on a single PC.

This user guide is to support an integrator or administrator with initial Read-a-Card set up.

Read-a-Card is designed to be configured by an integrator or administrator using its tab-based user interface. Read-a-Card is then typically run in the background by an end-user, with limited user privileges, to avoid changes to Read-a-Card configuration parameters in normal use.

Developer use of Read-a-Card, to integrate card reading seamlessly into third-party applications, is addressed in a separate [Developer Guide](#).



1.1 Installing Read-a-Card software

1. Insert your Read-a-Card CD or USB stick and locate the **Read-a-Card.msi** file in the Setup folder

Or

Visit readacard.com/update to download the latest version from the Read-a-Card website.

2. Double-click or open the **Read-a-Card.msi** file to start the installation, and follow the on-screen instructions until complete.

Note: Only administrators can install and configure Read-a-Card, while all types of user can run the software.

1.2 Installing your card reader

You will need to install drivers for your card reader before it can be used. These often install automatically via plug-and-play. If not, you may need to locate them on the Read-a-Card CD or USB stick, or the manufacturer's web site.

1.2.1 Currently supported readers

Read-a-Card has been tested to work with various readers that conform with the PC/SC 2.0 standard, as well as other PC/SC 1.0 and proprietary devices. Please check the [latest compatibility list](#) on the Read-a-Card website.



1.3 Licensing Read-a-Card

Read-a-Card can be licensed in various ways. The options available depend on your choice of card reader.

- **Reader-based license:** With these options Read-a-Card is licensed for use on any PC, with as many readers as you need, as long as one licensed reader is connected.
 - **SAM license** - uses a SIM-sized smart card that fits inside a compatible SIM reader, or a compatible contactless reader with a SAM slot. A SAM license is the most flexible option, because the SAM can be moved from one reader or PC to another, but it only suits a limited number of card readers.
 - **e-license** - works for any reader with an electronic serial number or ID, which is the case for most OMNIKEY, Identiv, SCM and Gemalto readers. (With Read-a-Card in trial mode, the Reader ID will show on the **About tab** after reading a card, if your reader has an electronic ID.) E-licenses can be requested online, using the [Order license](#) button the **About tab**, but can also be supplied in a file for offline activation.
- **PC-based license:** is the best option if you want to use many different readers, or your preferred reader does not support other licensing methods. After you scan the activation card or label supplied, using a standard 13.56MHz card reader attached to the PC running Read-a-Card, the Read-a-Card software will be licensed for use on that PC . If it needs to be moved to another PC, there is a procedure for removing and re-licensing your software.

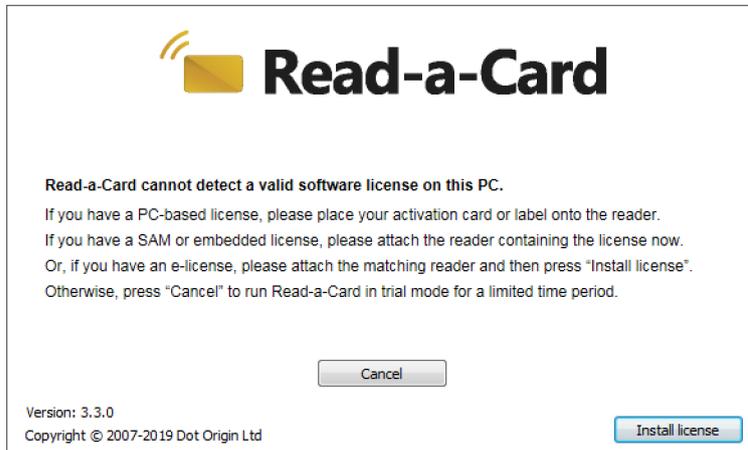
Note: If you have a PC-based license, do keep your software activation card or label safe. You will need this if you ever want to move the Read-a-Card PC-based software license to a new PC.



1.4 Starting Read-a-Card

1. After **Installing Read-a-Card software**, attach your reader to the PC.

2. To run the Read-a-Card software either double-click the Read-a-Card icon  on the Windows desktop, or select the program in the Windows Start menu.



3. Follow the on-screen instructions for your license type.

If you have a reader-based e-license:

- Use the  button.
- Choose Fetch license from server and then use .

You can repeat this process on any number of PCs, but the licensed reader must always be attached to the PC when Read-a-Card is run.

Note: To activate Read-a-Card with a non-PC/SC-compliant reader (such as the ACS ACR120):

- Use the  button and enter trial mode initially.
- On the **Reader Type tab** select the appropriate reader type.
- Then, close Read-a-Card.

The next time you run Read-a-Card you will be able to use your reader for software activation.

If you have a PC-based license:

- Present your activation card or tag, which is supplied in the box, to the reader.

When complete, keep the activation card or tag safe – you will need it if you ever move the software to another PC!

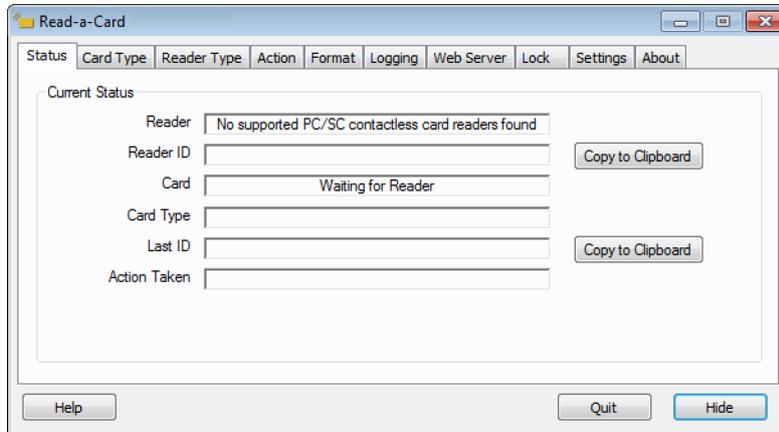
If you have a reader-based SAM license:

- If you received a SIM-sized smart card license, insert it into a compatible SIM reader or contactless reader with SAM slot. (If you bought the license and reader together your reader or USB token may already have the SAM inside.) This reader must be connected to your PC.
- Read-a-Card will automatically detect the SAM license in an attached reader whenever it is run.



1.5 Start up interface

When started, Read-a-Card will open displaying the **Status tab** by default, unless the ‘Start Read-a-Card hidden (in tooltray)’ option has been chosen on the **Settings tab**.



Select the minimise or **Hide** button at any time to close the Read-a-Card main window, but leave Read-a-Card active, with its icon in the tool tray.

Use the **Quit** button to close the application completely.

1.6 Tool tray icons

Read-a-Card uses different tool tray icons to display the card reader status while the program is hidden.

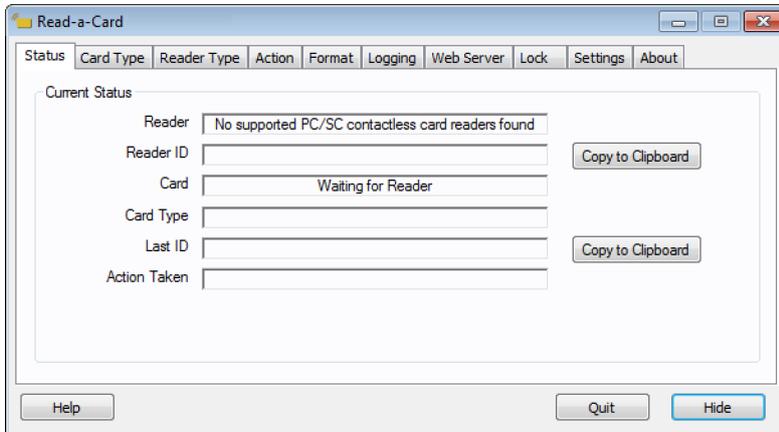
Read-a-Card can be configured to start automatically with Windows and/or start minimised in the tool tray. Set this up on the **Settings tab**. (This feature is not available in trial mode.)

	No card reader detected
	Card reader ready to scan a card
	Card reader has successfully detected one or more cards
	Card reader has detected a card type that should cause an action (chosen on the Card Type tab) and has taken the configured action (defined on the Action tab).



1.7 Status tab

The Status tab displays information about the current reader and the most recent card detected.



- **Reader** is the name of the reader detected by Read-a-Card. If no reader can be found, a message will be displayed.
- **Reader ID** is the unique reader serial number, if the reader supports this feature. The ID will only appear after a card is presented.
- **Card** displays a message indicating when a card has been detected by the reader.
- **Card Type** displays the type of card that has been detected by the reader. If the type is not known by Read-a-Card it will display 'Generic Card'.
- **Last ID** displays the unique ID of the last card detected by the reader.
- **Action Taken** displays the action(s) that were taken when the last card was presented. These actions are set up on the **Action tab**.

