Reference Guide

Presort Object

32/64 BIT

Multiplatform

MELISSA DATA®

Presort Object

Reference Guide

Melissa Data Corporation

Copyright

Trademarks

Melissa Data Corporation

22382 Avenida Empresa
Rancho Santa Margarita, CA 92688-2112

Phone: 1-800-MELISSA (1-800-635-4772)
Fax: 949-589-5211

E-mail: info@MelissaData.com
Internet: www.MelissaData.com

For the most recent version of this document, visit
http://www.melissadata.com/

Document Code: DQTAPIPORG
Revision Number: 26042016.10

Dear Developer,

I would like to take this opportunity to thank you for your interest in Melissa Data products and introduce you to the company.

Melissa Data has been a leading provider of data quality and address management solutions since 1985. Our data quality software, Cloud services, and data integration components verify, standardize, consolidate, enhance and update U.S., Canadian, and global contact data, including addresses, phone numbers, and email addresses, for improved communications and ROI. More than 5,000 companies rely on Melissa Data to gain and maintain a single, accurate and trusted view of critical information assets.

This manual will guide you through the functions of our easy-to-use programming tools. Your feedback is important to me, so please don't hesitate to email your comments or suggestions to me at: Ray@MelissaData.com.

I look forward to hearing from you.

Best Wishes,

Raymond F. Melissa

President/CEO

Table of Contents

Setup Methods

InitializeDataFiles

Required.

Connects the current instance of Presort Object to its data files in preparation for a new presort operation.

This must be called before any other operations can be performed. Setting properties or calling another method without first calling this method will generate an error.

You must set the SetPathToPresortDataFiles property with a valid path and call the SetLicenseString method before calling this method.

Return: Enumerated Value.

Returns a non-zero enumerated value of the type ProgramStatus.

If Presort Object initialization is successful, this method will return zero. If initialization is not successful, call the GetInitializeErrorString method to determine the cause. Any other result indicates that the object did not initialize successfully.

Syntax

```
ProgramStatus = object->InitializeDataFiles()
```

C

```
long int = mdPresortInitializeDataFiles(object)
```

COM

```
Dim result as PresortObjectLib.ProgramStatus
result = object.InitializeDataFiles()
```

SetLicenseString

Required.

Input: String Value.

Sets the license string which enables the use of Presort Object.

Without the license string, the object will not function.

This property must be set before calling the InitializeDataFiles method.

You may also use environment variables to set the license string. For information on how to do this, please refer to: http://www.melissadata.com/tech/address-object-faq/using-address-object/index.asp?CATE=0#1

Return: Boolean Value.

If a True value is returned, the license string was valid.

If a False value is returned, the license string was invalid.

Syntax

```
BooleanValue = object->SetLicenseString(StringValue)
```

C

```
int = mdPresortSetLicenseString(object, *char)
```

COM

```
BooleanValue = object.SetLicenseString(StringValue)
```

SetPathToPresortDataFiles

Required.

Input: String Value.

Sets the path to the directory containing the data files used by Presort Object. This must contain a valid path to the directory containing the Presort Object data files.

The data files are as follows:

3600r_t.pdf	mdL004A.dat	mdL201A.dat
3602n_T.pdf	mdL004B.dat	mdL201B.dat
3602r_T.pdf	mdL004C.dat	mdL601.dat
mdCityState.dat	mdL005.dat	mdL602.dat
mdDropship.dat	mdL007.dat	mdL801A.dat
mdDSF.dat	mdL008.dat	mdL801B.dat
mdL001.dat	mdL009A.dat	mdPresort.lic
mdL002A.dat	mdL009B.dat	mdScheme5.dat
mdL002B1.dat	mdL010.dat	mdValid5.dat
mdL002B2.dat	mdL011A.dat	PostalRatesFiles.txt
mdL002C.dat	mdL011B.dat	

This property must be set before calling the InitializeDataFiles method.

Syntax

```
    void = object->SetPathToPresortDataFiles(StringValue)
```

C

```
    void = mdPresortSetPathToPresortDataFiles(object, *char)
```

COM

```
    object.PathToPresortDataFiles = StringValue
```

GetInitializeErrorString

Optional.

Return: String Value.

Returns a description of the error generated by an unsuccessful call to InitializeDataFiles.

Call this method if InitializeDataFiles fails to initialize Presort Object.

Syntax
`StringValue = object->GetInitializeErrorString()`
C
`StringValue = mdPresortGetInitializeErrorString(object)`
COM
`StringValue = object.GetInitializeErrorString()`

GetParametersErrorString

Optional.

Return: String Value.

Returns a description of any error generated by calls to UpdateParameters, AddRecord, PSSetPrecanceledStampValue, or DoPresort.

If any of these return a false value when called, use this method to return a string value describing the cause for the first error encountered.

Syntax
`StringValue = object->GetParametersErrorString()`
C
`StringValue = mdPresortGetParametersErrorString(object)`
COM
`StringValue = object.GetParametersErrorString()`

GetBuildNumber

Optional.

Input: String Value.

Returns the build number of the currently installed Presort Object.

Have this number available when contacting Melissa Data for technical support.

Syntax

```
StringValue = object->GetBuildNumber()
```

C

```
StringValue = mdPresortGetBuildNumber(object)
```

COM

```
StringValue = object.GetBuildNumber()
```

GetLicenseStringExpirationDate

Optional.

Input: Date Value.

Returns the date when your current license string will expire and Presort Object will no longer function.

To use Presort Object after this date, you will need to contact Melissa Data's sales department at 1-800-MELISSA and purchase an extension to your subscription.

Syntax

```
DateValue = object->GetLicenseStringExpirationDate()
```

C

```
DateValue = mdPresortGetLicenseStringExpirationDate(object)
```

COM

```
DateValue = object.GetLicenseStringExpirationDate()
```

GetDatabaseExpirationDate

Optional.

Input: Date Value.

Returns the date when your current data files will expire.

This allows you to confirm that the data files that you are using are the latest ones. If this method is called before InitializeDataFiles, an exception will occur.

Syntax

```
DateValue = object->GetDatabaseExpirationDate()
```

C

```
DateValue = mdPresortGetDatabaseExpirationDate(object)
```

COM

```
DateValue = object.GetDatabaseExpirationDate()
```

GetDatabaseDate

Optional.

Input: Date Value.

Returns the date of the current database.

If this method is called before InitializeDataFiles, an exception will occur.

Syntax

```
DateValue = object->GetDatabaseDate()
```

C

```
DateValue = mdPresortGetDatabaseDate(object)
```

COM

```
DateValue = object.GetDatabaseDate()
```

Input Methods

Sort Settings

SetBusinessResidentialIndicator

Optional.

Input: String Value.

Indicates whether the current record is a business or residential address. It accepts either a "B" for business address or "R" for a residential address.

SetBusinessResidentialIndicator is required to receive Saturation and High Density discounts.

Syntax

```
void = object-SetBusinessResidentialIndicator(StringValue)
```

C

```
void = mdPresortSetBusinessResidentialIndicator
    (object, *char)
```

COM

```
object.BusinessResidentialIndicator = StringValue
```

SetCarrierRoute

Optional.

Input: String Value.

Sets the Carrier Route number for the next record to be added to the sort.

SetCarrierRoute is required to presort for carrier route postage, Saturation, High Density, and eLOT® discounts

Syntax

```
void = object->SetCarrierRoute(StringValue)
```

C

```
void = mdPresortSetCarrierRoute(object, *char)
```

COM

```
object.CarrierRoute = StringValue
```

SetContinueContainerNumber

Optional.

Input: Boolean Value.

Enables continuous numbering of containers between sortation levels.

If set to True, the numbering of containers on reports will be contiguous through all containers, regardless of sortation levels.

If set to False, numbering on the reports will restart between sortation levels.

For example, if a list includes 50 Automation and 40 Nonautomation containers, and this method is set to false, then the numbering of the Automation containers will be 1 through 50 and the Nonautomation containers will be numbered 1 through 40. If this method were set to True, then the Non-Automation containers will be number 51 through 90.

Syntax

```
void = object->SetContinueContainerNumber(BooleanValue)
```

C

```
void = mdPresortSetContinueContainerNumber(object, int)
```

COM

```
object.ContinueContainerNumber = BooleanValue
```

SetDeliveryPointCode

Optional.

Input: String Value.

Sets the Delivery Point Code for the current record.

The Delivery Point Code is a two digit value used to generate the Delivery Point Barcode number for each address. For street addresses it is usually the last two digits of the street number.

If you have the full twelve-digit delivery point barcode number, the Delivery Point Code is found at digits eleven and twelve.

SetDeliveryPointCode is required for Automation, Saturation, High Density, and eLot discounts.

Syntax
`void = object->DeliveryPointCode(StringValue)`
C
`void = object->DeliveryPointCode(StringValue)`
COM
`object.DeliveryPointCode = StringValue`

SetIMBSerialNumber

Optional.

Input: All Digits String Value.

Sets the Intelligent Mail barcode number.

This allows you to assign a distinct number for each mail piece in the IMB.

For more information see “Intelligent Mail barcode” on page 106.

Syntax
`void = object->SetIMBSerialNumber(AllDigitsStringValue)`
C
`void = mdPresortSetIMBSerialNumber(AllDigitsStringValue)`
COM
`object.IMBSerialNumber = AllDigitsStringValue`

SetLOTNumber

Optional.

Input: String Value.

Sets the Line-Of-Travel number required for any Standard Mail® LOT sortation.

The Line-of-Travel number designates approximately where the submitted address falls within the ZIP + 4®.

SetLOTNumber is required if the sortation selected using SetPresortSettings contains the designation "LOT."

Syntax

```
void = object->SetLOTNumber(StringValue)
```

C

```
void = mdPresortSetLOTNumber(object, *char)
```

COM

```
object.LOTNumber = StringValue
```

SetLOTOrder

Optional.

Input: String Value.

Sets the indicator for ascending or descending Line-of-Travel order. Set to "A" for ascending order (lowest to highest LOT Number) and "D" for descending (highest to lowest LOT Number.)

