CEN

WORKSHOP

AGREEMENT

CWA 16008-10

August 2009

ICS 35.240.40

English version

J/eXtensions for Financial Services (J/XFS) for the Java Platform - Release 2009 - Part 10: Check Reader/Scanner Device Class Interface - Programmer's Reference

This CEN Workshop Agreement has been drafted and approved by a Workshop of representatives of interested parties, the constitution of which is indicated in the foreword of this Workshop Agreement.

The formal process followed by the Workshop in the development of this Workshop Agreement has been endorsed by the National Members of CEN but neither the National Members of CEN nor the CEN Management Centre can be held accountable for the technical content of this CEN Workshop Agreement or possible conflicts with standards or legislation.

This CEN Workshop Agreement can in no way be held as being an official standard developed by CEN and its Members.

This CEN Workshop Agreement is publicly available as a reference document from the CEN Members National Standard Bodies.

CEN members are the national standards bodies of Austria, Belgium, Bulgaria, Cyprus, Czech Republic, Denmark, Estonia, Finland, France, Germany, Greece, Hungary, Iceland, Ireland, Italy, Latvia, Lithuania, Luxembourg, Malta, Netherlands, Norway, Poland, Portugal, Romania, Slovakia, Slovenia, Spain, Sweden, Switzerland and United Kingdom.

EUROPEAN COMMITTEE FOR STANDARDIZATION
COMITÉ EUROPÉEN DE NORMALISATION
EUROPÄISCHES KOMITEE FÜR NORMUNG

Management Centre: Avenue Marnix 17, B-1000 Brussels

Ref. No.:CWA 16008-10:2009 E

Contents

Foreword

This CWA contains the specifications that define the J/eXtensions for Financial Services (J/XFS) for the Java ™ Platform, as developed by the J/XFS Forum and endorsed by the CEN J/XFS Workshop. J/XFS provides an API for Java applications which need to access financial devices. It is hardware independent and, by using 100% pure Java, also operating system independent.

The CEN J/XFS Workshop gathers suppliers (among others the J/XFS Forum members), service providers as well as banks and other financial service companies. A list of companies participating in this Workshop and in support of this CWA is available from the CEN Secretariat, and at http://www.cen.eu/cenorm/sectors/sectors/isss/activity/jxfs_membership.asp. The specification was agreed upon by the J/XFS Workshop Meeting of 2009-05-6/9 in Brussels, and the final version was sent to CEN for publication on 2009-06-12.

The specification is continuously reviewed and commented in the CEN J/XFS Workshop. The information published in this CWA is furnished for informational purposes only. CEN makes no warranty expressed or implied, with respect to this document. Updates of the specification will be available from the CEN J/XFS Workshop public web pages pending their integration in a new version of the CWA (see http://www.cen.eu/cenorm/sectors/sectors/isss/activity/jxfs_cwas.asp).

The J/XFS specifications are now further developed in the CEN J/XFS Workshop. CEN Workshops are open to all interested parties offering to contribute. Parties interested in participating and parties wanting to submit questions and comments for the J/XFS specifications, please contact the J/XFS Workshop Secretariat hosted in CEN (jxfs-helpdesk@cen.eu).

Questions and comments can also be submitted to the members of the J/XFS Forum through the J/XFS Forum web-site http://www.jxfs.net.

This CWA is composed of the following parts:

- Part 1: J/eXtensions for Financial Services (J/XFS) for the Java Platform – Release 2009 - Base Architecture - Programmer's Reference
- Part 2: J/eXtensions for Financial Services (J/XFS) for the Java Platform - Release 2009 - Pin Keypad Device Class Interface - Programmer's Reference
- Part 3: J/eXtensions for Financial Services (J/XFS) for the Java Platform - Release 2009 - Magnetic Stripe & Chip Card Device Class Interface - Programmer's Reference
- Part 4: J/eXtensions for Financial Services (J/XFS) for the Java Platform - Release 2009 - Text Input/Output Device Class Interface - Programmer's Reference
- Part 5: J/eXtensions for Financial Services (J/XFS) for the Java Platform - Release 2009 - Cash Dispenser, Recycler and ATM Device Class Interface - Programmer's Reference
- Part 6: J/eXtensions for Financial Services (J/XFS) for the Java Platform - Release 2009 - Printer Device Class Interface - Programmer's Reference
- Part 7: J/eXtensions for Financial Services (J/XFS) for the Java Platform - Release 2009 - Alarm Device Class Interface - Programmer's Reference
- Part 8: J/eXtensions for Financial Services (J/XFS) for the Java Platform - Release 2009 - Sensors and Indicators Unit Device Class Interface - Programmer's Reference
- Part 9: J/eXtensions for Financial Services (J/XFS) for the Java Platform - Release 2009 - Depository Device Class Interface - Programmer's Reference
- Part 10: J/eXtensions for Financial Services (J/XFS) for the Java Platform - Release 2009 - Check Reader/Scanner Device Class Interface - Programmer's Reference (deprecated in favour of Part 13)
- Part 11: J/eXtensions for Financial Services (J/XFS) for the Java Platform - Release 2009 - Camera Device Class Interface - Programmer's Reference
- Part 12: J/eXtensions for Financial Services (J/XFS) for the Java Platform - Release 2009 - Vendor Dependant Mode Specification - Programmer's Reference
- Part 13: J/eXtensions for Financial Services (J/XFS) for the Java Platform – Scanner Device Class Interface - Programmer's Reference (recommended replacement for Part 10)

Note: Java and all Java-based trademarks and logos are trademarks of Sun Microsystems, Inc. The Java Trademark Guidelines are currently available on the web at http://www.sun.com All other trademarks are trademarks of their respective owners.

This CEN Workshop Agreement is publicly available as a reference document from the National Members of CEN : AENOR, AFNOR, ASRO, BDS, BSI, CSNI, CYS, DIN, DS, ELOT, EVS, IBN, IPQ, IST, LVS, LST, MSA, MSZT, NEN, NSAI, ON, PKN, SEE, SIS, SIST, SFS, SN, SNV, SUTN and UNI.

Comments or suggestions from the users of the CEN Workshop Agreement are welcome and should be addressed to the CEN Management Centre.

History

Main Differences to CWA 14394-10:2004 are:

- open job handling clarified at base architecture level so specific chapter in this document is removed.
- specific declaration of result codes used by each job has been removed, and now result refers to common section at the end of the document.
- New JxfsCHKStatusSelectorEnum enumeration introduced to allow use of new getStatus method defined in base architecture documentation.

1 Scope

This document describes the Check Reader/Scanner class based on the basic architecture of J/XFS which is similar to the JavaPOS architecture. It is event driven and asynchronous.

This specification has been superseded by the new part 13: Scanner Class Interface and it is now deprecated. It is strongly suggested to use the new device class interface for new implementations.

Three basic levels are defined in JavaPOS. For J/XFS this model is extended by a communication layer, which provides device communication that allows distribution of applications and devices within a network. So we have the following layers in J/XFS:

- Application
- Device Control and Device Manager
- Device Communication
- Device Service

Application developers program against control objects and the Device Manager which reside in the Device Control layer. This is the usual interface between applications and J/XFS devices. Device Control objects access the Device Manager to find an associated Device Service. Device Service objects provide the functionality to access the real device (i.e. like a device driver).
During application startup the Device Manager is responsible for locating the desired Device Service object and attaching this to the requesting Device Control object. Location and/or routing information for the Device Manager reside in a central repository.

To support Check Reader/Scanner devices the basic Device Control structure is extended with various properties and methods specific to this device which are described on the following pages.