The ID of the reader or the last card read can be copied to the clipboard, by pressing the **Copy to Clipboard** button. This can be useful if you want to paste the data manually into another application or document.



2 Simple applications

Read-a-Card supports many different card reading applications. This section describes a few simple examples you might want to set up.

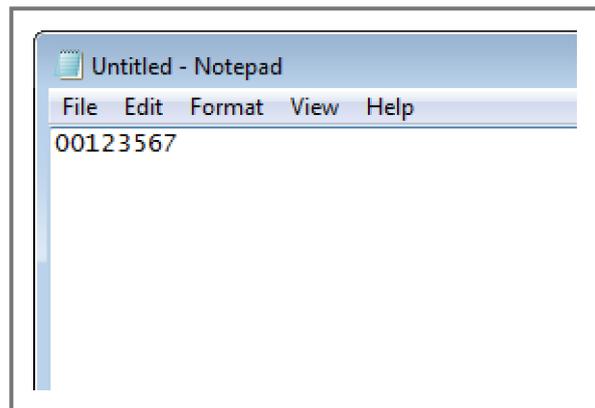
Developer use of Read-a-Card, to integrate card reading seamlessly into third-party applications, is addressed in a separate [Developer Guide](#).

2.1 Default - Keyboard wedge functionality

When the software is first installed, the keyboard wedge feature is enabled by default. Any card ID that is read will be typed automatically into other applications, at the current cursor location.

To try this out:

- a. Open up another Windows application, such as Notepad.
- b. Present a suitable card to your reader.
- c. If the other application is on top and the text cursor is displayed ready to receive typed characters, the ID of a card will appear there when a card is presented to your reader.



This feature is enabled on the:

1. **Action tab** - select Place ID in keyboard buffer. (This is the default).
2. **Format tab** - adjust the exact format of the ID using the options on this tab.

Note: By default, the card will not be read a second time if the same card is detected again inside 10 seconds. If this does not suit your application, this 'de-dupe' feature can be modified on the **Settings tab**.

Depending on the application, you may need to select a particular field before the card is presented for this to work correctly. This can be controlled using custom prefix and suffix characters, as in the **Insert RFID card ID into Word or Excel example**. More advanced integration options are explained in the following sections and the [Developer Guide](#).



2.2 Insert RFID card ID into Word or Excel example

If the end-user positions the cursor in a Word or Excel document, then presents a card to a reader, with Read-a-Card set up in this way, they will see card data inserted at their chosen cursor position.

This could be used if you are issuing ID cards to students and keeping an Excel record of which card was issued to which student, as in this example.

	A	B	C
1	User	Course	Card Issued
2	Etheridge, Peter	Med	34506
3	Pettifer, Rose	Sci	12534
4	Mayhew, Dan	Sci	
5			
6			
7			
8			

Steps to set this up in Read-a-Card:

1. **Action tab** - use the default setting Place ID in keyboard buffer.
2. **Format tab** - control how data are read from the card, and the format in which they should be inserted into a document. This can include anything needed before or after the data, as a Keyboard Prefix or Keyboard Suffix. In this example, choose Keyboard Suffix 'CR' for a carriage return after the card ID. This will enter the data into Excel then move the cursor to the next Card Issued cell.

In other situations you might select Advanced Keyboard Format and define additional words, characters, tabs or carriage returns using the **Advanced Keyboard Format insert codes** listed below. Use this to minimise the manual input needed from end-users with each card read.

2.2.1 Advanced Keyboard Format insert codes

%n card ID	\r carriage return
%t card type	\t tab
%i reader ID	\n new line
%r reader name	\e enter
%c facility code	\\ is \
%u card number	\a ALT applied to next key
%f iClass format	\c CTRL applied to next key
%x card unique ID	\s SHIFT applied to next key
\xHH to insert a non-printable character where HH is the ASCII code in hex	

Example Advanced Keyboard Format insert codes:

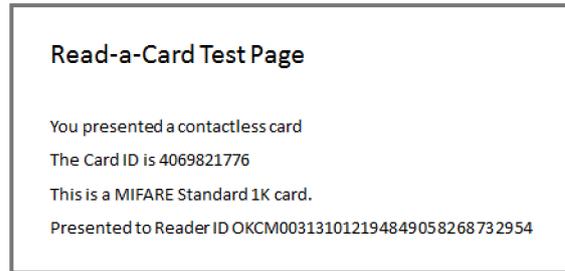
CardID=%n Card Type=%t Reader ID=%i iClassFormat=%f\r will include words to identify the data elements then move to the next line.

%n\t%i\r will put card ID in one cell, reader ID in the next cell then move to the next line.



2.3 Launch URL when a card is presented example

When an end-user presents a card to a reader, with Read-a-Card set up in this way, a web page will be opened and populated with data from the presented card.



Steps to set this up in Read-a-Card:

1. **Card Type tab** - change the default from All Cards, if you only want Read-a-Card to react to certain card types.
2. **Reader Type tab** - change the default from All supported PC/SC Contactless Card Readers only if you want to restrict the readers to use.
3. **Action tab** - choose Launch URL (in web browser).

Enter a URL, including any elements to be read from the card by using the **URL launch insert codes** listed below.

If preferred, you can call a URL directly from Read-a-Card using HTTP, which will result in a 'silent' request. For this choose Request URL via HTTP (without launching browser) instead.

4. **Format tab** - control how data should be read from the card.

2.3.1 URL launch insert codes

%n card ID	%u card number
%t card type	%f iClass format
%i reader ID	%s timestamp
%r reader name	%x card unique ID
%c facility code	

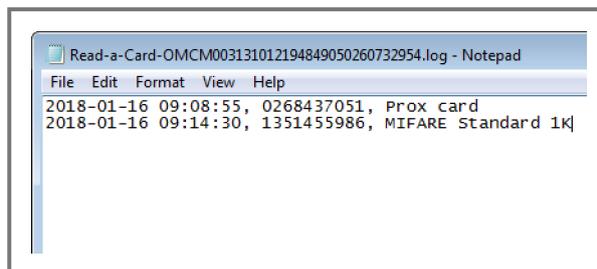
Example URL including URL insert codes:

`http://www.example.com/read.html?ID=%nType=%t&Rdr=%i` includes data elements read from the card that then affect how the web page is populated when it opens.



2.4 Log contactless card information example

When an end-user presents a card to a reader, with Read-a-Card set up in this way, a log will be populated with data from the presented card.



Steps to set this up in Read-a-Card:

1. **Card Type tab** - change the default from All Cards, if you only want Read-a-Card to react to certain card types.
2. **Reader Type tab** - change the default from All supported PC/SC Contactless Card Readers only if you want to restrict the readers to use.
3. **Action tab**- use the default setting Place ID in keyboard buffer, and optionally Play sound to give confirmation a card has been read.
4. **Format tab** - control how data should be read from the card.
5. **Logging tab** - choose Log card entry to file and enter a path to specify the required log file, possibly extended with additional **Log file name insert codes**. Then choose whether each new card presented should Append to log or Replace log, whether you need CSV or XML format. Select the data items to log each time, choosing from Card Type, Reader ID Number, Timestamp and iClass Format (if available).

2.4.1 Log file name insert codes

%n card ID	%u card number
%t card type	%f iClass format
%i reader ID	%s timestamp
%r reader name	%x card unique ID
%c facility code	

Example log file name and path making use of these insert codes:

c:\Read-a-Card\%s\Read-a-Card-%i will allow you to create a series of new log files. Each one will be named as originating from a particular reader ID and filed in a folder labelled with the event timestamp.

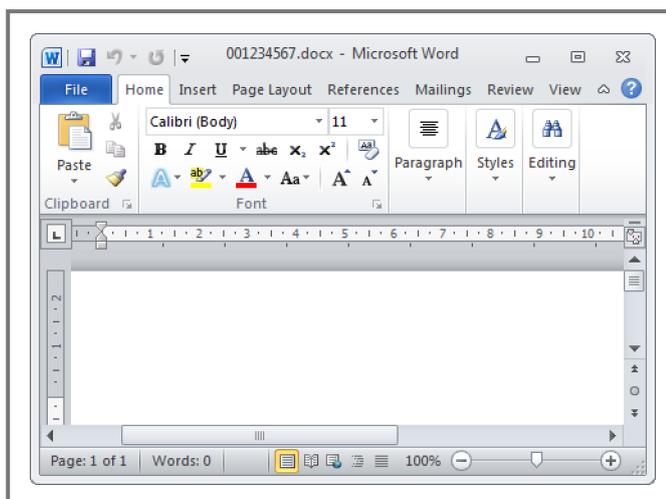


2.5 Run command example

When an end-user presents a card to a reader, with Read-a-Card set up in this way, it can trigger a document to open, named with your reader ID, or run a batch file based on the presented card information.