The Line-of-Travel Order designates whether the Post Office delivers mail in the current ZIP + 4 in ascending or descending order according to the LOT Number.

SetLOTOrder is required if the sortation selected using SetPresortSettings contains the designation "LOT."

Syntax

```
void = object->SetLOTOrder(StringValue)
```

C

```
void = mdPresortSetLOTOrder(object, *char)
```

COM

```
object.LOTOrder = StringValue
```

SetMailersID

Required.

Input: String Value. Six or nine numeric characters.

Sets the Mailer ID issued by the USPS®. This number is used by the Intelligent Mail® barcode system and is different from the mail permit number.

You can apply for a Mailer ID by visiting USPS.COM® and clicking on the Business Customer Gateway link at the bottom of the page.

Syntax

```
void = object->SetMailersID(StringValue)
```

C

```
void = mdPresortSetMailersID(object, *char)
```

COM

```
object.MailersID = StringValue
```

SetPlus4

Optional.

Input: String Value.

Sets the 4-digit ZIP Code add-on for the current record. If the ZIP + 4 for the address is 92688-2112, then the Plus4 would be 2112.

SetPlus4 is required in order to receive the Automation discount. You must use a CASS certified product like Melissa Data's Address Object to append Plus4 codes to your database.

Syntax

```
void = object->SetPlus4(StringValue)
```

C

```
void = mdPresortSetPlus4(object, *char)
```

COM

```
object.Plus4 = StringValue
```

SetPSPresortResidualPieces

Optional.

Input: Boolean Value. Default setting is True.

If set to True, this will include residual mail pieces (mail pieces that do not qualify for any sortation discounts) on the Postage Statement. For First-Class, the residual report will be included on the First-Class Postage Statement. For Standard, the residual report will be on a second First-Class Postage Statement.

If set to False, this will exclude the residual mail pieces.

Syntax

```
void = object->SetPSPresortResidualPieces(BooleanValue)
```

C

```
void = mdPresortSetPSPresortResidualPieces(object, int)
```

COM

```
object.PSPresortResidualPieces = BooleanValue
```

SetRecordID

Optional.

Input: String Value.

Sets a unique identifier that aids in linking each record in the presorted list back to a single address record in the original database.

After a successful call to the DoPresort method, you can pass value to the GetRecord method to retrieve records. It is also returned by the GetRecordID method when you call the GetFirstRecord and GetNextRecord methods, enabling you to match the records in the presorted list to the original database.

Syntax

```
void = object->SetRecordID(StringValue)
```

C

```
void = mdPresortSetRecordID(object, *char)
```

COM

```
object.RecordID = StringValue
```

SetSackWeight

Optional.

Input: Single Precision Value.

Sets the maximum weight of a single sack of Standard Flats to either 30 or 70 pounds.

Sacks for presorted Standard flats come in either 30 or 70 pound sizes. 70 pound sacks are used by default. Set the value to 30 to change to the smaller sack size.

Syntax

```
void = object->SetSackWeight(SinglePrecisionValue)
```

C

```
void = mdPresortSetSackWeight(object, float)
```

COM

```
SackWeight = SinglePrecisionValue
```

SetWalkSequence

Optional.

Input: String Value.

Sets the walk sequence number for the current record.

The Walk Sequence number represents the exact order of delivery for addresses within a given carrier route. Each address within the carrier route will have a unique Walk Sequence number.

Sorting according to the Walk Sequence number puts the mail pieces in the exact order of delivery, so Walk Sequence sortations will have the lowest postage rates.

Walk Sequence numbers are not returned by CASS Certified address verification tool such as Melissa Data's Address Object. A Walk Sequence number must be included in a list when purchased or appended by a USPS-certified vendor.

SetWalkSequence is required for Saturation and High Density discounts.

Syntax

```
void = object->SetWalkSequence(StringValue)
```

C

```
void = mdPresortSetWalkSequence(object, *char)
```

COM

```
object.WalkSequence = StringValue
```

SetZip

Required.

Input: String Value.

Sets the 5-digit ZIP Code for the current record.

A 5-digit ZIP Code is necessary to successfully include a record in a presort, so this property must be set before calling the AddRecord method.

Syntax

```
void = object->SetZip(StringValue)
```

C

```
void = mdPresortSetZip(object, *char)
```

COM

```
object.Zip = StringValue
```

Setting Piece Dimensions

After setting or updating the following properties, you must call the UpdateParameters method before calling either the AddRecord or DoPresort methods.

SetPieceHeight

Required.

Input: Single Precision Value.

Sets the dimension, in inches, of the edge of the mail piece perpendicular to the address as read.

- For letters the height must be between 3.5 and 6.125 inches.
- For cards, the height must be between 3.5 and 4.25 inches. (First-Class Only)
- For flats, the height must be between 5 and 12 inches.

Syntax

```
void = object->SetPieceHeight(SinglePrecisionValue)
```

C

```
void = mdPresortSetPieceHeight(object, float)
```

COM

```
object.PieceHeight = SinglePrecisionValue
```

SetPieceLength

Required.

Input: Single Precision Value.

Sets the dimension, in inches, of the edge of the mail piece parallel to the address as read.

- For letters the length must be between 5 and 11.5 inches.
- For cards, the length must be between 5 and 6 inches. (First-Class Only)
- For flats the length must be between 6 and 15 inches.

Syntax
`void = object->SetPieceLength(SinglePrecisionValue)`
C
`void = mdPresortSetPieceLength(object, float)`
COM
`object.PieceLength = SinglePrecisionValue`

SetPieceThickness

Required.

Input: Single Precision Value.

Sets the thickness, in inches, of the single mail piece from the current mailing.

To follow the USPS method of calculating the thickness, place 10 to 20 mail pieces in a container and tilt the container at 45 degrees. After the pieces have compressed naturally, you can measure the thickness of the stack and calculate the average thickness of each piece.

- For letters, the thickness can be between 0.007 and 0.25 inches.
- For cards, the thickness can be between 0.007 and 0.016 inches. (First-Class Only)
- For flats, the thickness must be between 0.007 and 0.75 inches.

Syntax
`void = object->SetPieceThickness(SinglePrecisionValue)`
C
`void = mdPresortSetPieceThickness(object, float)`
COM
`object.PieceThickness = SinglePrecisionValue`

SetPieceWeight

Required.

Input: Single Precision Value.

Sets the weight, in ounces, of a single mail piece.

- The maximum weight for letters and cards is 3.5 ounces.
- The maximum weight for flats is 13 ounces for First-Class Mail® and 16 ounces for Standard Mail.

Syntax
`void = object->SetPieceWeight(SinglePrecisionValue)`
C
`void = mdPresortSetPieceWeight(object, float)`
COM
`object.PieceWeight = SinglePrecisionValue`

Postage Statement Methods

Permit Holder Contact Information

The following properties are used to enter the contact information for the permit holder for the current mailing. This information will appear on the Qualification Report and Postage Statement.

SetPSPermitHolderName

Required.

Input: String Value.

Sets the permit holder's contact name.

Syntax
`void = object->SetPSPermitHolderName(StringValue)`
C
`void = mdPresortSetPSPermitHolderName(object, *char)`
COM
`object.PSPermitHolderName = StringValue`

SetPSPermitHolderCompany

Optional.

Input: String Value.

Sets the permit holder's company name, if any.

Syntax
`void = object->SetPSPermitHolderCompany(StringValue)`
C
`void = mdPresortSetPSPermitHolderCompany(object, *char)`
COM
`object.PSPermitHolderCompany = StringValue`

SetPSPermitHolderStreet

Required.

Input: String Value.

Sets the permit holder's street address.

Syntax

```
void = object->SetPSPermitHolderStreet(StringValue)
```

C

```
void = mdPresortSetPSPermitHolderStreet(object, *char)
```

COM

```
object.PSPermitHolderStreet = StringValue
```

SetPSPermitHolderCity

Required.

Input: String Value.

Sets the city from the permit holder's address.

Syntax

```
void = object->SetPSPermitHolderCity(StringValue)
```

C

```
void = mdPresortSetPSPermitHolderCity(object, *char)
```

COM

```
object.PSPermitHolderCity = StringValue
```

SetPSPermitHolderState

Required.

Input: String Value.

Sets the State from the permit holder's address.

Syntax

```
void = object->SetPSPermitHolderState(StringValue)
```

C

```
void = mdPresortSetPSPermitHolderState(object, *char)
```

COM

```
object.PSPermitHolderState = StringValue
```

SetPSPermitHolderZIP

Required.

Input: String Value.

Sets the permit holder's ZIP Code™.

Syntax

```
void = object->SetPSPermitHolderZIP(StringValue)
```

C

```
void = mdPresortSetPSPermitHolderZIP(object, *char)
```

COM

```
object.PSPermitHolderZip = StringValue
```

SetPSPermitHolderPhone

Required.

Input: String Value.

Sets the permit holder's phone number.

Syntax

```
void = object->SetPSPermitHolderPhone(StringValue)
```

C

```
void = mdPresortSetPSPermitHolderPhone(object, *char)
```

COM

```
object.PSPermitHolderPhone = StringValue
```

SetPSPermitHolderEmail

Required.

Input: String Value.

Sets the permit holder's email address.

Syntax

```
void = object->SetPSPermitHolderEmail(StringValue)
```

C

```
void = mdPresortSetPSPermitHolderEmail(object, *char)
```

COM

```
object.PSPermitHolderEmail = StringValue
```

SetPSPermitHolderListName

Required.

Input: String Value.

Sets the name of the database containing the list being presorted.

Syntax
`void = object->SetPSPermitHolderListName(StringValue)`
C
`void = mdPresortSetPSPermitHolderListName(object, *char)`
COM
`object.PSPermitHolderListName = StringValue`

SetPSNonProfitAuthNumber

Optional.

Input: String Value.

Sets the authorization number needed to claim non-profit status.

The Non-Profit Authorization Number is printed on the Postage Statement. This alerts the Post Office that you are allowed to claim Non-Profit rates.

Syntax
`void = object->SetPSNonProfitAuthNumber(StringValue)`
C
`void = mdPresortSetPSNonProfitAuthNumber(object, *char)`
COM
`object.PSNonProfitAuthNumber = StringValue`

SetPSCAPSNumber

Optional.