2 Overview

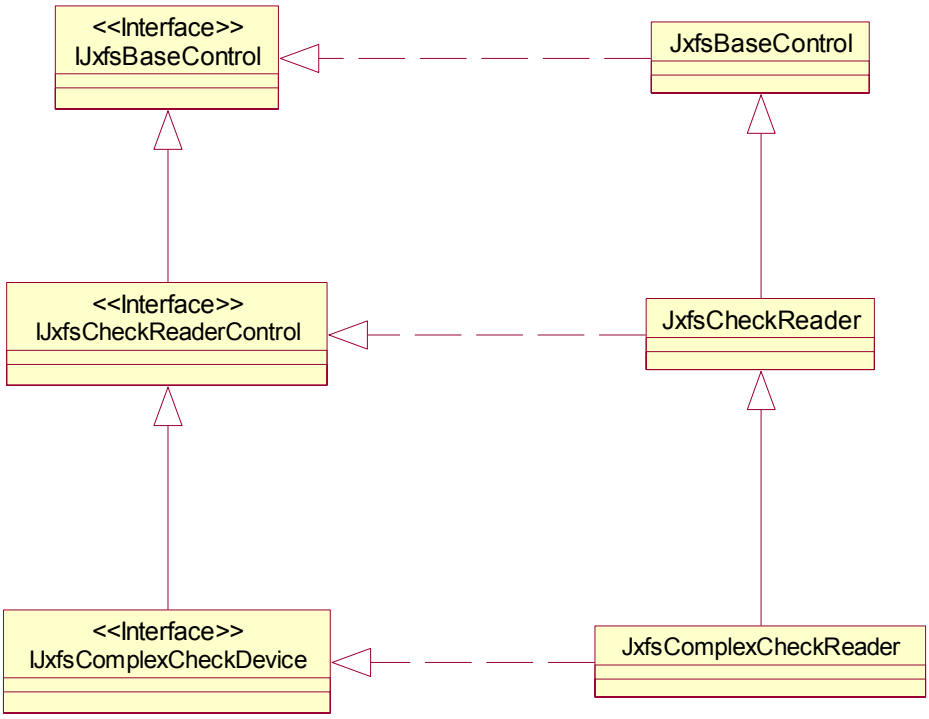
2.1 Description

The J/XFS Check Reader/Scanner Device Support allows for the operation of devices with a range of features, from small hand-held read-only devices where checks are manually swiped through one at a time, to much larger devices which automatically feed checks by the batch past a reader, an encoder, an endorser, an optional image scanner, to be sorted into one of several pockets.

In the U.S. checks are always encoded in magnetic ink for reading by Magnetic Ink Recognition (MICR), and a single font is always used. In other areas some countries use MICR and some use Optical Character Recognition (OCR) character sets, with different fonts.

As well as the rest of J/XFS device controls, J/XFS Check Reader/Scanner devices use the event driven model and the same behavioral model. Therefore, the application will instantiate a J/XFS Check Reader/Scanner Device Control Object and then use the available methods to do I/O. When an I/O method is called, the J/XFS Check Reader/Scanner Device Service will attempt to process the requested I/O. If the request is invalid or an exception is encountered, the application will be notified by a J/XFS exception. Completion of the request will be reported by an event. Thus the application must register itself with the J/XFS Check Reader/Scanner Device Control Object for the various types of events it wishes to handle.

2.2 Class Hierarchy



2.3 Classes and Interfaces

The following classes and interfaces are used by the J/XFS CheckReader Device Controls. In order to support the definition of the different properties of the different devices (see Introduction), the Device Controls are defined in a class hierarchy.

Class or Interface	Name	Description	Extends or Implements
Interface	**IJxfsBaseControl**	Base interface for all the device controls. Contains methods common to all the device controls.	
Interface	**IJxfsCheckReaderControl**	Base interface for CheckReader controls. Contains method declarations specific to CheckReader controls.	Extends: **IJxfsBaseControl**
Interface	**IJxfsCheckReaderService**	Base interface for CheckReader services. Contains the methods specific to the device services for the CheckReader device category.	Extends: IJxfsBaseService
Interface	**IJxfsComplexCheckDevice**	Interface for complex check devices. Contains method declarations specific to complex check devices.	Extends: **IJxfsCheckReaderControl**
Interface	**IJxfsComplexCheckReaderService**	Interface for complex CheckReader services. Contains the methods specific to the device services for the complex check devices.	Extends: **IJxfsCheckReaderService**
Class	**JxfsBaseControl**	Base class for all the device controls. Contains properties common to all the device controls.	
Class	**JxfsCheckReader**	Base class for CheckReader controls. Contains properties specific to CheckReader device controls.	Extends: **JxfsBaseControl** Implements: **IJxfsCheckReaderControl**
Class	**JxfsComplexCheckReader**	Base class for check reader controls supporting the IJxfsComplexCheckDevice interface	Extends: **JxfsCheckReader** Implements: **IJxfsComplexCheckDevice**