Steps to set this up in Read-a-Card:

1. **Card Type tab** - change the default from All Cards, if you only want Read-a-Card to react to certain card types.
2. **Reader Type tab** - change the default from All supported PC/SC Contactless Card Readers only if you want to restrict the readers to use.
3. **Action tab** - choose Run command and identify the file that should be opened, together with any card information to be passed at this time. This file name can include the **Run command insert codes** shown below. A file name of `c:\Read-a-Card\%i.docx` will open a Word document, named with your reader ID.
4. **Format tab** - control how data should be read from the card.



2.5.1 Run command insert codes

%n card ID	%u card number
%t card type	%f iClass format
%i reader ID	%s timestamp
%r reader name	%x card unique ID
%c facility code	

Example file names using these insert codes:

`c:\Read-a-Card\%i.docx` will allow you to open a series of new Word files. Each named as originating from the reader ID of the presented card.

`c:\Read-a-Card\example.bat %n %t %i %f` will allow you to run a batch file expecting the parameters card ID, card type, reader ID and iClass format.



2.6 NFC Smart Poster example

If the end-user presents a Smart Poster tag to the reader, with Read-a-Card configured in this way, the appropriate web page will be displayed.

The steps are:

1. **Card Type tab** - change the default from All Cards, if you only want Read-a-Card to react to certain card types.
2. **Reader Type tab** - change the default from All supported PC/SC Contactless Card Readers only if you want to restrict the readers to use.
3. **Format tab** - choose 'Plug-in: NFC Decoder' from the Card ID source dropdown list.
4. **Action tab** - choose Launch URL (in web browser), setting the URL to %n.
(When using the NFC Decoder %n represents the URI and %c any content from the Smart Poster text field. These fields can be used in a Launch URL or Run command on this tab, or as part of the Advanced Keyboard Format template on the **Format tab**.)



3 Configuration through the user interface

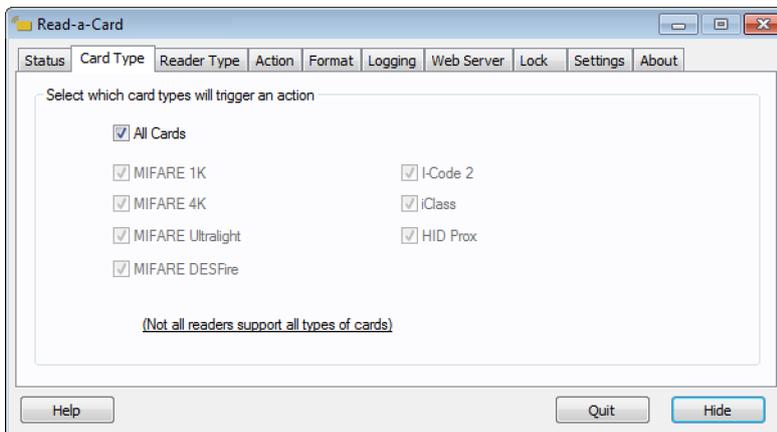
The Read-a-Card interface has several tabs that are used for configuration: **Card Type tab**, **Reader Type tab**, **Action tab**, **Format tab**, **Logging tab**, **Web Server tab** and **Settings tab**. Any changes made to these tabs take effect immediately. All settings are saved automatically.

When you are finished use the **Lock tab** to prevent accidental changes to configuration by the end-user.

Note: Only administrators can install and configure Read-a-Card, while all types of user can run the software.

3.1 Card Type tab

On the Card Type tab you can limit the card types that cause Read-a-Card to take action.



By default All Cards cause actions, but this can be limited so that Read-a-Card only reads certain types by deselecting All Cards and selecting or deselecting the possible types listed.

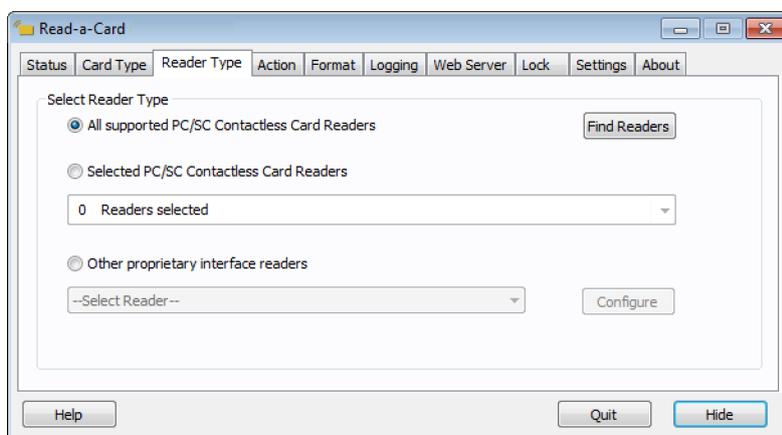
Whatever the selection made here, all cards will still be detected by the reader and displayed on the **Status tab**. The Card Type tab only limits the card types which cause Read-a-Card to take an action, defined on the **Action tab**, such as playing a sound, writing a log, or copying the card ID into another application.

Note: If a card type is greyed-out it cannot be selected. This is because your chosen reader cannot read that card type. Some readers are unable to read certain types of card, or cannot distinguish between certain card types, see **Currently supported readers**.



3.2 Reader Type tab

On the Reader Type tab you can limit which readers Read-a-Card monitors.



By default Read-a-Card monitors All supported PC/SC Contactless Card Readers. This can be changed to monitor card readers that use proprietary drivers or to select or ignore certain PC/SC readers.

Note: Read-a-Card can only detect readers that have been properly installed and are recognised by Windows.

If Selected PC/SC Contactless Card Readers is chosen, the dropdown menu will display currently detected readers, which can then be individually enabled or disabled.

Note: If readers do not have unique serial numbers then all readers of that type will be grouped, and can therefore only be enabled or disabled together. Readers that have unique serial numbers can be individually enabled or disabled.

Use the button to refresh the dropdown list of detected card readers. You may need this if you connect or disconnect a reader while Read-a-Card is running.

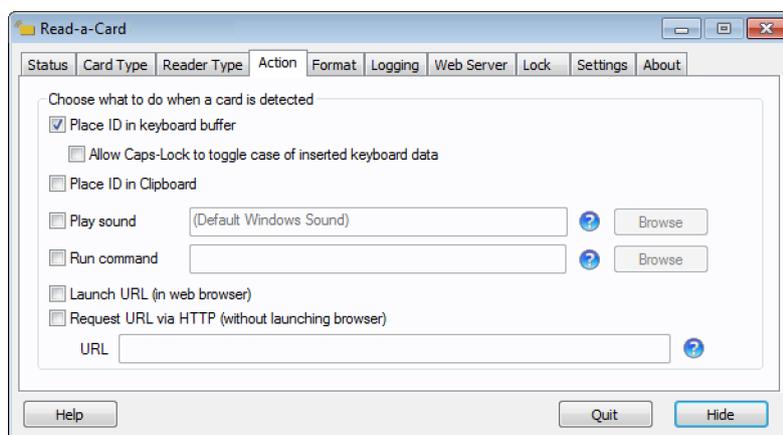
Choosing Other proprietary interface readers allows specific non-PC/SC readers to be used. You can normally select only one proprietary reader at a time.

Note: For GemTag X1010 IP readers you can the port numbers and IP addresses of up to ten readers.



3.3 Action tab

On the Action tab you can define one or more actions for Read-a-Card to take when a card is read.



The default action is to copy the card ID to the keyboard buffer, see **Default - Keyboard wedge functionality**. This feature can be disabled by deselecting Place ID in keyboard buffer.

To maintain consistency of alphanumeric IDs, Caps-Lock will not normally affect this data. There is an option to Allow Caps-Lock key to toggle case of inserted keyboard data if this is needed in your NFC tag application.

Place ID in Clipboard places the ID of the card detected into the Windows clipboard.

Play sound plays a sound when a card is detected. The default Windows beep can be changed to another .wav sound file, by using the button to find a file or typing its location.

Run command allows you to define a command to run (such as open a document) when a card is detected. This command will be invoked by the Windows shell. The reader ID, card type and card ID can be used in your command. (The  button provides a reminder of **Run command insert codes**). For more see the **Run command example** in this guide.

Launch URL (in web browser) allows you to set a URL to launch when a card is detected. The URL will be launched by the Windows shell, using the default web browser. Data read from the card (reader ID, card type, card ID, and iClass/Prox format where applicable) can be used in the URL. (The  button provides a reminder of **URL launch insert codes**). There is a **Launch URL when a card is presented example** in this guide.