Input: String Value.

Sets the Centralized Account Process System (CAPS) number to be printed on the Postage Statement, if any.

CAPS is an electronic postage payment system that provides business mailers a centralized, convenient, and cost-effective way pay for postage. It provides an electronic alternative to presenting checks and cash for postage and fees at multiple post offices.

Syntax

```
    void = object->SetCAPSNumber(StringValue)
```

C

```
    void = mdPresortSetCAPSNumber(object, *char)
```

COM

```
    object.PSCAPSNumber = StringValue
```

SetPSCustomerNumber

Optional.

Input: String Value.

Sets any customer number to be printed on the Postage Statement.

If the mailing is being handled by a mailing house for a client, that client's customer number, if any, can be entered here.

Syntax

```
    void = object->SetCustomerNumber(StringValue)
```

C

```
    void = mdPresortSetCustomerNumber(object, *char)
```

COM

```
    object.PSCustomerNumber = StringValue
```

Mailing Agent Information

The following properties are used to enter the contact information of the mailing agent for the current mailing, if a mailing agent was used.

The mailing agent's contact information appears on the postage statement created by Presort Object.

SetPSMailingAgentName

Optional.

Input: String Value.

Sets the mailing agent's contact name.

Syntax

```
void = object->SetPSMailingAgentName(StringValue)
```

C

```
void = mdPresortSetPSMailingAgentName(object, *char)
```

COM

```
object.PSMailingAgentName = StringValue
```

SetPSMailingAgentCompany

Optional.

Input: String Value.

Sets the mailing agent's company name, if any.

Syntax

```
void = object->SetPSMailingAgentCompany(StringValue)
```

C

```
void = mdPresortSetPSMailingAgentCompany(object, *char)
```

COM

```
object.PSMailingAgentCompany = StringValue
```

SetPSMailingAgentStreet

Optional.

Input: String Value.

Sets the mailing agent's street address.

Syntax
`void = object->SetPSMailingAgentStreet(StringValue)`
C
`void = mdPresortSetPSMailingAgentStreet(object, *char)`
COM
`object.PSMailingAgentStreet = StringValue`

SetPSMailingAgentCity

Optional.

Input: String Value.

Sets the city from the mailing agent's address.

Syntax
`void = object->SetPSMailingAgentCity(StringValue)`
C
`void = mdPresortSetPSMailingAgentCity(object, *char)`
COM
`object.PSMailingAgentCity = StringValue`

SetPSMailingAgentState

Optional.

Input: String Value.

Sets the State from the mailing agent's address.

Syntax
`void = object->SetPSMailingAgentState(StringValue)`
C
`void = mdPresortSetPSMailingAgentState(object, *char)`
COM
`object.PSMailingAgentState = StringValue`

SetPSMailingAgentZIP

Optional.

Input: String Value.

Sets the mailing agent's ZIP Code.

Syntax

```
void = object->SetPSMailingAgentZIP(StringValue)
```

C

```
void = mdPresortSetPSMailingAgentZIP(object, *char)
```

COM

```
object.PSMailingAgentZip = StringValue
```

SetPSMailingAgentPhone

Optional.

Input: String Value.

Sets the mailing agent's phone number.

Syntax

```
void = object->SetPSMailingAgentPhone(StringValue)
```

C

```
void = mdPresortSetPSMailingAgentPhone(object, *char)
```

COM

```
object.PSMailingAgentPhone = StringValue
```

SetPSMailingAgentCRID

Optional.

Input: String Value.

Sets the CRID of the mailing agent. This is the USPS ID.

Syntax

```
void = object->SetPSMailingAgentCRID(StringValue)
```

C

```
void = mdPresortSetPSMailingAgentCRID(object, *char)
```

COM

```
object.PSMailingAgentCRID = StringValue
```

Individual or Organization Information

The following properties set the name for the individual or organization for whom the current mailing is being prepared, if any. This information will appear on the Postage Statement created by Presort Object.

SetPSIndividualOrOrganizationName

Optional.

Input: String Value.

Sets the individual or organization's name.

Syntax

```
void = object->SetPSIndividualOrOrganizationName
    (StringValue)
```

C

```
void = mdPresortSetPSIndividualOrOrganizationName
    (object, *char)
```

COM

```
object.PSIndividualOrOrganizationName = StringValue
```

SetPSIndividualOrOrganizationCompany

Optional.

Input: String Value.

Sets the individual or organization's company name, if any.

Syntax

```
void = object->SetPSInidividualOrOrganizationCompany
    (StringValue)
```

C

```
void = mdPresortSetPSInidividualOrOrganizationCompany
    (object, *char)
```

COM

```
object.PSInidividualOrOrganizationCompany = StringValue
```

SetPSIndividualOrOrganizationStreet

Optional.

Input: String Value.

Sets the individual or organization's street address.

Syntax

```
void = object->SetPSInidividualOrOrganizationStreet
    (StringValue)
```

C

```
void= mdPresortSetPSInidividualOrOrganizationStreet
    (object, *char)
```

COM

```
object.PSInidividualOrOrganizationStreet = StringValue
```

SetPSIndividualOrOrganizationCity

Optional.

Input: String Value.

Sets the city from the party's address.

Syntax

```
void = object->PSIndividualOrOrganizationCity(StringValue)
```

C

```
void = mdPresortPSIndividualOrOrganizationCity
    (object, *char)
```

COM

```
object.PSIndividualOrOrganizationCity = StringValue
```

SetPSIndividualOrOrganizationState

Optional.

Input: String Value.

Sets the State from the individual or organization's address.

Syntax

```
void = object->SetPSInidividualOrOrganizationState
    (StringValue)
```

C

```
void = mdPresortSetPSInidividualOrOrganizationState
    (object, *char)
```

COM

```
object.PSInidividualOrOrganizationState = StringValue
```

SetPSIndividualOrOrganizationZIP

Optional.

Input: String Value.

Sets the individual or organization's ZIP Code.

Syntax

```
void = object->SetPSInidividualOrOrganizationZIP
    (StringValue)
```

C

```
void = mdPresortSetPSInidividualOrOrganizationZIP
    (object, *char)
```

COM

```
object.PSInidividualOrOrganizationZip = StringValue
```

SetPSIndividualOrOrganizationCRID

Optional.

Input: String Value.

Sets the CRID of the individual or organization. This is the USPS ID.

Syntax

```
void = object->SetPSInidividualOrOrganizationCRID
    (StringValue)
```

C

```
void = mdPresortSetPSInidividualOrOrganizationCRID
    (object, *char)
```

COM

```
object.PSInidividualOrOrganizationCRID = StringValue
```

Post Office of Mailing Information

The post office of mailing is the USPS branch that granted the mailing permit being used for the current mailing. It is required to determine the mailing's qualification for automation discounts.

Without this information, you will get an error when you call the DoPresort method.

This information appears on all reports and the Postage Statement.

SetPSPostOfficeOfMailingCity

Required.

Input: String Value.

Sets the city for the post office of mailing.

Syntax

```
void = object->SetPSPostOfficeOfMailingCity(StringValue)
```

C

```
void = mdPresortSetPSPostOfficeOfMailingCity(object, *char)
```

COM

```
object.PSPostOfficeOfMailingCity = StringValue
```

SetPSPostOfficeOfMailingState

Required.

Input: String Value.

Sets the State for the post office of mailing.

Syntax

```
void = object->SetPSPostOfficeOfMailingState(StringValue)
```

C

```
void = mdPresortSetPSPostOfficeOfMailingState
   (object, *char)
```

COM

```
object.PSPostOfficeOfMailingState = StringValue
```

SetPSPostOfficeOfMailingZIP

Required.

Input: String Value.

Sets the ZIP Code for the post office of mailing.

Syntax

```
void = object->SetPSPostOfficeOfMailingZIP(StringValue)
```

C

```
void = mdPresortSetPSPostOfficeOfMailingZIP(object, *char)
```

COM

```
object.PSPostOfficeOfMailingZip = StringValue
```

Postage Type

Postage type defaults to metered when SetPSPermitImprint and SetPSPrecanceledStamp are either not called or set to False.

SetPSPermitImprint

Optional.

Input: Boolean Value. Default value is False.

If set to True, the current mailing will use a permit imprint as the method of postage payment.

If set to False, Presort Object will use metered mail as the payment method.

This information will be indicated on your Postage Statement.

Syntax

```
void = object->SetPSPermitImprint(BooleanValue)
```

C

```
void = mdPresortSetPSPermitImprint(object, int)
```

COM

```
object.PSPermitImprint = BooleanValue
```

SetPSPrecanceledStamp

Optional.

Input: Boolean Value. Default setting is False.

The payment method preference will be indicated on your postage statement.

If set to True, indicates that the current mailing will use a precanceled stamps as the postage payment method and SetPSPrecanceledStampValue must be set.

If set to False, Presort Object will use metered mail as the postage payment method.

Syntax

```
void = object->SetPSPrecanceledStamp(BooleanValue)
```

C

```
void = mdPresortSetPSPrecanceledStamp(object, int)
```

COM

```
object.PSPrecanceledStamp = BooleanValue
```

SetPSPrecanceledStampValue

Optional.

Input: Single Precision Value.

Sets the postage value in cents of a precanceled stamp used for the current mailing. (Ex: 0.45)

This must be called if the SetPSPrecanceledStamp property is set to True.

Return: Boolean Value.

If a True value is returned, the method was called correctly.

If a False value is returned, an error occurred. A value of zero or less was inputted. Call GetParametersErrorString() to determine the cause.

Syntax

```
Boolean = object->SetPSPrecanceledStampValue
    (SinglePrecisionValue)
```

C

```
int = mdPresortSetPSPrecanceledStampValue(object, float)
```

COM

```
Boolean = object.PSPrecanceledStampValue
    (SinglePrecisionValue)
```

Mailing Parameters

SetPSMailingDate

Required.

Input: String Value.

Sets the mailing date to be printed on the Postage Statement. This should reflect the date that the mail pieces are deposited with the Post Office.