2.4 Support Classes

Class or Inter-face	Name	Description	Extends / Implements
Interface	**JxfsConst**	Interface containing the Jxfs constants that are common to several device categories	--
Interface	**JxfsCHKConst**	Interface containing the Jxfs constants that are common to all the CheckReader device controls.	--
Class	**JxfsCHKData**	Data class that contains data returned in Operation Complete events for CheckReader *readData()* operation.	Extends: **JxfsType**
Class	**JxfsCHKProcessData**	Data class that contains data required to perform check processing.	Extends: **JxfsType**
Class	*JxfsStatusEvent* *JxfsOperationCompleteEvent* *JxfsIntermediateEvent*	The Device Service creates instances of these classes and delivers them through the J/XFS CheckReader Device Control's event callbacks to the application	Extends: **JxfsEvent**
Class	**JxfsException**	Exception class. The J/XFS CheckReader Device Control creates and throws exceptions on method failure and property access failure.	Extends: **java.lang.Exception**

3 Device behavior

3.1 Handling of null parameters

If null is passed as a method parameter, a JxfsException exception with the errorCode property set to JXFS_E_PARAMETER_INVALID will be thrown, unless the handling of a null parameter is explicitly specified for a particular method.

4 Classes and Interfaces

All operation methods return an identificationID. If an operation cannot be processed because of an error detected before the asynchronous processing of the method begins (i.e. before the calling thread returns) a JxfsException is thrown.
After processing has taken place, a JxfsOperationCompleteEvent is generated which contains detailed information about the status of the operation, i.e., if it failed or succeeded, and eventually additional data as a result.

The Constants, Error Codes, Exceptions, Status Codes and Support Classes that are used in the methods are described in special chapters at the end of the documentation.

4.1 Access to properties

Please note the following when determining the meaning of a property's **Access**:

R	The property is read only.
W	The property is write only.
R/W	The property may be read or written.

To access these properties the applications must use the appropriated methods specified by the JavaBean specification.

get*Property*

Syntax	**Property *get*Property *() throws JxfsException***
Description	Returns the requested property.
Parameter	**None**
Event	No additional events are generated.
Exceptions	Some possible JxfsException *value codes*: JXFS_E_CLOSED JXFS_E_UNREGISTERED JXFS_E_REMOTE

set*Property*

Syntax	**void *set*Property *(value) throws JxfsException***
Description	Sets the requested property.
Parameter	The desired property value.
Event	No additional events are generated
Exceptions	Some possible JxfsException *value codes*: JXFS_E_CLOSED JXFS_E_UNREGISTERED JXFS_E_REMOTE JXFS_E_PARAMETER_INVALID

4.2 IJxfsCheckReaderControl

4.2.1 Introduction

The J/XFS CheckReader Device Control Subclass is defined in JxfsCheckReader and is a subclass of JxfsBaseControl. Its interface is defined in IJxfsCheckReaderControl interface which is a subclass of IJxfsBaseControl interface. The purpose of the J/XFS CheckReader Device Control object is to allow passing data and control between the application and the device support code so that the associated device can be accessed.

This is a base class intended for handling of check readers/scanners without printing nor sorting capabilities. Should a device have these additional functions, its Device Control will also implement the *IJxfsComplexCheckDevice* interface.

Summary

Although IJxfsCheckReaderControl is an interface, and therefore properties do not apply, properties are detailed here with the objective to provide guidance on the implementation of those classes that will implement this interface.

Therefore, the IJxfsCheckReaderControl consists on the following methods:

- Getters of listed properties.
- Methods listed.

Property	Type	Access	Initialized after
complex	boolean	R	After successful open
readMICR	boolean	R	After successful open
readOCR	boolean	R	After successful open
imageCapture	int	R	After successful open
readFonts	java.util.Vector	R	After successful open
mediaStatus	JxfsMediaStatus	R	After successful open
lampStatus	int	R	After successful open

Method	Return	May be used after
Get*Property*	*Property*	After successful open
readData	identificationID	After successful open

4.2.2 Properties

complex Property (R)

Type	***boolean***
Initial Value	Depends on device type.
Description	Indicates if the device is a complex one or not, i.e., if it has automatic feeding, sorting and/or printing capabilities

readMICR Property (R)

Type	***boolean***
Initial Value	Depends on device type.
Description	Indicates if the device can read MICR on checks. True means it can read MICR, false it cannot.

readOCR Property (R)

Type	***boolean***
Initial Value	Depends on device type.
Description	Indicates if the device can read OCR on checks. True means it can read OCR, false it cannot.

imageCapture Property (R)

Type	***int***
Initial Value	Depends on device type.
Description	Indicates image capture is supported if any. Depending on the device type it will be set with one of the following values:

Value	**Meaning**
JXFS_CHK_IMAGE_NONE	Image capture is not supported.
JXFS_CHK_IMAGE_FRONT	Front image capture is supported.
JXFS_CHK_IMAGE_REAR	Rear image capture is supported.
JXFS_CHK_IMAGE_BOTH	Front and rear image capture are supported.

readFonts Property (R)

Type	***java.util.Vector***
Initial Value	Depends on device type.
Description	It holds a vector of strings with the names of all the fonts supported for reading.

mediaStatus Property (R)

Type	***JxfsMediaStatus***
Initial Value	A JxfsMediaStatus (see related section in Base Architecture document).
Description	Specifies the state of the media.
Event	If the value of this property changes, the Device Service will send all registered StatusListeners a Status Event with the following values:

Field	**Value**
status	JXFS_S_CHK_MEDIA_STATUS *mediaStatus* has changed.
details	A ***JxfsMediaStatus*** object.

lampStatus Property (R)

Type	***int***
Initial Value	Depends on device status at open.
Description	Specifies the status of the check reader imaging lamp as one of the following values:

Value	**Meaning**
JXFS_CHK_LAMP_OK	The lamp is OK.
JXFS_CHK_LAMP_FADING	The lamp should be changed.

Event	If the value of this property changes, the Device Service will send all registered Status Listeners a JxfsStatusEvent with a status value of:

Field	**Value**
status	JXFS_S_CHK_LAMP_STATUS *lampStatus* has changed.
details	None.

4.2.3 Methods

readData

Syntax

identificationID readData () throws JxfsException;

identificationID readData (boolean getImage) throws JxfsException;

Description

This method launches a read operation to obtain the check identification data as well as image data from the check if requested.

If media is present, the read operation is performed immediately. Otherwise, the device waits until it is present or the operation is cancelled.
After a successful completion of this input operation, a *JxfsOperationCompleteEvent* event is issued to inform the application of the results.

Absence of getImage parameter implies a value of false for it.

Parameter

Type	Name	Meaning
boolean	getImage	Specifies if image data from the check must be returned or not.

Event

JxfsOperationCompleteEvent
When a *readData ()* operation is completed a *JxfsOperationCompleteEvent* event will be sent by the CheckReader Device Control to all registered JxfsOperationComplete Listeners. It will contain the data read.

Field	Value
operationID	JXFS_O_CHK_READDATA
identificationID	Identification ID of complete operation.
result	Common or device dependent error code. (See section on *Error Codes*).
data	A **JxfsCHKData** object. It contains check identication data as well as image data if requested and available.

JxfsIntermediateEvent
JxfsIntermediateEvent can be sent by CheckReader Device Control to all registered IntermediateListeners

Field	Value
operationID	JXFS_O_CHK_READDATA
identificationID	Identification ID of operation.
reason	JXFS_I_CHK_NO_MEDIA_PRESENT The read operation request cannot progress because there is no media inserted. JXFS_I_CHK_MEDIA_INSERTED The read operation request continues because a media has been inserted.
data	null

Exceptions

Some possible JxfsException *value codes*. See section on JxfsExceptions for other JxfsException value codes.

Value	Meaning
JXFS_E_CHK_NOTSUPPORTEDCAP	The service does not have a capability requested in this command

4.2.4 IJxfsComplexCheckDevice

Introduction

This interface contains those properties and functions required for complex check devices that, for instance, automatically feed checks by the batch past a reader, an encoder, an endorser, to be sorted into one of several pockets.

It is intended that this interface will be implemented by device controls that represent physical devices with these feeding, sorting and/or printing capabilities.

Summary

Although IJxfsComplexCheckDevice is an interface, and therefore properties do not apply, properties are detailed here with the objective to provide guidance on the implementation of those classes that will implement this interface.

Therefore, the IJxfsComplexCheckDevice consists on the following methods:

- Getters of listed properties.
- Methods listed.