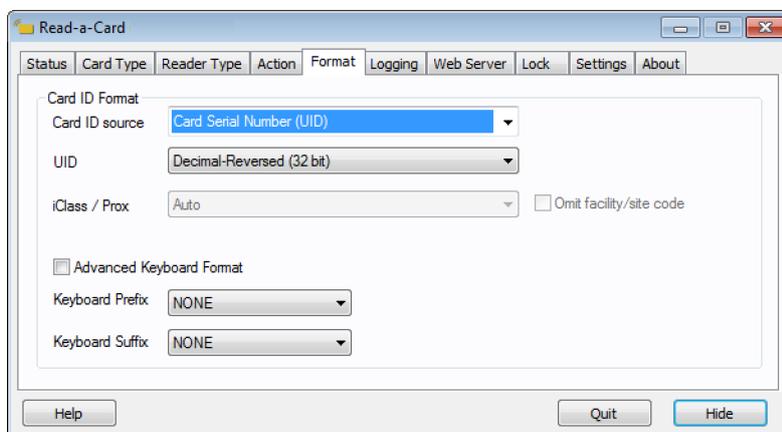
Request URL via HTTP (without launching browser) allows a URL to be sent as a silent HTTP request to a web server. The result of this request, returned by the server, is ignored. This option allows card presentations to be silently logged by a web server.

Note: Options on the **Settings tab** can affect these actions. By default no action will be taken for duplicate card detections by the same reader within 10 seconds.



3.4 Format tab

On the Format tab you can select the card ID source and control the format used when a card ID is inserted in the keyboard buffer.



Card ID source on the **Format tab** lets you select a source of ID information. This defaults to the Card Serial Number (UID). If you select HID iClass / Prox or HID iClass / Prox / Mobile ID, then Read-a-Card will try to read this data, rather than UID. The data obtained will only be valid if your reader can read those credentials. Choose a Read-a-Card plug-in to decode any other type of ID .

Some plug-ins will require further configuration before they can be used. There is more information about **Using Read-a-Card plug-ins** in the Appendices.

UID lets you choose the output format for UIDs. This can be Hex Standard, Hex Reversed, Decimal Standard (32 or 64 bit) or Decimal Reversed (32 or 64 bit).

Note: Be aware you could accidentally truncate data if you pick the wrong UID option, for instance a DESFire 56bit card will need the 64bit option, not 32bit.

If you choose HID iClass / Prox or HID iClass / Prox / Mobile ID as your Card ID source, the iClass / Prox dropdown menu lets you select either the specific HID data format for your cards or 'Auto'. (Although 'Auto' may not be able to distinguish between all card formats in all cases). Select 'Raw data (Hex)' if you just want to examine the format.

Select Omit facility/site code from HID iClass/Prox card IDs, if it is not required in your application.

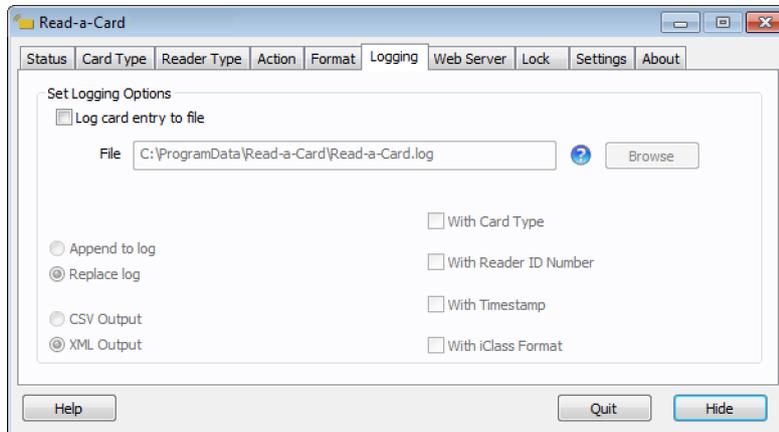
The Keyboard Prefix and Keyboard Suffix dropdown menus let you add extra characters to the keyboard buffer, either before or after each card is presented.

Choosing Advanced Keyboard Format allows you to create a single keyboard format template. Using insert codes you can create any combination of reader ID, card type, card ID, card CSN/UID. You can also include ordinary characters and a variety of key presses. (The  button provides a reminder of **Advanced Keyboard Format insert codes**).



3.5 Logging tab

On the Logging tab you can control the optional logging of card and reader information every time a card is presented.



Logging is enabled by selecting Log card entry to file. Then the log location and file name can be specified. There is a **Log contactless card information example** in this guide.

You can create separate log files per card ID, or reader ID, or timestamp. If you want to include card IDs, reader IDs, card types, and timestamps as part of the log file location or name, the  button provides a reminder of **Log file name insert codes**.

The **Logging timestamp format** can be manually configured in the `Read-a-Card.ini` file (see **Advanced features** section).

The Replace log option will cause each card presentation to overwrite any existing log file of the same file name and location. The Append to log option will add new log entries to the end of the existing file.

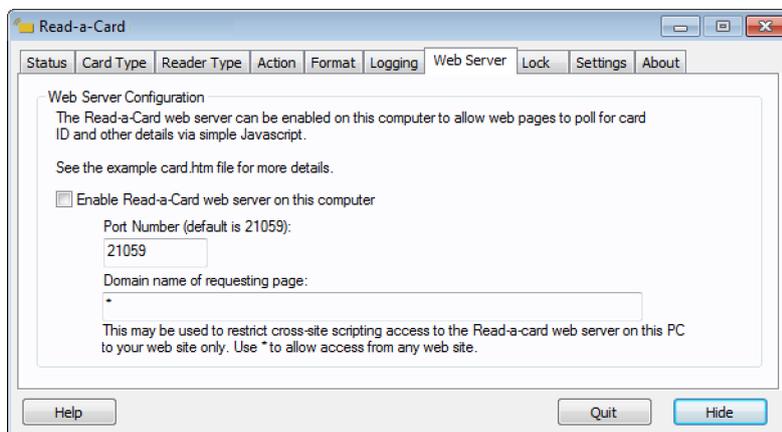
The CSV Output and XML Output options allow you to choose the format of log files.

The card ID will always be logged. Additional content can also be recorded in the log files by selecting some or all of With Card Type, With Reader ID Number, With Timestamp, or With iClass/Prox Format.



3.6 Web Server tab

On the Web Server tab you can configure Read-a-Card's built-in web server so that web pages can read card data through Read-a-Card, when they are viewed in a web browser on the same computer.



This feature provides an easy way for a web-based card enrolment system to autofill its card ID field before submitting the information to a central server.

The web server can be enabled by selecting Enable Read-a-Card web server on this computer.

While enabled, Read-a-Card can communicate with a web page on the specified port number and domain.

Port Number defines the port on which the Read-a-Card web server listens. It defaults to 21059.

By default, the Domain name of requesting page is set to asterisk (*). This means that any web page you visit, which contains the necessary Javascript, can obtain your card details from Read-a-Card when a card is presented. You can restrict that access to a specific website by entering its domain name here.

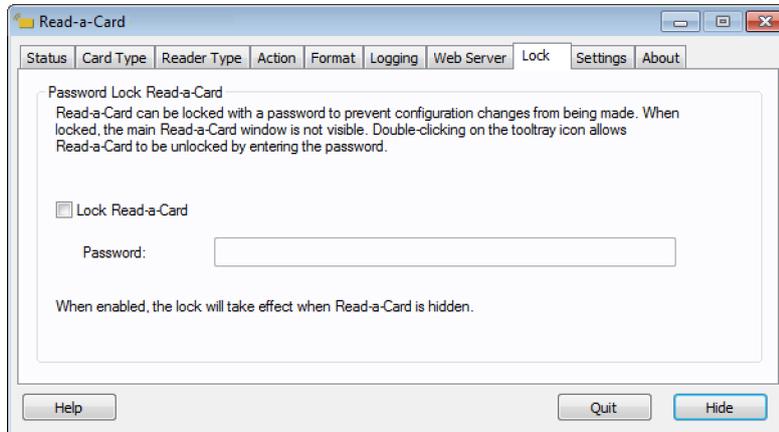
Any changes made on this tab will have an immediate effect on the Read-a-Card web server.

The **Developer Guide** contains more information on setting up a web page to communicate with Read-a-Card, including a `card.htm` example web page. (This example code is available as a file called `Card.htm` installed with the Read-a-Card application, usually in `C:\Program Files (x86)\Read-a-Card.`)



3.7 Lock tab

On the Lock tab, you can password protect your Read-a-Card settings and interface.