Syntax

```
void = object->SetPSMailingDate(StringValue)
```

C

```
void = mdPresortSetPSMailingDate(object, *char)
```

COM

```
object.PSMailingDate = StringValue
```

SetPSFedAgencyCode

Optional.

Input: String Value.

Sets the Federal agency code to be printed on the Postage Statement, if any.

Federal agencies would use this property to record a special agency number. If you are not a federal agency, do not use this property.

Syntax

```
void = object->SetPSFederalAgencyCode(StringValue)
```

C

```
void = mdPresortSetPSFederalAgencyCode(object, *char)
```

COM

```
object.PSFederalAgencyCode = StringValue
```

SetPSStatementSeqNumber

Optional.

Input: String Value.

Sets the postage statement sequence number to be printed on the Postage Statement.

Some mailings need to use more than one Postage Statement. When this happens, the postage statements will have sequence numbers (1 of 4, 2 of 4, etc.). Pass that number to this method.

You do not need to call this method if you are using a single Postage Statement.

Syntax

```
void = object->SetPSStatementSeqNumber(StringValue)
```

C

```
void = mdPresortSetPSStatementSeqNumber(object, *char)
```

COM

```
object.PSStatementSeqNumber = StringValue
```

SetPermitNumber

Required.

Input: String Value.

Sets a unique number used to identify the mailer to the USPS. This information is included on the Postage Statement and Qualification Report.

The mailing permit number is separate from the Mailer ID used for the Intelligent Mail barcode system. This would normally be the bulk permit number assigned by the USPS when you turned in form 3615 at the local Post Office.

For information on applying for a Bulk Mail Permit, visit USPS.COM and click on Business. Follow the links for payment options, under the link about Postage meters.

Syntax

```
void = object->SetPermitNumber(StringValue)
```

C

```
void = mdPresortSetPermitnumber(object, *char)
```

COM

```
object.PermitNumber = StringValue
```

SetPSCASSDate

Required.

Input: String Value.

Sets the date when the current list was subjected to address verification. This date is supplied by the customer and follows the "MMDDYYYY" format. Ex: 06272012.

To ensure the accuracy of move information, the Postage Statement must list the date when the mailing list was processed with a CASS Certified™ address verification software.

Syntax

```
    void = object->SetPSCASSDate(StringValue)
```

C

```
    mdPresortSetPSCASSDate(object, *char)
```

COM

```
    object.PSCASSDate = StringValue
```

Move Update Method

Use these methods to designate the method used to update move (change-of-address) information for the list used in the current mailing. **One and only one of the following methods should called and set to True.** This information will be reflected on the Postage Statement.

SetPSASE

Input: Boolean Value.

If set to True, this mailing uses Ancillary Service Endorsements (ASE) for move updates.

Ancillary Service Endorsements (ASE's) assist in the delivery of your mail or the update of your mailing list. Undeliverable-as-addressed mail is forwarded, returned to the sender, or treated as dead mail as authorized by that particular mail class.

Syntax

```
void = object->SetPSASE(BooleanValue)
```

C

```
void = mdPresortSetPSASE(object, int)
```

COM

```
object.PSASE = BooleanValue
```

SetPSFASTForward

Input: Boolean Value.

If set to True, this mailing uses FASTforward® for move updates.

FASTforward prints the new address on the mail piece just before it enters the mail-stream.

Syntax

```
void = object->SetPSFASTForward(BooleanValue)
```

C

```
void = mdPresortSetPSFASTForward(object, int)
```

COM

```
object.PSFASTForward = BooleanValue
```

SetPSNCOA

Input: Boolean Value.

If set to True, this mailing uses NCOA$^{Link®}$ for move updates.

Use this setting if you have used an NCOALink update service such as Melissa Data's Smart Mover Web Service, NCOALink update service, or Smart Mover Real-time Update.

Syntax

```
void = object->SetPSNCOA(BooleanValue)
```

C

```
void = mdPresortSetPSNCOA(object, int)
```

COM

```
object.PSNCOA = BooleanValue
```

SetPSACS

Input: Boolean Value.

If set to True, this mailing uses Address Change Service (ACS™) for move updates.

In ACS, mailers modify their mailing label format to include a mailer identification (participant) code assigned by the National Customer Support Center (NCSC).

Syntax

```
void = object->SetPSACS(BooleanValue)
```

C

```
void = mdPresortSetPSACS(object, int)
```

COM

```
object.PSACS = BooleanValue
```

SetPSAltMethod

First-Class Mail Only.

Input: Boolean Value.

If set to True, this mailing uses one of two alternate update methods.

In addition to the four USPS-approved move update methods described above, there are two alternative methods. These are only available for First-Class Mail.

Syntax

```
void = object->SetPSAltMethod(BooleanValue)
```

C

```
void = md_PSetPSAltMethod(object, int)
```

COM

```
object.PSAltMethod = BooleanValue
```

SetPSMultiple

Input: Boolean Value.

If set to True, this mailing uses two or more update methods.

Use this setting when more than one move update method was used on the current list.

Syntax

```
void = object->SetPSMultiple(BooleanValue)
```

C

```
void = mdPresortSetPSMultiple(object, int)
```

COM

```
object.PSMultiple = BooleanValue
```

SetPSOneCode

Input: Boolean Value.

If set to True, this mailing uses OneCode ACS® for move updates.

OneCode ACS lets mailers use the barcode to access the agency's electronic Address Change Service (ACS) to obtain move information in the event that someone relocates after a mail piece has entered the mail-stream.

Syntax

```
void = object->SetPSOneCode(BooleanValue)
```

C

```
void = mdPresortSetPSOneCode(object, int)
```

COM

```
object.PSOneCode = BooleanValue
```

SetPSAltAddFmt

Input: Boolean Value.

If set to True, this mailing uses an Alternative Address Format.

Syntax

```
void = object->SetPSAltAddFmt(BooleanValue)
```

C

```
void = mdPresortSetPSAltAddFmt(object, int)
```

COM

```
object.PSAltAddFmt = BooleanValue
```

Tray Tag Properties

The following properties are used to set up tray tags.

SetTTBorder

Optional.

Input: Boolean Value. Default setting is False.

If set to True, enables the printing of a border around each container tag.

If set to False, this is disabled.

This is useful if you print tray tags on standard paper instead of perforated forms, to assist in cutting the tags to the proper size.

Syntax
`void = object->SetTTBorder(BooleanValue)`
C
`void = mdPresortSetTTBorder(object, int)`
COM
`object.TTBorder = BooleanValue`

SetTTNumberOfPieces

Optional.

Input: Boolean Value. Default setting is False.

If set to True, enables the printing of the number of mail-pieces in each container on the first line of each container tag.

If set to False, this is disabled.

Printing the number of mail-pieces on the tray tag is a convenient way to verify your counts when assembling the mailing.

Syntax

```
void = object->SetTTNumberOfPieces(BooleanValue)
```

C

```
void = mdPresortSetTTNumberOfPieces(object, int)
```

COM

```
object.TTNumberOfPieces = BooleanValue
```

SetTTContainerNumber

Optional.

Input: Boolean Value. Default setting is False.

If set to True, enables the printing of the serialized number for each container on the first line of each container tag.

If set to False, this is disabled.

Numbering of containers can be continuous for an entire mailing or it can be restarted for each sortation type. For more information, see SetContinueContainerNumber.

Syntax

```
void = object->SetTTContainerNumber(BooleanValue)
```

C

```
void = mdPresortSetTTContainerNumber(object, int)
```

COM

```
object.TTContainerNumber = BooleanValue
```

SetTTContainerSize

Optional.

Input: Boolean Value. Default setting is False.

For letters only. This sets the printing of the container size (1' or 2') on the first line of each container tag.

If set to True, this will append the container size on the container tags.

If set to False, it will not.

Printed	Container Type
Sack	Sack (Flats)
1'	1-foot Tray (Letters)
2'	2-foot Tray (Letters)
EMM	Large tub for outsized mail pieces

Syntax

```
    void = object->SetTTContainerSize(BooleanValue)
```

C

```
    void = mdPresortSetTTContainerSize(object, int)
```

COM

```
    object.TTContainerSize = BooleanValue
```

SetTTOther

Optional.

Input: String Value.

Sets the contents of additional text that will be printed on the first line of the container tag. This can include any additional information the mailer feels is necessary to identify the contents of the containers in this mailing.

Syntax

```
void = object->SetTTOther(StringValue)
```

C

```
void = mdPresortSetTTOther(object, *char)
```

COM

```
object.TTOther = StringValue
```

SetTTParameterPositionX

Optional.

Input: String Value.

Sets the distance, in points, between the expected left margin and the actual left margin. Enter a positive number to print further to the right or a negative number to print further to left. Decimal values may be used in a string format. Ex: “4.5”

Not all printers are the same and these differences can lead to misalignment when printing container tags on perforated forms. Points, the unit of measure for this property, are a standard unit of measure for printing. Refer to this chart to convert inches to points, if necessary:

1/8" =	9 points
1/6" =	12 points
1/4" =	18 points
1/2" =	36 points
1" =	72 points

It may be necessary to print a sample run of container tags, and then measure the differences between what is printed and the actual dimensions of the form.

Syntax

```
void = object->SetTTParameterPositionX(StringValue)
```

C

```
void = mdPresortSetTTParameterPositionX(object, *char)
```

COM

```
object.TTParameterPositionX = StringValue
```

SetTTParameterPositionY

Optional.

Input: String Value.

Sets the distance, in points, between the expected top margin and the actual top margin. Enter a positive number to print lower or a negative number to print closer to the top of the page. Decimal values may be used in a string format. Ex: “4.5”

Points, the unit of measure for this property, are a standard unit of measure for printing. See the chart above to convert inches to points, if necessary.

Syntax

```
void = object->SetTTParameterPositionY(StringValue)
```

C

```
void = mdPresortSetTTParameterPositionY(object, *char)
```

COM

```
object.TTParameterPositionY = StringValue
```

SetTTParameterWidth

Optional.

Input: String Value.

Scales the width of container tags.

The value of this property is a multiplier that controls how wide the container tags will print. 1.0 = the default size. 1.05 = 5% wider. 0.95 = 5% narrower.