Property	Type	Access	Initialized after
autoFeed	boolean	R	After successful open
endorser	boolean	R	After successful open
encoder	boolean	R	After successful open
stamp	int	R	After successful open
numPockets	int	R	After successful open
encodeFonts	java.util.Vector	R	After successful open
autoFeedOn	boolean	R	After successful open
inkStatus	int	R	After successful open

Method	Return	May be used after
get*Property*	*Property*	After successful open
processCheck	identificationID	After successful open
setAutoFeed	identificationID	After successful open

4.2.5 Properties

autoFeed Property (R)

Type	***boolean***
Initial Value	Depends on device type.
Description	Indicates if the device has batch autofeed capability. True means it has autofeed capability, false means it doesn't.

endorser Property (R)

Type	***boolean***
Initial Value	Depends on device type.
Description	Indicates if the device has a programmable endorser. True means it does have one, false it doesn't.

encoder Property (R)

Type	***boolean***
Initial Value	Depends on device type.
Description	Indicates if the device has an encoder. True means it does have one, false it doesn't.

stamp Property (R)

Type	***int***
Initial Value	Depends on device type.
Description	Indicates supported stamping modes if any. Depending on the device type it will be set with one of the following values:

Value	**Meaning**
JXFS_CHK_STAMP_NONE	Stamping is not supported.
JXFS_CHK_STAMP_FRONT	Front stamping is supported.
JXFS_CHK_STAMP_REAR	Rear stamping is supported.
JXFS_CHK_STAMP_BOTH	Front and rear stamping are supported.

numPockets Property (R)

Type	***int***
Initial Value	Depends on device type.
Description	Indicates the number of pockets the device has. If 0 or 1, the device has no pockets.

encodeFonts Property (R)

Type	***java.util.Vector***
Initial Value	Depends on device type.
Description	It holds a vector of strings with the names of all the fonts supported for encoding.

autoFeedOn Property (R)

Type	***boolean***
Initial Value	Same value as *autoFeed* property.
Description	Indicates if the device has the autofeed capability activated or not. True means it is activated, false means it isn't.

inkStatus Property (R/W)

Type	***int***
Initial Value	Depends on device status at open.
Description	Specifies the status of the ink in the check reader as one of the following values:

Value	**Meaning**
JXFS_CHK_INK_FULL	Ink supply in device is full.
JXFS_CHK_INK_LOW	Ink supply in device is low.
JXFS_CHK_INK_OUT	Ink supply in device is empty.

Event If the value of this property changes, the Device Service will send all registered Status Listeners a JxfsStatusEvent with a status value of:

Field	**Value**
status	JXFS_S_CHK_INK_STATUS *inkStatus* has changed.
details	None.

4.2.6 Methods

processCheck

Syntax

identificationID processCheck (JxfsCHKProcessData processData) throws JxfsException;

Description

This method is used to encode the amount field of the current check, optionally stamp and endorse the check, and select a pocket to which the check will be sorted if the device supports these capabilities.

Parameter

Type	Name	Meaning
JxfsCHKProcessData	processData	Object that holds all the required data for check processing.

Event

JxfsOperationCompleteEvent

When a *processCheck ()* operation is completed a *JxfsOperationCompleteEvent* event will be sent by CheckReader Device Control to all registered JxfsOperationComplete Listeners.

Field	Value
operationID	JXFS_O_CHK_PROCESS
identificationID	Identification Id of complete operation.
result	Common or device dependent error code. (See section on *Error Codes*).
data	null

Exceptions

Some possible JxfsException *value codes*. See section on JxfsExceptions for other JxfsException value codes.

Value	Meaning
JXFS_E_CHK_NOTSUPPORTEDCAP	The service does not have a capability requested in this command

setAutoFeed

Syntax

identificationID setAutoFeed (boolean onOff) throws JxfsException;

Description

This method is used to activate or deactivate the autofeed mechanism if the device supports this capability. Current status is shown by *autoFeedOn* property.

Parameter

Type	Name	Meaning
boolean	onOff	If true, specifies that the autofeed mechanism should be turned on. If false, specifies that the autofeed mechanism should be turned off.

Event

JxfsOperationCompleteEvent

When a *setAutoFeed ()* operation is completed a *JxfsOperationCompleteEvent* event will be sent by CheckReader Device Control to all registered JxfsOperationComplete Listeners.

Field	Value
operationID	JXFS_O_CHK_AUTOFEED
identificationID	Identification Id of complete operation.
result	Common or device dependent error code. (See section on *Error Codes*).
data	null

Exceptions

Some possible JxfsException *value codes*. See section on JxfsExceptions for other JxfsException value codes.