This makes it easy to ensure Read-a-Card runs properly and in the background for any end-user of your application. It is not a complete security layer, but it is an easy way to keep end-users from changing the way Read-a-Card behaves accidentally.

You can turn the lock on by selecting Lock Read-a-Card and entering a password.

When the lock is enabled and you choose to the user interface, the password will be required to show the user interface again.

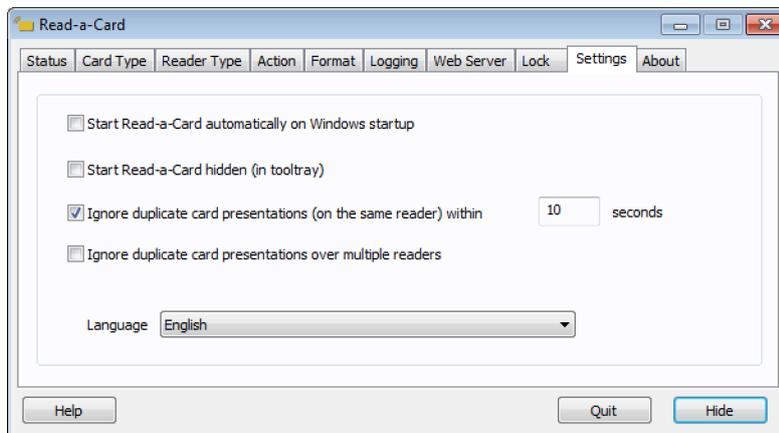
You can then disable the lock by deselecting Lock Read-a-Card.

Note: If your password is lost or forgotten, you can unlock the system by deleting the line in the `Read-a-Card.ini` file that begins with `Password=`. (See **Accessing program settings directly** for more information.)



3.8 Settings tab

Select Start Read-a-Card automatically to start Read-a-Card when Windows starts.



To start Read-a-Card minimised in the tool tray, rather than showing the **Status tab**, select

Start Read-a-Card hidden (in tooltray).

Note: The start Read-a-Card hidden feature is not available in trial mode.

Choose Ignore duplicate card presentations to enable the de-dupe feature and set a time period during which duplicate cards will be ignored. By default this is enabled and the time period is set to 10 seconds. Deselect Ignore duplicate card presentations if this does not suit your application.

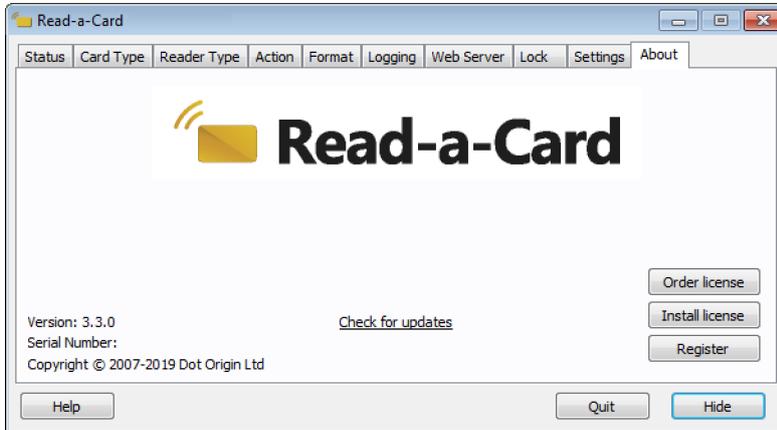
You can extend this by selecting Ignore duplicate card presentations over multiple readers. Then Read-a-Card will disregard duplicate card presentations across **all** readers connected to the computer, for the time period set.

The language dropdown box allows you to set the language you wish Read-a-Card to use. Alternative languages may only be available with certain types of software license.



3.9 About tab

The About tab reports Read-a-Card version number and license details. It allows you to request and install a license, register your product and check for software updates.



You can use the button to request a license for the currently connected reader, and install e-licenses using the button. This is the same as on the start up page, see **Starting Read-a-Card**.

When you don't have a license, the 'Serial Number:' reported here will be the identifier for the currently connected reader, if available. When you have a license, the 'Serial Number:' field reports your license identifier instead.

3.10 Obtaining updates

To check if you are running the latest version of Read-a-Card select the [Check for updates](#) link on the **About tab** or visit the Read-a-Card website at: www.readacard.com/update



4 Advanced features

4.1 Using Read-a-Card with other applications

Read-a-Card provides a number of ways for other applications to make use of Read-a-Card's functionality. There is a Windows Messaging API (application programming interface) and Read-a-Card web server which are explained further in the [Developer Guide](#). Full details of the developer APIs and code samples are available for download from <https://readacard.com/support>.

4.2 Accessing program settings directly

All program settings are saved in a file called `Read-a-Card.ini` which can be manually edited or accessed if required, in this folder:

On Windows XP/2000: `C:\Documents and Settings\All Users\Application Data\Read-a-Card\`

On Windows Vista/7/8/10: `C:\ProgramData\Read-a-Card\`

4.3 Logging timestamp format

The date and time format used by Read-a-Card's logging feature can be configured within the `Read-a-Card.ini` file. The date and time may be optionally used both in the content of the log data and as part of the file or folder name of the log. This feature makes it possible to create discrete log files for each day, month or year by including `%s` in the log file name on the **Logging tab**.

Two `.ini` file settings control the format of the timestamp that will be used. The setting `Timestamp` controls the format of the timestamp used within the content of the log file, while `FileNameTimestamp` controls the format of the timestamp that may be incorporated in the file or folder name. This allows a full date/time timestamp to be recorded for each card entry in a set of log files, where a discrete file is created per date.

The default values for these settings in the `Read-a-Card.ini` file are as follows:

```
[LOG]
Timestamp = %Y-%m-%d %H:%M:%S
FileNameTimestamp = %Y-%m-%d
```

The following format options may be used to control the format of the timestamp:



%a Abbreviated weekday name	%S Second as decimal number (00 - 59)
%A Full weekday name	%U Week of year as decimal number, with Sunday as first day of week (00 - 53)
%b Abbreviated month name	%w Weekday as decimal number (0 - 6, where Sunday is 0)
%B Full month name	%W Week of year as decimal number, with Monday as first day of week (00 - 53)
%c Date and time representation appropriate for location	%x Date representation for current locale
%d Day of month as decimal number (01 - 31)	%X Time representation for current locale
%H Hour in 24-hour format (00 - 23)	%y Year without century, as decimal number (00 - 99)
%I Hour in 12-hour format (01 - 12)	%Y Year with century, as decimal number
%j Day of year as decimal number (001 - 366)	%z Either the time-zone name or time-zone abbreviation, depending on registry settings; no characters if time-zone is unknown
%m Month as decimal number (01 - 12)	%Z Percent sign
%M Minute as decimal number (00 - 59)	
%p Current location AM/PM indicator for 12-hour clock	

The # flag can prefix any formatting code. In this case, the meaning of the format code is changed by that # as follows:

%#c Long date and time representation appropriate for current location. For example:

'Tuesday, March 14, 1995, 12:41:29'.

%#x Long date representation appropriate to current location. For example:

'Tuesday, March 14, 1995'.

%#d, %#H, %#I, %#j, %#m, %#M, %#S, %#U, %#w, %#W, %#y, %#Y Remove leading zeros (if any).

is ignored for all other flags.



4.4 Run scripts and open URLs

On the **Action tab**:

- The 'Run command' allows you to use command line interfaces or scripts that are dependent on the card ID which has been read. There is a **Run command example**.
- The 'Launch URL' option lets you create a similar action for your website. There is a **Launch URL when a card is presented example**.

Both of these actions allow multiple parameters to be used, such as the reader serial number (if available), card type, or time stamp, as well as the card ID.

4.5 Keyboard buffer templates

On the **Format tab** you can create an advanced template for the keyboard buffer to use. This greatly increases the range of the **Default - Keyboard wedge functionality** by allowing you to insert special characters or keyboard commands, to direct the information you want to where it is needed. The **Insert RFID card ID into Word or Excel example** uses this facility.

4.6 Enable web server

On the **Web Server tab** you can enable a web server from which your web site or application can receive card data obtained through Read-a-Card. There is more about this in the [Developer Guide](#).

4.7 Lock settings

On the **Lock tab** you can lock the Read-a-Card user interface to prevent end-users from accidentally modifying the settings after an application has been set up.