Syntax

```
void = object->SetTTParameterWidth(StringValue)
```

C

```
void = mdPresortSetTTParameterWidth(object, *char)
```

COM

```
object.TTParameterWidth = StringValue
```

SetTTParameterHeight

Optional.

Input: String Value.

Scales the height of container tags.

The value of this property is a multiplier that controls how tall the container tags will print. 1.0 = the default size. 1.05 = 5% taller. 0.95 = 5% shorter.

Syntax

```
void = object->SetTTParameterHeight(StringValue)
```

C

```
void = mdPresortSetTTParameterHeight(object, *char)
```

COM

```
object.TTParameterHeight = StringValue
```

Standard-Only Methods

Standard-Only Methods.

SetSTD_Auto_5dg_Scheme

Optional.

Input: Boolean Value. Default setting is True.

If set to True, enables the Standard Automation 5-Digit Scheme.

If set to False, it will be disabled.

Syntax

```
void = object->SetSTD_Autp_5dg_Scheme(BooleanValue)
```

C

```
void = mdPresortSetSTD_Auto_5dg_Scheme(object, int)
```

SetSTD_Auto_5dg

Optional.

Input: Boolean Value. Default setting is True.

If set to True, enables the Standard Automation 5-Digit.

If set to False, it will be disabled.

Syntax

```
void = object->SetSTD_Auto_5dg(BooleanValue)
```

C

```
void = mdPresortSetSTD_Auto_5dg(object, int)
```

SetSTDNonProfit

Optional.

Input: Boolean Value. Default setting is False.

If set to True, enables Non-Profit postage rates for Standard Mail.

If set to False, normal Standard Mail rates will be used.

Syntax

```
void = object->SetSTDNonProfit(BooleanValue)
```

C

```
void = mdPresortSetSTDNonProfit(object, int)
```

COM

```
object.STDNonProfit = BooleanValue
```

SetIgnoreDSF

Optional.

Input: Boolean Value. Default setting is False.

If set to True, the Delivery Sequence File (DSF) for walk sequence sorting will be ignored.

If set to False, it will not.

This property should only be used when using a list purchased from a list broker who can provide documentation that the list's walk sequence is based on Computerized Delivery Sequence (CDS) data.

Syntax

```
void = object->SetIgnoreDSF(BooleanValue)
```

C

```
void = mdPresortSetIgnoreDSF(object, int)
```

COM

```
object.IgnoreDSF = BooleanValue
```

Destinations Properties

SetProduceDropShipForms

Optional.

Input: Boolean Value.

This method allows you to produce form 8285 and 3602C. These forms are required for drop shipping. On a correct call to the method it will return a value of true.

Syntax
`void = object->SetProduceDropShipForms(BooleanValue)`
C
`void = mdPresortSetProduceDropShipForms(object, int)`
COM
`object.ProduceDropShipForms = BooleanValue`

SetPOMasNDC

Optional.

Input: Boolean Value.

Call and set to True to indicate that the Post Office of Mailings is also an Network Distribution Center (NDC).

Normally, 200 mail pieces are required to claim the NDC destination discount. If your Post Office of Mailing is an NDC, the USPS waives this requirement.

Do not call if the Post Office of Mailings is not an NDC.

Syntax
`void = object->SetPOMasNDC(BooleanValue)`
C
`void = mdPresortSetPOMasNDC(object, int)`
COM
`object.POMasNDC = BooleanValue`

SetNDCCity

Optional.

Input: String Value.

Sets the name of the city containing of a Network Distribution Center.

Syntax

```
void = object->SetNDCCity(StringValue)
```

C

```
void = mdPresortSetNDCCity(object, *char)
```

COM

```
object.NDCCity = StringValue
```

SetNDCState

Optional.

Input: String Value.

Sets the name of the state containing of a Network Distribution Center.

Syntax

```
void = object->SetNDCState(StringValue)
```

C

```
void = mdPresortSetNDCState(object, *char)
```

COM

```
object.NDCState = StringValue
```

SetNDCZip

Optional.

Input: String Value.

Sets the ZIP Code of a Network Distribution Center.

This is normally the first three digits of the ZIP codes serviced by the NDC.

After setting the SetNDCCity, SetNDCState, and SetNDCZip properties, call the AddNDC method to add the NDC to the list of facilities used for the current presort operation.

Syntax

```
void = object->SetNDCZip(StringValue)
```

C

```
void = mdPresortSetNDCZip(object, *char)
```

COM

```
object.NDCZip = StringValue
```

AddNDC

Optional.

Return: Boolean Value.

Uses the current values of the SetNDCCity, SetNDCState and SetNDCZip properties to add an NDC to the list of facilities used for the current presort operation.

If a True value is returned, the NDC is successfully added.

If a False value is returned, it is an invalid destination. One or more of the properties referenced were not populated before calling this method or the destination is not an NDC.

Syntax

```
BooleanValue = object->AddNDC()
```

C

```
int = mdPresortAddNDC(object)
```

COM

```
BooleanValue = object.AddNDC()
```

SetPOMasSCF

Optional.

Input: Boolean Value.

Call and set to True to indicate that the Post Office of Mailing is also an Sectional Center Facility (SCF).

Normally, 200 mail pieces are required to claim the SCF destination discount. If your Post Office of Mailing is an SCF, the USPS waives this requirement.

Do not call if the Post Office of Mailings is not an SCF.

Syntax

```
void = object->SetPOMasSCF(BooleanValue)
```

C

```
void = mdPresortSetPOMasSCF(object, int)
```

COM

```
object.POMasSCF = BooleanValue
```

SetSCFCity

Optional.

Input: String Value.

Sets the name of the city containing a Sectional Center Facility.

Syntax

```
void = object->SetSCFCity(StringValue)
```

C

```
void = mdPresortSetSCFCity(object, *char)
```

COM

```
object.SCFCity = StringValue
```

SetSCFState

Optional.

Input: String Value.

Sets the name of the state containing of a Sectional Center Facility.

Syntax
`void = object->SetSCFState(StringValue)`
C
`void = mdPresortSetSCFState(object, *char)`
COM
`object.SCFState = StringValue`

SetSCFZip

Optional.

Input: String Value.

Sets the ZIP Code of a Sectional Center Facility.

This is normally the first three digits of the ZIP codes serviced by the SCF.

After setting the SetSCFCity, SetSCFState, and SetSCFZip properties, call the AddSCF method to add the SCF to the list of facilities used for the current presort operation.

Syntax
`void = object->SetSCFZip(StringValue)`
C
`void = mdPresortSetSCFZip(object, *char)`
COM
`object.SCFZip = StringValue`

AddSCF

Optional.

Return: Boolean Value.

Uses the current values of the SetSCFCity, SetSCFState, and SetSCFZip properties to add an SCF to the list of facilities used for the current presort operation.

If a True value is returned, the SCF is successfully added.

If a False value is returned, it is an invalid destination. One or more of the properties referenced were not populated before calling this method or the destination is not an SCF.

Syntax

```
BooleanValue = object->AddSCF()
```

C

```
int = mdPresortAddSCF(object)
```

COM

```
BooleanValue = object.AddSCF()
```

SetDDUCity

Optional.

Input: String Value.

Sets the name of the city containing of a Destination Delivery Unit.

Syntax

```
void = object->SetDDUCity(StringValue)
```

C

```
void = mdPresortSetDDUCity(object, *char)
```

COM

```
object.DDUCity = StringValue
```

SetDDUState

Optional.

Input: String Value.

Sets the name of the state containing of a Destination Delivery Unit.

Syntax
`void = object->SetDDUState(StringValue)`
C
`void = mdPresortSetDDUState(object, *char)`
COM
`object.DDUState = StringValue`

SetDDUZip

Optional.

Input: String Value.

Sets the main ZIP Code serviced by a Destination Delivery Unit.

If the DDU facility services more than one five-digit ZIP Code, add these ZIP Codes one at a time using the SetDDUMoreZip property.

After setting the SetDDUCity, SetDDUState, and SetDDUZip properties, call the AddDDU method to add the DDU to the list of facilities used for the current presort operation.

Syntax
`void = object->SetDDUZip(StringValue)`
C
`void = mdPresortSetDDUZip(object, *char)`
COM
`object.DDUZip = StringValue`

SetDDUMoreZip

Optional.

Input: String Value.

Sets additional ZIP codes serviced by the same Destination Delivery Unit, one at time.

If the current DDU services more than one five-digit ZIP Code, use this property to add these ZIP codes, one at a time, before calling the AddDDU method.

Syntax

```
void = object->SetDDUMoreZip(StringValue)
```

C

```
void = mdPresortSetDDUMoreZip(object, *char)
```

COM

```
object.DDUMoreZip = StringValue
```

AddDDU

Optional.

Input: String Value.

This method uses the current values of the SetDDUZip, SetDDUCity and SetDDUState properties to add an DDU to the list of facilities used for the current presort operation.

This method will return a True value if the DDU is successfully added. If one or more of the related properties (SetDDUZip, SetDDUCity, and SetDDUState) were not populated before calling this method, it will return a zero or false value. Check the GetParametersErrorString method to determine the cause of the error.

Syntax

```
BooleanValue = object->AddDDU()
```

C

```
int = mdPresortAddDDU(object)
```

COM

```
BooleanValue = object.AddDDU()
```

Processing Methods

ProduceReports

Optional.

Input: String Path, String Name.

Sets a file path, a file name, then creates and saves a PDF file containing the Mailing Summary, Qualification Reports (QR), Container Tags (TT), Presort Settings Report (PP) and Postage Statement (PS.)

The String Path must be a valid path to an existing directory. It must be called before a call to the DoPresort method.

Naming Conventions:

Mailing Summary, Qualification Reports, Container Tags and Presort Settings Reports:

```
Report Type_Mail Class_Piece Type_Qualification Type_
    String Name.File Extension
```

Postage Statement:

```
Postage Statement Form Number_String Name.File Extension
```

Return: Boolean Value.

If a True value is returned, the requirements were met and the PDF files will be generated.

If a False value is returned, the requirements were not met. Call GetInitializeErrorString() to determine the cause.