Value	Meaning
JXFS_E_CHK_NOTSUPPORTEDCAP	The service does not have a capability requested in this command

5 Support Classes

5.1 JxfsCHKData

Summary

This class contains the data returned by a *JxfsOperationCompleteEvent* event for *readData()* operation.

Implements: -- **Extends:** JxfsType

Property	Type	Access	Initialized after
checkData	java.lang.String	R	
checkImage	byte[]	R	

Method	Return	May use after
get*Property*	*Property*	
JxfsCHKData	(constructor of the class)	

5.1.1 Properties

checkData Property (R)

Type ***java.lang.String***
Description Contains the raw data read from the current check.

checkImage Property (R)

Type ***byte[]***
Description Contains the image data from the current check in TIFF 6.0 format if requested and available. Otherwise it is *null*.

5.1.2 Methods

JxfsCHKData Constructor

Syntax ***JxfsCHKData (java.lang.String checkData)***

JxfsCHKData (java.lang.String checkData, byte[] checkImage)
Description Constructor of the class.

5.2 JxfsCHKProcessData

This class provides properties to specify which type of process should be applied to the current check.

Summary

Implements: -- **Extends:** JxfsType

Property	Type	Access	Initialized after
stampFront	boolean	R/W	
stampBack	boolean	R/W	
stampX	int	R/W	
stampY	int	R/W	
endorseFront	boolean	R/W	
endorseBack	boolean	R/W	
sortOnly	boolean	R/W	
pocket	int	R/W	
encodeData	java.lang.String	R/W	
encodeFont	java.lang.String	R/W	
endorseData	java.lang.String	R/W	

Method	Return	May use after
get*Property*	*Property*	
set*Property*	*void*	
JxfsCHKProcessData	(constructor of the class)	

5.2.1 Properties

stampFront Property (R/W)

Type ***boolean***
Description Specifies whether the check must be stamped at the front page or not.

stampBack Property (R/W)

Type ***boolean***
Description Specifies whether the check must be stamped at the back page or not.

stampX Property (R/W)

Type ***int***
Description Specifies the horizontal position for stamping (if selectable) expressed in millimeters from the left hand side of the check.

stampY Property (R/W)

Type ***int***
Description Specifies the vertical position for stamping (if selectable) expressed in millimeters from the top of the check.

endorseFront Property (R/W)

Type ***boolean***
Description Specifies whether the check must be endorsed at the front page or not.

endorseBack Property (R/W)

Type ***boolean***
Description Specifies whether the check must be endorsed at the back page or not.

sortOnly Property (R/W)

Type ***boolean***
Description Specifies whether the process applied to the check must be just sorting or not.

pocket Property (R/W)

Type ***int***
Description Specifies destination pocket. It is ignored if no sorter is present. Pockets are numbered starting from 0.

encodeData Property (R/W)

Type ***java.lang.String***
Description Contains the data to be encoded.

encodeFont Property (R/W)

Type ***java.lang.String***
Description Contains the font to be used when encoding.

endorseData Property (R/W)

Type ***java.lang.String***
Description Contains the data required for endorsement.

5.2.2 Methods

JxfsCHKProcessData Constructor

Syntax	***JxfsCHKProcessData (boolean stampFront, boolean stampBack, int stampX, int stampY, boolean endorseFront, boolean endorseBack, boolean sortOnly, int pocket, java.lang.String encodeData, java.lang.String encodeFont, java.lang.String endorseData)***
Description	Constructor of the class.

6 Enum Classes

6.1 JxfsCHKStatusSelectorEnum

This enumeration class is used for the base *getStatus(java.util.List)* method.

Extends	Implements
JxfsStatusSelectorEnum	

Field	Returned Type	Description
status	JxfsStatus	General status of the device.
mediaStatus	JxfsMediaStatus	Status of the current media.
lampStatus	Integer	Status of the check reader imaging lamp
inkStatus	Integer	Status of the ink in the check reader. This status is available only if the device service implements the *IJxfsComplexCheckDevice* interface
autoFeedOn	Boolean	Indicates if the device has the autoFeed feature activated or not. This status is only available if the device service implements the *IJxfsComplexCheckDevice* interface.