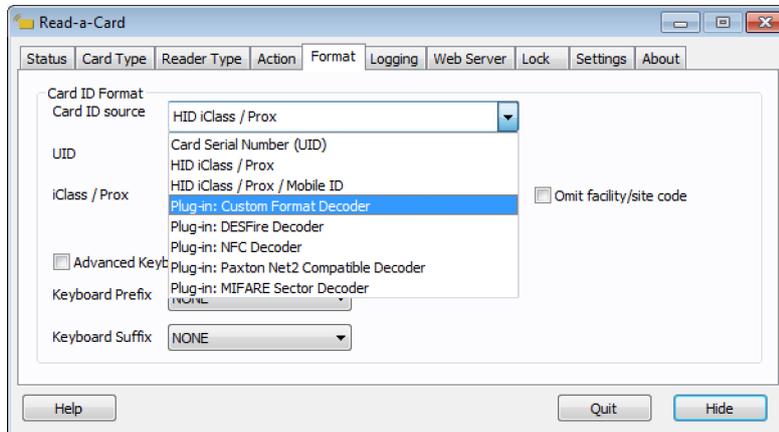
4.8 Logging structures

On the **Logging tab** you can easily create folder structures and logs with names based on information gathered by Read-a-Card, such as which reader detected the card or the current date. The **Log contactless card information example** demonstrates this facility. The Read-a-Card logs that are created can be used by your software and read back in real time.



4.9 Using Read-a-Card plug-ins

Read-a-Card will display all available plug-ins in the Card ID source dropdown menu on the **Format** tab. All plug-ins have the 'Plug-in:' prefix in that menu.



Some plug-ins only require that Read-a-Card is licensed for the plug-in to function; others require specific configuration data. If a plug-in appears in the dropdown list but is 'greyed out', it is unable to work for one of these reasons.

Note: Only one plug-in can be in use at any time.

Clicking on the  next to the format dropdown box will pop-up an information window showing the currently selected plug-in format name, version, and other features. This window also contains a link to edit the `.ini` configuration file for the plug-in and a link to help in this guide.

Read-a-Card currently includes:

- **NFC Decoder** for decoding certain types of text NDEF records stored in NFC tags;
- **Paxton Net2 Compatible Decoder** for converting card IDs to the format used by Paxton systems;
- **DESFire Decoder** for reading secure application DESFire files;
- **MIFARE Sector Decoder** for decoding data stored in MIFARE Classic sectors, including those protected by specific access keys; and a
- **Custom Format Decoder** to extract specific data in particular formats.

If you require something else, or have any other requirements for decoding and returning contactless card ID information, please contact us via support@read-a-card.com to discuss how we can help.



A NFC Decoder

The Read-a-Card NFC Decoder plug-in turns Read-a-Card into an NFC launcher application. This plug-in can decode Smart Poster NDEF records from any of the standard NFC tag types, and take appropriate action.

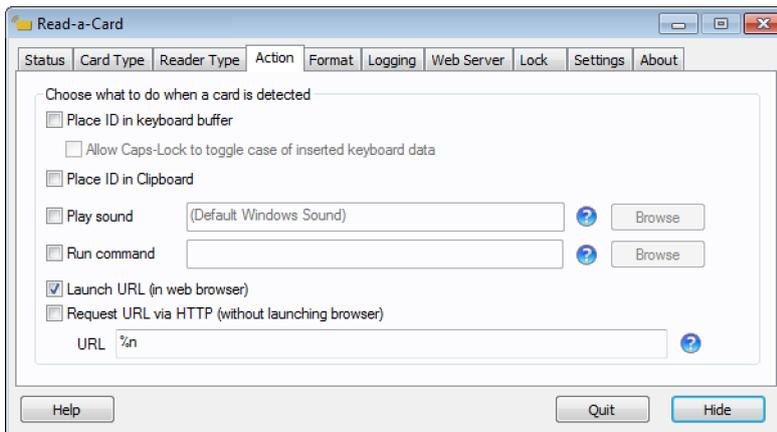
The plug-in will extract the Smart Poster URI and text information, if present, to use with Read-a-Card actions. For example, Read-a-Card can be configured to launch the Smart Poster URI page in a web browser, or to insert the Smart Poster URI and/or text fields into the PC's keyboard buffer.

A.1 Configuration of the NFC plug-in

No configuration is required beyond selecting the NFC plug-in from the Card ID source dropdown menu on the **Format tab**.

A.2 Set up required actions

Select the **Action tab** to choose what you want to happen when a Smart Poster tag is decoded by your reader.



If you want to launch the Smart Poster URI page in a web browser follow the **NFC Smart Poster example** in this guide.

Read-a-Card actions can use any combination of the URI and text fields retrieved from the tag. When using the NFC Decoder %n represents the URI and %c any content from the Smart Poster text field. These fields can be used in a Launch URL or Run command on this tab, or as part of the Advanced Keyboard Format template on the **Format tab**.

Read-a-Card's logging feature can record the Smart Poster URI obtained from each tag presented.

If you have any other requirements or questions about using NFC technology with Read-a-Card please contact us at support@read-a-card.com.



B Paxton Net2 Compatible Decoder

The Read-a-Card Paxton Net2 Decoder plug-in sets up Read-a-Card to convert the UID of various cards, such as MIFARE Classic, to the specific format used by Paxton Net2 access systems. This involves stripping certain bits and converting the results to decimal.

No configuration is required beyond selecting the Paxton Net2 Decoder plug-in from the Card ID source dropdown menu on the **Format tab**.



C DESFire Decoder

By default, Read-a-Card will read the CSN/UID of a contactless card. If you want to read secure application data from a specific file on a DESFire card, you will need the DESFire Decoder plug-in. This allows Read-a-Card to return that data, formatted according to your requirements. It will also allow you to optionally store access keys securely in hardware, using a SAM (secure access module).

Read-a-Card supports reading and decoding of IDs stored in a nominated secure file on DESFire EV1 and EV2 cards of any size (2K, 4K or 8K). It can work in conjunction with Dot Origin's Key-ID Encoder software, or with other solutions for encoding and managing DESFire cards in a multi-application environment.

C.1 Configuration of DESFire Decoder plug-in

Read-a-Card plug-ins may be configured using a Windows configuration file. In the case of the DESFire Decoder, this file is named `DESFireDecoder.ini`.

This `.ini` file should be with the `Read-a-Card.ini` file. You will find this:

On Windows XP/2000: `C:\Documents and Settings\All Users\Application Data\Read-a-Card\`

On Windows Vista/7/8/10: `C:\ProgramData\Read-a-Card\`

For the DESFire Sector decoder plug-in to work, `DESFireDecoder.ini` must at least contain the following:

```
[Config]
PluginVersion=1.1
PluginName=DESFire Decoder
```

Note: A configuration file with the minimum essential content is generated when you choose this plug-in on the **Format tab**. Take care not to delete these lines, as you add other settings.

`PluginVersion` specifies the minimum version of the plug-in that this configuration expects. If this value is greater than the actual plug-in version, an error message will be displayed when the plug-in is selected.



C.2 Modes of operation

Read-a-Card can operate in three modes, offering different options for how it obtains the necessary key and configuration information for decoding a DESFire card when presented to a reader:

- Using **DESFire Decoder with Key-ID hardware template mode**, stored in a Key-ID SAM;
- Using **DESFire Decoder with Key-ID software template mode**, stored in a Key-ID .tid file;
- **DESFire Decoder configuration in the plug-in .ini file only.**

The first two options are designed to work in conjunction with Dot Origin's Key-ID Encoder software, while the third provides flexibility to configure Read-a-Card to read from cards issued and used by other software.

C.2.1 DESFire Decoder with Key-ID hardware template mode

In this mode, no further configuration is necessary beyond selecting the DESFire plug-in from the Card ID source dropdown menu on the **Format tab**.

Simply attach a suitable Key-ID SAM to the PC, usually by inserting it into a small USB SIM reader.

You will be asked to enter the user passphrase the first time a DESFire card is presented to the reader. After that the plug-in will automatically decode and return the stored ID as required.

C.2.2 DESFire Decoder with Key-ID software template mode

In this mode, select the DESFire plug-in from the Card ID source dropdown menu on the **Format tab**.

Save the software template on your PC and add the following entries to the [Config] section of the .ini file to point to that template:

```
ReadUsing=Template
TemplateFile=<location of Key-ID software template .tid file>
```

Note: If you add extra settings do not repeat the [Config] section heading. There must only be one [Config] section in a .ini file.

C.2.3 DESFire Decoder configuration in the plug-in .ini file only

If a hardware or software template is not being used, you can specify all of the required details in the DESFireDecoder.ini file itself.

Documentation on this option is available on request.



D MIFARE Sector Decoder

By default, Read-a-Card will read the CSN/UID of a contactless MIFARE Classic card. If instead you wish to read a specific sector of a MIFARE Classic card, you will need the MIFARE Sector Decoder plug-in to ensure Read-a-Card can return that data, formatted according to your requirements.

The MIFARE Sector Decoder plug-in supports reading and decoding IDs stored in sectors on MIFARE 1K, 4K, Ultralight, Ultralight C and NTAG203 card types.