Syntax

```
BooleanValue = object->ProduceReports(String Path, String Name)
```

C

```
int = mdPresortProduceReports(*char Path, *char Name)
```

COM

```
BooleanValue = object.ProduceReports(String Path, String Name)
```

DoPresort

Required.

Launches the presort operation for all records previously submitted via the AddRecord method according to any settings specified prior to calling this method.

DoPresort first checks to see if your mailing meets the minimum requirements for the selected mail class and sortation.

After a successful call to this method, use the GetRecord, GetFirstRecord, and GetNextRecord methods to retrieve the presort data for each address record.

Return: Boolean Value.

If a True value is returned, the operation was sucessful.

If a False value is returned, an error occurred. Call GetParametersErrorString() to determine the cause.

Syntax

```
BooleanValue = object->DoPresort()
```

C

```
int = mdPresortDoPresort(object)
```

COM

```
BooleanValue = object.DoPresort()
```

SetPresortSettings

Required.

Input: Enumerated Value.

Sets the sortation options to use, including Mail Class, mail piece type, and automation settings.

If this method is not called, an error will occur when you call DoPresort.

Return: String Value.

Returns a list of settings being used, separated by a forward-slash character.

For languages that do not support named enumerations, use the long integer value shown. The possible values are listed below.

First-Class Only:

Value	Enumerated Value	Description
1	FCM_LTR_AUTO_NONAUTO	First Class Mail, Letter, Automation & Non-Automation Sorts
2	FCM_LTR_AUTO	First Class Mail, Letter, Automation Sortations Only
3	FCM_LTR_NONAUTO	First Class Mail, Letter, Non-automation Sortation Only
4	FCM_LTR_NONMACH	First Class Mail, Letter, Non-machinable
41	FCM_POSTCARD_AUTO_NONAUTO	First Class Mail, Post Card, Automation & Non-Automation Sorts
42	FCM_POSTCARD_AUTO	First Class Mail, Post Card, Automation Sortations Only
43	FCM_POSTCARD_NONAUTO	First Class Mail, Post Card, Non-automation Sortation Only
51	FCM_FLAT_COTray	First Class Mail, Flat, Automation and Non-automation sorts, Co-traying enabled.
52	FCM_FLAT_AUTO	First Class Mail, Flat, Automation-eligible pieces only.
53	FCM_FLAT_NONAUTO	First Class Mail, Flat, Non-automation pieces only.
54	FCM_FLAT_DISABLE_COTRAY	First Class Mail, Flat, Automation and Non-automation pieces, Co-traying disabled.
55	FCM_FLAT_COTRAY_FSM1000	First Class Mail, Flat, Automation and Non-automation pieces, Co-traying enabled, Use FSM1000 sorting hardware.
56	FCM_FLAT_Disable_COTRAY _FSM1000	First Class Mail, Flat, Automation and Non-automation pieces, Co-sacking disabled, Use FSM1000 sorting hardware.
57	FCM_FLAT_AUTO_FSM1000	First Class Mail, Flat, Automation pieces, Use FSM1000 sorting hardware.
58	FCM_FLAT_NONAUTO_FSM1000	First Class Mail, Flat, Non-automation pieces, Use FSM1000 sorting hardware.

Standard Only:

Value	Enumerated Value	Description
101	STD_LTR_ECRRT_AUTO_NONAUTO	Standard Mail, Letter, Enhanced Carrier Route, Automation and Non-automation pieces.
102	STD_LTR_ECRRT_NONAUTO	Standard Mail, Letter, Enhanced Carrier Route, Non-automation pieces.
103	STD_LTR_AUTO_NONAUTO	Standard Mail, Letter, Automation and Non-automation pieces.
104	STD_LTR_AUTO	Standard Mail, Letter, Automation pieces.
105	STD_LTR_NONAUTO	Standard Mail, Letter, Non-automation pieces.
106	STD_LTR_NONMACH	Standard Mail, Letter, Non-machinable pieces.
107	STD_LTR_ECRRT_NONMACH	Standard Mail, Letter, Enhanced Carrier Route, Non-machinable pieces.
108	STD_LTR_ECRRT_AUTO	Standard Mail, Letter, Enhanced Carrier Route, Automation
109	STD_LTR_ECRRT	Standard Mail, Letter, Enhanced Carrier Route, Automation and Non-automation pieces
151	STD_FLAT_ECRRT_COSACK	Standard Mail, Flat, Enhanced Carrier Route, Automation and Non-automation pieces, Co-sacking enabled.
152	STD_FLAT_ECRRT_DISABLE_COSACK	Standard Mail, Flat, Enhanced Carrier Route pieces only, Co-sacking disabled.
153	STD_FLAT_COSACK	Standard Mail, Flat, Automation and Non-automation pieces, Co-sacking enabled.
154	STD_FLAT_DISABLE_COSACK	Standard Mail, Flat, Automation and Non-automation pieces, Co-sacking disabled.
155	STD_FLAT_AUTO	Standard Mail, Flat, Automation pieces.
156	STD_FLAT_NONAUTO	Standard Mail, Flats, Non-automation pieces
157	STD_FLAT_ECRRT_AUTO	Standard Mail, Flat, Enhanced Carrier Route Automation pieces only.
158	STD_FLAT_ECRRT_NONAUTO	Standard Mail, Flats, Enhanced Carrier Route, Non-automation pieces.
159	STD_FLAT_ECRRT	Standard Mail, Flats, Enhanced Carrier Route, Automation and Non-automation pieces.

Syntax

```
StringValue = object->SetPresortSettings
   (enum SortationCode)
```

C

```
StringValue = mdPresortSetPresortSettings(object, int)
```

COM

```
StringValue = object.PresortSettings(enum SortationCode)
```

UpdateParameters

Required.

Validates the mail piece dimension properties, verifying that they are within the correct ranges for the options selected using PresortSettings, before calling AddRecord or DoPresort.

The following properties are checked to be within the correct ranges:

SetPieceHeight, SetPieceLength, SetPieceThickness, and SetPieceWeight.

Valid ranges for these properties can change depending on the mail class and piece type selected. Call this method after calling the above properties and PresortSettings, before making any additional calls to the AddRecord or DoPresort methods.

Return: Boolean Value.

If a **False** value is returned, all values are inside the valid ranges for the selected mail class.

If a **True** value is returned, an error occurred. Call GetParametersErrorString() to determine the cause.

Syntax

```
BooleanValue = object->UpdateParameters()
```

C

```
int = mdPresortUpdateParameters(object)
```

COM

```
BooleanValue = object.UpdateParameters()
```

AddRecord

Required.

Adds a new record to the current list.

AddRecord requires SetRecordID and SetZip be set. Make sure to call SetPlus4 to receive the largest possible postage discount.

Repeat this method for each address record on your list.

Return: Boolean Value.

If a True value is returned, this method was successful.

If a False value is returned, an error occurred. Call GetParametersErrorString() to determine the cause.

Syntax
`BooleanValue = object->AddRecord()`
C
`int = mdPresortAddRecord(object)`
COM
`BooleanValue = object.AddRecord()`

GetRecord

Optional.

Input: String Value.

Retrieves a record from the presorted list based on the record ID submitted to the method. Matches the contents of SetRecordID property passed to the AddRecord method. Populates the result property after a successful call to DoPresort.

This is an alternate way to retrieve the results of a call to DoPresort. You can retrieve the record ID from your database, passing that information to this method and retrieve a specific record. This allows you to retrieve the presorted records in the order in which they appear in the original database and write this information back. You can then later use database queries to retrieve the data, sorting according to the value of the GetSequenceNumber property.

A successful call to this method populates the following properties:

GetRecordID	GetSequenceNumber	GetTrayNumber
GetEndorsementLine	GetBundleNumber	GetServiceTypeID
GetBarcodeID		

Return: Boolean Value.

If a True value is returned, the properties were successfully populated.

If a False value is returned, an error occurred. The record ID passed in does not match a record ID in the current presorted list or DoPresort has not been called.

Syntax

```
BooleanValue = object->GetRecord(StringValue)
```

C

```
int = mdPresortGetRecord(object, *char)
```

COM

```
BooleanValue = object.GetRecord(StringValue)
```

GetFirstRecord

Optional.

Retrieves the first record in the presorted list and populates the return values of properties described below with the presort information related to that record.

It cannot be called until a successful call has been made to the DoPresort method. Use GetFirstRecord and GetNextRecord to retrieve the results in the order created by a call to DoPresort. This is useful when writing the presorted records to a new database or file, rather than to the original database or file.

A successful call to this method populates the following properties with values for the first mail piece in the presorted list:

GetRecordID	GetSequenceNumber	GetTrayNumber
GetEndorsementLine	GetBundleNumber	GetServiceTypeID
GetBarcodeID		

Return: Boolean Value.

If a True value is returned, the call was successful and the first record was retrieved.

If a False value is returned, an error occurred. The ProduceReports method has not been called.

Syntax

```
BooleanValue = object->GetFirstRecord()
```

C

```
int = mdPresortGetFirstRecord(object)
```

COM

```
BooleanValue = object.GetFirstRecord()
```

GetNextRecord

Optional.

Retrieves the next record in the presorted list and populates the related properties with presort information related to that record.

It cannot be called until DoPresort and GetFirstRecord have been called. An error will occur if the last record in the presorted list has already been retrieved by a previous call to GetFirstRecord or GetNextRecord.

A successful call to this method populates the following properties:

GetRecordID	GetSequenceNumber	GetTrayNumber
GetEndorsementLine	GetBundleNumber	GetServiceTypeID
GetBarcodeID		

Return: Boolean Value.

If a True value is returned, the record was retrieved and the properties were successfully populated.

If a False value is returned, there are no more records in the presorted list.

Syntax

```
BooleanValue = object->GetNextRecord()
```

C

```
int = mdPresortGetNextRecord(object)
```

COM

```
BooleanValue = object.GetNextRecord()
```

SetProduceIMBCode

Optional.

Input: Boolean Value.

If set to True, Intelligent Mail barcodes will be included with the presort operation.

Do not call this method if you do not want to include Intelligent Mail barcodes.