7 Codes

7.1 Error Codes

Value	Meaning
JXFS_E_CHK_READFAILURE	No read conditions were satisfied.
JXFS_E_CHK_NOMEDIA	Media was removed before operation completion
JXFS_E_CHK_INVALIDMEDIA	No appropiated media was found.
JXFS_E_CHK_MEDIAJAMMED	Media is jammed.
JXFS_E_CHK_NOTSUPPORTEDCAP	The service does not have a capability requested in a command.
JXFS_E_CHK_PRINTERROR	No print conditions were satisfied.
JXFS_E_CHK_SWITCHFAILURE	Autofeed could not be changed.

7.2 Status Codes

Value	Meaning
JXFS_S_CHK_MEDIA_STATUS	*mediaStatus* property has changed.
JXFS_S_CHK_LAMP_STATUS	*lampStatus* has changed.
JXFS_S_CHK_INK_STATUS	*inkStatus* has changed.

7.3 Operation Codes

The following codes identify the operation that generated a JxfsOperationCompleteEvent or JxfsIntermediateEvent:

Value	Method
JXFS_O_CHK_READDATA	*readData*
JXFS_O_CHK_PROCESS	*processCheck*
JXFS_O_CHK_AUTOFEED	*setAutoFeed*

The following codes identify the reason for a JxfsIntermediateEvent:

Value	Meaning
JXFS_I_CHK_NO_MEDIA_PRESENT	The read operation request cannot progress because there is no media inserted.
JXFS_I_CHK_MEDIA_INSERTED	The read operation request continues because a media has been inserted.

7.4 Constants

Value	Meaning
JXFS_CHK_IMAGE_NONE	Image capture is not supported.
JXFS_CHK_IMAGE_FRONT	Front image capture is supported.
JXFS_CHK_IMAGE_REAR	Rear image capture is supported.
JXFS_CHK_IMAGE_BOTH	Front and rear image capture are supported.
JXFS_CHK_LAMP_OK	The lamp is OK.
JXFS_CHK_LAMP_FADING	The lamp should be changed.
JXFS_CHK_STAMP_NONE	Stamping is not supported.
JXFS_CHK_STAMP_FRONT	Front stamping is supported.
JXFS_CHK_STAMP_REAR	Rear stamping is supported.
JXFS_CHK_STAMP_BOTH	Front and rear stamping are supported.
JXFS_CHK_INK_FULL	Ink supply in device is full.
JXFS_CHK_INK_LOW	Ink supply in device is low.
JXFS_CHK_INK_OUT	Ink supply in device is empty.

7.5 Constant Definitions

7.5.1 Error Codes

Constant	Numerical Value
JXFS_E_CHK_READFAILURE	9001
JXFS_E_CHK_NOMEDIA	9002
JXFS_E_CHK_INVALIDMEDIA	9003
JXFS_E_CHK_MEDIAJAMMED	9004
JXFS_E_CHK_NOTSUPPORTEDCAP	9005
JXFS_E_CHK_PRINTERROR	9006
JXFS_E_CHK_SWITCHFAILURE	9006

7.5.2 Status Codes

Constant	Numerical Value
JXFS_S_CHK_MEDIA_STATUS	9020
JXFS_S_CHK_LAMP_STATUS	9021
JXFS_S_CHK_INK_STATUS	9022

7.5.3 Operation ID Codes

Constant	Numerical Value
JXFS_O_CHK_READDATA	9023
JXFS_O_CHK_PROCESS	9024
JXFS_O_CHK_AUTOFEED	9025

7.5.4 Intermediate Codes

Constant	Numerical Value
JXFS_I_CHK_NO_MEDIA_PRESENT	9026
JXFS_I_CHK_MEDIA_INSERTED	9027

7.5.5 General Codes

imageCapture Property Codes

Constant	Numerical Value
JXFS_CHK_IMAGE_NONE	9007
JXFS_CHK_IMAGE_FRONT	9008
JXFS_CHK_IMAGE_REAR	9009
JXFS_CHK_IMAGE_BOTH	9010

lampStatus Property Codes

Constant	Numerical Value
JXFS_CHK_LAMP_OK	9011
JXFS_CHK_LAMP_FADING	9012

stamp Property Codes

Constant	Numerical Value
JXFS_CHK_STAMP_NONE	9013
JXFS_CHK_STAMP_FRONT	9014
JXFS_CHK_STAMP_REAR	9015
JXFS_CHK_STAMP_BOTH	9016

inkStatus Property Codes

Constant	Numerical Value
JXFS_CHK_INK_FULL	9017
JXFS_CHK_INK_LOW	9018
JXFS_CHK_INK_OUT	9019