The following sections describe:

- **Configuration of MIFARE Sector Decoder plug-in** on page D-1
- **Sector data configuration per card type** on page D-2
- **MIFARE access keys** on page D-4
- **Decoding facility codes and card numbers** on page D-5
- **Read-a-Card actions** on page D-8

D.1 Configuration of MIFARE Sector Decoder plug-in

Read-a-Card plug-ins may be configured using a Windows configuration file. In the case of the MIFARE Sector Decoder plug-in, this file is named `RACFormatDecodeA.ini`.

This `.ini` file is in the folder with the `Read-a-Card.ini` file. You will find this:

On Windows XP/2000: `C:\Documents and Settings\All Users\Application Data\Read-a-Card\`

On Windows Vista/7/8/10: `C:\ProgramData\Read-a-Card\`

For the MIFARE Sector decoder plug-in to work, `RACFormatDecodeA.ini` must contain at least:

```
[Config]
ReadFromIniFile=1
PluginVersion=1.2
PluginName=MIFARE Sector Decoder
```

Note: A configuration file with the minimum essential content is generated when you choose this plug-in on the **Format tab**. Take care not to delete these lines, as you add other settings.

`PluginVersion` specifies the minimum version of the plug-in that this configuration expects. If this value is greater than the actual plug-in version, an error message will be displayed when the plug-in is selected.

There is a full **MIFARE Sector Decoder example configuration file** at the end of this section.



D.2 Sector data configuration per card type

For each of the main card types, a unique section name is used to configure how data is read from that card type. Different card types can have different data locations, keys and format conversion (as shown in the full **MIFARE Sector Decoder example configuration file** at the end of this section.)

The following sections are recognised:

[MIFARE_1K]

[MIFARE_4K]

[MIFARE_UL] (which includes MIFARE Ultralight, Ultralight C, NTAG203 and similar card types)

Note: Not all readers support reading data from Ultralight card types beyond the first 64 bytes.

As an example, with a MIFARE 1K card the following options might be used:

```
[MIFARE_1K]
ReadData=1      ;1 means do read sector data from this card type
StartSector=0  ;Integer. The sector from which to start reading.
StartBlock=1   ;Integer. The block from which to start reading.
StartOffset=0  ;Integer. The byte offset within the block from
                ;which to start reading.
BytesToRead=8  ;Integer. Number of bytes to read.
OutputFormat=0;Integer.How to format data (see OutputFormat
                ;section for more detail).
KeyType=1      ;Integer. 0=ASCII 1=Binary data as ASCII hex pairs
KeyA=1A2B3C4D5E6F ;Optional KeyA
KeyB=FFFFFFFFFFFFFF ;Optional KeyB
```

D.2.1 ReadData

To read a configured file or block the ReadData option must be set to 1. If this option is not set for a particular card type, the card's serial number (UID) will be returned as the default.

D.2.2 StartSector and StartBlock

The memory in a MIFARE 1K or 4K card is organised in numbered sectors. Each sector contains 4 or 16 blocks (depending on card type and sector number) and every block contains 16 bytes of data.

Any combination of sector and block numbering can be used. Compare the examples:

<pre>StartSector=3 StartBlock=1</pre>	<pre>StartSector= StartBlock=13</pre>
Will start reading from sector 3 in block 1. This configuration will read block number 13.	Has the same effect. This configuration will read block number 13.

D.2.3 StartOffset

A block of memory in a MIFARE 1K or 4K card contains 16 bytes of data (numbered 0 to 15) while the memory of MIFARE Ultralight card types is arranged in blocks of 4 bytes. You can configure the



plug-in to start reading from a certain byte number within a block using the `StartOffset`.

D.2.4 BytesToRead

Reading data that runs over multiple sectors and/or blocks will be handled automatically. For example, if `BytesToRead` is greater than 16.

D.2.5 OutputFormat

The `OutputFormat` value determines how the resulting bytes will be formatted. It offers an ASCII option in addition to the options available from the UID dropdown menu on the **Format tab**. It can have the following values:

0. Decimal reversed (default)

This option takes the last 4 bytes of data (or all bytes if fewer than 4 have been read) and interprets them as a little-endian binary 32 bit number (that is, the last byte in the sequence has the most significant value). The resulting 32 bit value is formatted as a 10 digit decimal number.

1. Decimal

This option takes the first 4 bytes of data (or all bytes if fewer than 4 have been read) and interprets them as a big-endian binary 32 bit number (that is, the first byte in the sequence has the most significant value). The resulting 32 bit value is formatted as a 10 digit decimal number.

2. Hex reversed

This option outputs all bytes of the data in hexadecimal (two ASCII numeric or uppercase hex characters per byte) in reverse order.

3. Hex standard

This option outputs all bytes of the data in hexadecimal (two ASCII numeric or uppercase hex characters per byte) in the same order that they were read.

4. Decimal 64 bit

This option takes the first 8 bytes of data (or all bytes if fewer than 8 have been read) and interprets them as a little-endian binary 64 bit number (that is, the last byte in the sequence has the most significant value). The resulting 64 bit value is formatted as a decimal number with no leading zeros.

5. Decimal 64 bit reversed

This option takes the last 8 bytes of data (or all bytes if fewer than 8 have been read) and interprets them as a big-endian binary 64 bit number (that is, the first byte in the sequence has the most significant value). The resulting 64 bit value is formatted as a decimal number with no leading zeros.



6. ASCII

This option outputs all bytes of data as ASCII printable characters. Any byte values that do not represent ASCII printable characters will be output as a '.' character.

D.3 MIFARE access keys

With MIFARE 1K and 4K cards, access to a sector's data may require one or two authentication keys. These keys (KeyA and KeyB) can be configured for each card type. The keys can be expressed as either an ASCII character string or as binary value hex pairs indicated by the `KeyType` setting:

`KeyType=1` indicates an ASCII String , for example 'my key'

`KeyType=0` indicates binary data as ASCII hex pairs, for example '6d79206b6579'

For example:

```
KeyType=0  
KeyA=my key
```

or

```
KeyType=1  
KeyA=6d79206b6579
```

If they are required, the `KeyType`, `KeyA` and `KeyB` options should be specified in the appropriate card type section (allowing different keys to be used with different card types).

D.3.1 Hardware option for access keys

It may not be appropriate to expose the access keys for your cards in a plain text configuration file. Dot Origin can supply an encrypted, secured copy of your configuration file on a smart card security module (SAM). When configured and licensed in this way, the MIFARE Sector Decoder plug-in will read all of its configuration data, including the access keys, from the SAM in a secure manner.

Please contact [Dot Origin](#) or your supplier for information on how to obtain a SAM with your configuration.



D.4 Decoding facility codes and card numbers

If you need to extract a facility code and card number from the bytes read from the card's memory, the MIFARE Sector Decoder plug-in can use **BitMask**, **DigitMask** or **Fixed facility codes**. The resulting ID (facility code followed by card number) will be seen on the **Status tab** and can be used where required, as %n in Advanced Keyboard Format, URL, HTTP or run command actions.

If they are required, the `BitMask`, `DigitMask` and `FAC` options should be specified in the appropriate card type section (allowing different options for each card type).

D.4.1 DigitMask

A `DigitMask` value indicates how the digits that result from the `OutputFormat` process should be distributed between the facility code and card ID.

For example:

```
DigitMask=xxfff-ccccx
```

Only `f` (facility code) and `c` (card number) characters in the `DigitMask` have any meaning. Any other characters indicate that the corresponding digit is to be ignored.

So, with the above `DigitMask` example, if the result of reading and formatting the data is the value 123456789012, the mask will match the data like this:

```
123456789012
```

```
xxfff-ccccx
```

So:

Facility code = 345

Card number = 78901

Note: By default any mask is left aligned with the data. Setting `MaskAlignment=1` would align the mask with the right hand edge of the card data. Setting `MaskAlignment=0` returns the default behaviour.

If you need the extracted facility code and/or card number to be padded with leading zeros, use:

```
FACDigits=4
```

```
CardNumDigits=7
```

Using these extra settings with the above example, would yield a facility code of 0345 and a card number of 0078901. The resulting ID (combined facility code and card number) would be 03450078901.



D.4.2 BitMask

BitMask allows more complex bitwise facility code and card number decoding.

Note: The BitMask and DigitMask options cannot be used together for a given card type. (If both are accidentally specified then only the DigitMask option will be used).

```
BitMask=-----pffffffffffffffffccccccccccccccccccccp-----  
ByteOrder=0  
BitOrder=0
```

The BitMask value is applied bitwise to the binary data read from the card. Only f (facility code) and c(card number) characters in the BitMask have any meaning. Any other characters indicate that the corresponding bit is to be ignored.