In order to successfully return an IMB, you must input a ZIP, Plus4 and Delivery Point Code.

You must install the usps4cb.ttf font to use Intelligent Mail barcodes. This font is located in the 'Extras' folder on the install disk. When printing the barcodes in this font, use 16pt as the font size.

Syntax

```
void = object->SetProduceIMBCode(BooleanValue)
```

C

```
void = mdPresortSetProduceIMBCode(object, int)
```

COM

```
object.ProduceIMBCode = BooleanValue
```

SetACSCodeSettings

Optional.

Input: Enumerated Value.

Sets the value for STID depending on what type of move update method is used.

Return: Boolean Value.

Syntax

```
stringValue = object->SetACSCodeSettings(enum ACSCode)
```

C

```
stringValue = mdPresortSetACSCodeSettings(object, int)
```

COM

```
stringValue = object.ACSCodeSettings(enum ACSCode)
```

First-Class Only:

Type	Move Update Method	IMb Trace	Name	Enum
Basic	ACS	OFF	FCM_ACS_ASR	1
Basic	ACS	OFF	FCM_ACS_ASR2	2
Basic	ACS	OFF	FCM_ACS_CSR	3
Basic	ACS	OFF	FCM_ACS_CSR2	4
Basic	ONCODE ACS	OFF	FCM_ACS_ASR_ONECODE	5
Basic	ONCODE ACS	OFF	FCM_ACS_ASR2_ONECODE	6
Basic	ONCODE ACS	OFF	FCM_ACS_CSR_ONECODE	7
Basic	ONCODE ACS	OFF	FCM_ACS_CSR2_ONECODE	8
Basic	ACS	ON	FCM_ACS_ASR_TRACE	11
Basic	ACS	ON	FCM_ACS_ASR2_TRACE	12
Basic	ACS	ON	FCM_ACS_CSR_TRACE	13
Basic	ACS	ON	FCM_ACS_CSR2_TRACE	14
Basic	ONCODE ACS	ON	FCM_ACS_ASR_ONECODE_TRACE	15
Basic	ONCODE ACS	ON	FCM_ACS_CSR_ONECODE_TRACE	16
Full Service	ACS	OFF	FCM_ACS_ASR_FS	19
Full Service	ACS	OFF	FCM_ACS_ASR2_FS	20
Full Service	ACS	OFF	FCM_ACS_CSR_FS	21
Full Service	ACS	OFF	FCM_ACS_CSR2_FS	22
Full Service	ACS	OFF	FCM_FULL_SERVICE_ACS_ASR_FS	23
Full Service	ACS	OFF	FCM_FULL_SERVICE_ACS_ASR2_FS	24
Full Service	ACS	ON	FCM_ACS_ASR_FS_TRACE	27
Full Service	ACS	ON	FCM_ACS_ASR2_FS_TRACE	28
Full Service	ACS	ON	FCM_ACS_CSR_FS_TRACE	29
Full Service	ACS	ON	FCM_ACS_CSR2_FS_TRACE	30
Full Service	ACS	ON	FCM_FULL_SERVICE_ACS_ASR_FS_TRACE	31
Full Service	ACS	ON	FCM_FULL_SERVICE_ACS_ASR2_FS_TRACE	32
Basic	ONCODE ACS	ON	FCM_ACS_ASR2_ONECODE_TRACE	35
Basic	ONCODE ACS	ON	FCM_ACS_CSR2_ONECODE_TRACE	36
Full Service	ACS	OFF	FCM_FULL_SERVICE_ACS_CSR_FS	37
Full Service	ACS	OFF	FCM_FULL_SERVICE_ACS_CSR2_FS	38
Full Service	ACS	ON	FCM_FULL_SERVICE_ACS_CSR_FS_TRACE	39

Type	Move Update Method	IMb Trace	Name	Enum
Full Service	ACS	ON	FCM_FULL_SERVICE_ACS_CSR2_FS_TRACE	40
Basic	ACS	OFF	FCM_ACS_RSR2	41
Basic	ACS	OFF	FCM_ACS_TRSR2	42
Basic	ONCODE ACS	OFF	FCM_ACS_RSR2_ONECODE	43
Basic	ONCODE ACS	OFF	FCM_ACS_TRSR2_ONECODE	44
Basic	ACS	ON	FCM_ACS_RSR2_TRACE	45
Basic	ACS	ON	FCM_ACS_TRSR2_TRACE	46
Basic	ONCODE ACS	ON	FCM_ACS_RSR2_ONECODE_TRACE	47
Basic	ONCODE ACS	ON	FCM_ACS_TRSR2_ONECODE_TRACE	48
Full Service	ACS	OFF	FCM_ACS_RSR2_FS	49
Full Service	ACS	OFF	FCM_ACS_TRSR2_FS	50
Full Service	ACS	OFF	FCM_FULL_SERVICE_ACS_RSR2_FS	51
Full Service	ACS	OFF	FCM_FULL_SERVICE_ACS_TRSR2_FS	52
Full Service	ACS	ON	FCM_ACS_RSR2_FS_TRACE	53
Full Service	ACS	ON	FCM_ACS_TRSR2_FS_TRACE	54
Full Service	ACS	ON	FCM_FULL_SERVICE_ACS_RSR2_FS_TRACE	55
Full Service	ACS	ON	FCM_FULL_SERVICE_ACS_TRSR2_FS_TRACE	56
Basic	Alt Method	OFF	FCM_MANUAL_CORRECTION	9
Full Service	Alt Method	OFF	FCM_MANUAL_CORRECTION_FS	25
Full Service	Alt Method	ON	FCM_MANUAL_CORRECTION_FS_TRACE	33
Basic	Alt Method	ON	FCM_MANUAL_CORRECTION_TRACE	17
Basic	Ancillary Service	OFF	FCM_MANUAL_CORRECTION	9
Full Service	Ancillary Service	OFF	FCM_MANUAL_CORRECTION_FS	25
Full Service	Ancillary Service	ON	FCM_MANUAL_CORRECTION_FS_TRACE	33
Basic	Ancillary Service	ON	FCM_MANUAL_CORRECTION_TRACE	17
Basic	Mulitple	OFF	FCM_MANUAL_CORRECTION	9
Full Service	Mulitple	OFF	FCM_MANUAL_CORRECTION_FS	25
Full Service	Mulitple	ON	FCM_MANUAL_CORRECTION_FS_TRACE	33
Basic	Mulitple	ON	FCM_MANUAL_CORRECTION_TRACE	17

Type	Move Update Method	IMb Trace	Name	Enum
Basic	N/A Alternative Address	OFF	FCM_MANUAL_CORRECTION	9
Full Service	N/A Alternative Address	OFF	FCM_MANUAL_CORRECTION_FS	25
Full Service	N/A Alternative Address	ON	FCM_MANUAL_CORRECTION_FS_TRACE	33
Basic	N/A Alternative Address	ON	FCM_MANUAL_CORRECTION_TRACE	17
Basic	N/A Alternative Address	ON	FCM_MANUAL_CORRECTION_TRACE	17
Basic	NCOA Link	OFF	FCM_MANUAL_CORRECTION	9
Full Service	NCOA Link	OFF	FCM_MANUAL_CORRECTION_FS	25
Full Service	NCOA Link	ON	FCM_MANUAL_CORRECTION_FS_TRACE	33
Basic	NCOA Link	ON	FCM_MANUAL_CORRECTION_TRACE	17
Full Service	None	OFF	FCM_NO_ADDR_CORRECTION_FS	26
Full Service	None	ON	FCM_NO_ADDR_CORRECTION_FS_TRACE	34
Basic	None	ON	FCM_NO_ADDR_CORRECTION_TRACE	18
Basic	None	OFF	FCM_NO_ADDR_CORRECTION	10

Standard Only:

Type	Move Update Method	IMb Trace	Name	Enum
Basic	ACS	OFF	STD_ACS_ASR	101
Basic	ACS	OFF	STD_ACS_CSR	102
Basic	ACS	OFF	STD_ACS_ASR2	125
Basic	ACS	OFF	STD_ACS_RSR2	126
Basic	ACS	ON	STD_ACS_ASR_TRACE	107
Basic	ACS	ON	STD_ACS_CSR_TRACE	108
Basic	ACS	ON	STD_ACS_ASR2_TRACE	129
Basic	ACS	ON	STD_ACS_RSR2_TRACE	130
Full Service	ACS	OFF	STD_ACS_ASR_FS	113
Full Service	ACS	OFF	STD_ACS_CSR_FS	114
Full Service	ACS	OFF	STD_FULL_SERVICE_ACS_ASR_FS	115
Full Service	ACS	OFF	STD_FULL_SERVICE_ACS_CSR_FS	116
Full Service	ACS	OFF	STD_ACS_ASR2_FS	133
Full Service	ACS	OFF	STD_ACS_RSR2_FS	134

Type	Move Update Method	IMb Trace	Name	Enum
Full Service	ACS	OFF	STD_FULL_SERVICE_ACS_ASR2_FS	135
Full Service	ACS	OFF	STD_FULL_SERVICE_ACS_RSR2_FS	136
Full Service	ACS	ON	STD_ACS_ASR_FS_TRACE	119
Full Service	ACS	ON	STD_ACS_CSR_FS_TRACE	120
Full Service	ACS	ON	STD_FULL_SERVICE_ACS_ASR_FS_TRACE	121
Full Service	ACS	ON	STD_FULL_SERVICE_ACS_CSR_FS_TRACE	122
Full Service	ACS	ON	STD_ACS_ASR2_FS_TRACE	137
Full Service	ACS	ON	STD_ACS_RSR2_FS_TRACE	138
Full Service	ACS	ON	STD_FULL_SERVICE_ACS_ASR2_FS_TRACE	139
Full Service	ACS	ON	STD_FULL_SERVICE_ACS_RSR2_FS_TRACE	140
Basic	Alt Method	OFF	STD_MANUAL_CORRECTION	105
Basic	Alt Method	ON	STD_MANUAL_CORRECTION_TRACE	111
Full Service	Alt Method	OFF	STD_MANUAL_CORRECTION_FS	117
Full Service	Alt Method	ON	STD_MANUAL_CORRECTION_FS_TRACE	123
Basic	Ancillary Service	OFF	STD_MANUAL_CORRECTION	105
Basic	Ancillary Service	ON	STD_MANUAL_CORRECTION_TRACE	111
Full Service	Ancillary Service	OFF	STD_MANUAL_CORRECTION_FS	117
Full Service	Ancillary Service	ON	STD_MANUAL_CORRECTION_FS_TRACE	123
Basic	Mulitple	OFF	STD_MANUAL_CORRECTION	105
Basic	Mulitple	ON	STD_MANUAL_CORRECTION_TRACE	111
Full Service	Mulitple	OFF	STD_MANUAL_CORRECTION_FS	117
Full Service	Mulitple	ON	STD_MANUAL_CORRECTION_FS_TRACE	123
Basic	N/A Alternative Address	OFF	STD_MANUAL_CORRECTION	105
Basic	N/A Alternative Address	OFF	STD_MANUAL_CORRECTION	105
Basic	N/A Alternative Address	ON	STD_MANUAL_CORRECTION_TRACE	111
Full Service	N/A Alternative Address	OFF	STD_MANUAL_CORRECTION_FS	117