Without an OutputFormat option, the bytes read from the card are first converted into a 64 bit integer value.

The ByteOrder option determines whether the first or last byte should be considered the most significant:

ByteOrder=0 means the first byte is the most significant.

ByteOrder=1 means the first byte is the least significant.

The BitOrder value may then be used to reverse the order of the bits in the resulting facility code and card number:

BitOrder=0 means that the leftmost character of the BitMask is matched to the most significant bit of the data.

BitOrder=1 means that the rightmost character of the BitMask is applied to the least significant bit of the data and that the bit order is reversed. (That is, the least significant bit is treated as the most significant in the resulting facility code and card number).

Note: Take care if setting both BitOrder and MaskAlignment at the same time, as both parameters reverse the alignment of the mask with the data. If BitOrder=1 and MaskAlignment=1 the alignment will be reversed twice, leaving the data and mask with the default left alignment.

So, for example, reading 6 bytes of data:

```
BytesToRead=6
```

The data read from the card is: C6 8B 06 92 48 14

```
ByteOrder=1
```

Reverses the byte order giving a 64 bit hex value of: 144892068BC6

In binary this is:

```
0001 0100 0100 1000 1001 0010 0000 0110 1000 1011 1100 0110
```

Matching this binary value to a BitMask and BitOrder then works as follows:



Example 1 - BitOrder=0

Using:

```
BitMask=-----pffffffffffffffffccccccccccccccccccp  
BitOrder=0
```

The mask matches the data like this:

```
000101000100100010010010000001101000101111000110  
-----pffffffffffffffffccccccccccccccccccp
```

So:

Facility code = 00010010001001b = 489h = 1161 (decimal)

Card number = 0010000001101000101b = 10345h = 66373 (decimal)

The following options allow the facility code and/or card number to be formatted with leading zeros:

```
FACDigits=4  
CardNumDigits=6
```

So using the above values the resulting facility code would be 0489, the card number would be 066373. The overall combined ID would be 0489066373.

Example 2 - BitOrder=1

As another example, consider the following BitMask with BitOrder=1 used on the same data:

```
BitMask=ccccccccccccccccccccffffffffffffffffffp  
BitOrder=1
```

In this case we would get the following match:

```
000101000100100010010010000001101000101111000110  
                pccccccccccccccccccccffffffffffffffffffp
```

The BitMask is right-aligned with the data due to BitOrder=1. This option also reverses the bit order giving:

Facility code = 11000111101000b = 31E8h = 12776 (decimal)

Card number = 1011000000100100100b = 58124h = 360740 (decimal)

In this case the FACDigits=4 and CardNumDigits=6 options will not cause any additional leading zeros to be added because the values are already contain at least this number of digits. The resulting combined ID will be 12776360740.

D.4.3 Fixed facility codes

A fixed facility code for all cards of a particular type can be configured using the setting:

```
FAC=123; (String. The fixed facility code.)
```



D.5 Read-a-Card actions

You can control which Read-a-Card actions are taken (depending on the type of card) using the `ActionMask` setting.

If required, the `ActionMask` option should be specified in the appropriate card type section. This allows the plug-in configuration for a particular card type to suppress actions that may otherwise be enabled in Read-a-Card's generic configuration.

```
ActionMask=00
```

The `ActionMask` value is a hex pair, for example '7E' for Action Mask 0111 1110

The following is the bitwise representation of the Action Mask:

```
X X X X   X X X X  
7 6 5 4   3 2 1 0
```

Bit	Use	Enable	Disable
7	Not Used	N/A	N/A
6	Log Action	0	1
5	URL/HTTP Action	0	1
4	Command Action	0	1
3	Sound Action	0	1
2	Clipboard Action	0	1
1	Keyboard Action	0	1
0	Not used	N/A	N/A



D.6 MIFARE Sector Decoder example configuration file

Here is example content for a complete `RACFormatDecodeA.ini` configuration file:

```
[Config]
ReadFromIniFile=1
PluginVersion=1.2
PluginName=MIFARE Sector Decoder
FormatUIName=ACME Ltd Card Format
```

```
[MIFARE_1K]
ReadData=1
StartSector=0
StartBlock=1
StartOffset=4
OutputFormat=3
BytesToRead=8
FormatName=MIFARE 1K (sector decoder)
ByteOrder=1
BitOrder=0
DigitMask=fffccccc
FACDigits=3
CardNumDigits=5
```

```
[MIFARE_4K]
ReadData=1
StartSector=0
StartBlock=1
BytesToRead=16
FormatName=MIFARE 4K (sector decoder)
ByteOrder=1
BitOrder=0
BitMask=-----ffffffffffffffffccccccccccccccccccccp-----
FACDigits=4
CardNumDigits=6
KeyType=1
KeyA=123456789abc
KeyB=FFFFFFFFFFFF
```

```
[MIFARE_UL]
ReadData=1
StartBlock=4
BytesToRead=4
FormatName=MIFARE Ultralight (sector decoder)
```



E Custom Format Decoder

By default, Read-a-Card will read the CSN/UID of a contactless card and format it in one of six ways. If you want to format the CSN/UID differently, for example by reading only part of the UID, you will need the Custom Format Decoder plug-in to ensure Read-a-Card can return that data, formatted according to your requirements.

The Custom Format Decoder supports reading and decoding UIDs from all card types supported by Read-a-Card. It can also be used to decode the credentials read from HID Prox or iClass cards, if a suitable HID/OMNIKEY reader is used.

E.1 Configuration of Custom Format Decoder plug-in

Read-a-Card plug-ins may be configured using a Windows configuration file. In the case of the Custom Format Decoder, this file is named `CustomFormatDecoder.ini`.

This `.ini` file should be with the `Read-a-Card.ini` file. You will find this:

On Windows XP/2000: `C:\Documents and Settings\All Users\Application Data\Read-a-Card\`

On Windows Vista/7/8/10: `C:\ProgramData\Read-a-Card\`

For the Custom Format decoder plug-in to work, `CustomFormatDecoder.ini` must at least contain the following:

```
[Config]
PluginVersion=1.0
PluginName=Custom Format Decoder
```

Note: A configuration file with the minimum essential content is generated when you choose this plug-in on the **Format tab**. Take care not to delete these lines, as you add other settings.

`PluginVersion` specifies the minimum version of the plug-in that this configuration expects. If this value is greater than the actual plug-in version, an error message will be displayed when the plug-in is selected.

E.1.1 Rename a plug-in

The Read-a-Card format name displayed in the 'Card ID source' dropdown box can optionally be configured by including this parameter in the `[Config]` section of any plug-in `.ini` file:

```
FormatUIName=My name for this plug-in
```

Note: If you add extra settings do not repeat the `[Config]` section heading. There must only be one `[Config]` section in a `.ini` file.



E.2 Configuration for different card types

In addition to the `[Config]` section, the following sections are recognised in this plug-in configuration file:

```
[Prox]
[iClass]
[UID]
```

The `[Prox]` and `[iClass]` sections allow specific configuration settings to be specified for each card type. The `[UID]` section specifies how the UID should be formatted for all other card types. If either the `[Prox]` or `[iClass]` section is missing and a Prox or iClass card is presented then the settings in the `[UID]` section will be used to format the Prox or iClass credential data.

Note: Only HID OMNIKEY iClass readers are capable of reading the credential from an iClass card physical access application area. Other readers may be able to read the iClass Card Serial Number (CSN), effectively the UID of an iClass card.

E.2.1 Decode UIDs and HID credentials

This plug-in supports two methods of extracting and formatting the raw data read from the card. These are **DigitMask** and **BitMask**, shared with the MIFARE Sector Decoder plug-in. These options are applicable to both UIDs (from any card type) and HID credentials (either Prox or iClass).

`DigitMask` is used in conjunction with the **OutputFormat** setting. It may be used to extract and arrange whole digits from the card number after it has been interpreted, according to the selected `OutputFormat`.

`BitMask` operates on individual bits within the raw data of the UID or HID data. It can be used to discard start, stop and parity bits and to select which bits should be assigned to the facility code and which bits assigned to the card number. When using the `BitMask` option, the resulting card number and optional facility code are always returned as decimal values.

Note: The `BitMask` and `DigitMask` options cannot be used together for a given card type. (If both are accidentally specified then only the `DigitMask` option will be used).

`FACDigits` and `CardNumDigits` are options which may be used with either the `DigitMask` or `BitMask` options. They produce a consistent number of digits for the facility code or card number by adding leading zeros as required.

You control the Custom Format Decoder plug-in in exactly the same way as for the MIFARE Sector Decoder plug-in. (For details, see earlier sections on **DigitMask**, **BitMask**, **OutputFormat** and **Read-a-Card actions**).