Type	Move Update Method	IMb Trace	Name	Enum
Full Service	N/A Alternative Address	ON	STD_MANUAL_CORRECTION_FS_TRACE	123
Basic	NCOA Link	ON	STD_MANUAL_CORRECTION_TRACE	111
Full Service	NCOA Link	OFF	STD_MANUAL_CORRECTION_FS	117
Full Service	NCOA Link	ON	STD_MANUAL_CORRECTION_FS_TRACE	123
Basic	None	OFF	STD_NO_ADDR_CORRECTION	106
Basic	None	ON	STD_NO_ADDR_CORRECTION_TRACE	112
Full Service	None	OFF	STD_NO_ADDR_CORRECTION_FS	118
Full Service	None	ON	STD_NO_ADDR_CORRECTION_FS_TRACE	124
Basic	ONCODE ACS	OFF	STD_ACS_ASR_ONECODE	103
Basic	ONCODE ACS	OFF	STD_ACS_CSR_ONECODE	104
Basic	ONCODE ACS	OFF	STD_ACS_ASR2_ONECODE	127
Basic	ONCODE ACS	OFF	STD_ACS_RSR2_ONECODE	128
Basic	ONCODE ACS	ON	STD_ACS_ASR_ONECODE_TRACE	109
Basic	ONCODE ACS	ON	STD_ACS_CSR_ONECODE_TRACE	110
Basic	ONCODE ACS	ON	STD_ACS_ASR2_ONECODE_TRACE	131
Basic	ONCODE ACS	ON	STD_ACS_RSR2_ONECODE_TRACE	132

Output Properties

All of the following properties are populated by a call to the GetRecord, GetFirstRecord, or GetNext Record methods after a successful call to the DoPresort method.

GetBarcodeID

Optional.

Return: String Value.

Returns information needed to generate the Intelligent Mail barcode for the current mail piece.

Syntax

```
StringValue = object->GetBarcodeID()
```

C

```
StringValue = mdPresortGetBarcodeID(object)
```

COM

```
StringValue = object.BarcodeID
```

GetBundleNumber

Optional.

Return: Long Integer.

Returns the package, bundle or group number of the current record. Use this number for sorting your list before printing address labels.

Syntax

```
LongInteger = object->GetBundleNum()
```

C

```
long int = mdPresortGetBundleNum(object)
```

COM

```
LongInteger = object.BundleNum
```

GetBundleZipCode

Optional.

Return: String Value.

Returns the destination 3 or 5-digit ZIP Code for the bundle containing the current record.

Syntax

```
StringValue = object->GetBundleZipCode()
```

C

```
StringValue = mdPresortGetBundleZipCode(object)
```

COM

```
StringValue = object.BundleZipCode
```

GetCINCode

Optional.

Return: String Value.

Returns the 3-digit Content Identifier Number (CIN) Code for the current record.

DMM Reference

708.6.1.4

Syntax

```
StringValue = object->GetCINCode()
```

C

```
StringValue = mdPresortGetCINCode(object)
```

COM

```
StringValue = object.CINCode
```

GetEndorsementLine

Optional.

Return: String Value.

Returns the Endorsement Line sort type code for the current record.

This value is usually included as the first line of the label, providing a shorthand for the ZIP Code, carrier route and sortation for the current piece. This code also helps match the label with the Qualification Report.

Syntax

```
StringValue = object->GetEndorsementLine()
```

C

```
StringValue = mdPresortGetEndorsementLine(object)
```

COM

```
StringValue = object.EndorsementLine
```

GetIMBAlphaCode

Optional.

Return: String Value.

Returns an alphabetic representation of the Intelligent Mail barcode.

You must have SetProduceIMBCode to True for this property to return a value.

In order to successfully return an IMB, you must input a ZIP, Plus4 and Delivery Point Code.

For more information see “Intelligent Mail barcode” on page 106.

Syntax

```
StringValue = object->GetIMBAlphaCode()
```

C

```
StringValue = mdPresortGetIMBAlphaCode(object)
```

COM

```
StringValue = object.IMBAlphaCode
```

GetIMBNumericCode

Optional.

Return: String Value.

Returns a numeric representation of the Intelligent Mail barcode.

You must have SetProduceIMBCode to True for this property to return a value.

In order to successfully return an IMB, you must input a ZIP, Plus4 and Delivery Point Code.

For more information see "Intelligent Mail barcode" on page 106.

Syntax

```
StringValue = object->GetIMBNumericCode()
```

C

```
StringValue = mdPresortGetIMBNumericCode(object)
```

COM

```
StringValue = object.IMBNumericCode
```

GetIMBSerialNumber

Optional.

Return: All Digits String Value.

Returns the Intelligent Mail barcode number.

You must have assigned a value to SetIMBSerialNumber for this property to return a value.

This allows you to retrieve the distinct number for each mail piece in the IMB.

For more information see "Intelligent Mail barcode" on page 106.

Syntax

```
void = object->GetIMBSerialNumber(AllDigitsStringValue)
```

C

```
void = mdPresortGetIMBSerialNumber(AllDigitsStringValue)
```

COM

```
object.GetIMBSerialNumber = AllDigitsStringValue
```

GetRecordID

Optional.

Return: String Value.

Returns a unique value passed to the SetRecordID property before calling the DoPresort method. This links each record in the presorted list back to the corresponding address record in the original database.

Syntax

```
StringValue = object->GetRecordID()
```

C

```
StringValue = mdPresortGetRecordID(object)
```

COM

```
StringValue = object.RecordID
```

GetSequenceNumber

Optional.

Return: Long Integer.

Returns the number which indicates the exact position of the current record when the list is sorted in presort order.

Syntax

```
LongInteger = object->GetSequenceNumber()
```

C

```
long int = mdPresortGetSequenceNumber(object)
```

COM

```
LongInteger = object.SequenceNumber
```

GetServiceTypeID

Optional.

Return: String Value.

Returns the service type ID which is part of the intelligent mail barcode.

Syntax
`StringValue = object->GetServiceTypeID()`
C
`StringValue = mdPresortGetServiceTypeID(object)`
COM
`StringValue = object.ServiceTypeID`

GetTrayNumber

Optional.

Return: Long Integer.

Returns the container number for the current record.

Syntax
`LongInteger = object->GetTrayNum()`
C
`long int = mdPresortGetTrayNum(object)`
COM
`LongInteger = object.TrayNum`

GetTrayZipCode

Optional.

Return: String Value.

Returns the destination facility ZIP Code for the container of current record.

The TrayZIPCode is the 3- or 5-digit ZIP Code destination for the current container from the appropriate DMM Module L Labeling List or mail piece address depending upon the sortation level assigned.

Syntax

```
StringValue = object->GetTrayZipCode()
```

C

```
StringValue = mdPresortGetTrayZipCode(object)
```

COM

```
StringValue = object.TrayZipCode
```

GetZipAsString

Optional.

Return: String Value.

Returns the ZIP Code for the current record.

Syntax

```
StringValue = object->ZipAsString()
```

C

```
StringValue = mdPresortZipAsString(object)
```

COM

```
StringValue = object.ZipAsString
```

Mail.Dat Properties

Mail.dat is a database that contains industry standards for mailing, including presorting. This helps facilitate efficient communication among those providing mailing services. Required properties are only required if you will be using Mail.Dat.

ProduceMailDatFiles

Required.

Input: Data Path String Value, File Name String Value.

Produces the Mail.Dat files.

Syntax

```
BooleanValue = object->ProduceMailDatFiles
   (DataPathStringValue, FileNameStringValue)
```

C

```
int = mdPresortProduceMailDatFiles
   (object, DataPath*char, FileName*char)
```

COM

```
BooleanValue = object.ProduceMailDatFiles
   (DataPathStringValue, FileNameStringValue)
```

SetPSPostOfficeOfMailingPlus4

Required.

Input: String Value.

Sets the Post Office of Mailing Plus4 code.

Syntax

```
void = object->SetPSPostOfficeOfMailingPlus4(StringValue)
```

C

```
void = mdPresortSetPSPostOfficeOfMailingPlus4
   (object, *char)
```

COM

```
object.PSPostOfficeOfMailingPlus4 = StringValue
```

SetMDMachineID

Optional.

Input: String Value.

Sets the Machine ID of the machine that applied the barcode to the mail piece.

Syntax

```
void = object->SetMDMachineID(StringValue)
```

C

```
void = mdPresortSetMDMachineID(object, *char)
```

COM

```
object.MDMachineID = StringValue
```

SetMDJobID

Required.

Input: String Value.

Sets the Job ID, also known as the Mail.dat serial number. This value should be unique compared to all other Job IDs set.

Syntax

```
void = object->SetMDJobID(StringValue)
```

C

```
void = mdPresortSetMDJobID(object, *char)
```

COM

```
object.MDJobID = StringValue
```

SetMDHDRIDEAllianceVersion

Required.

Input: String Value.

Sets the IDEAlliance version number.

Syntax

```
void = object->SetMDHDRIDEAllianceVersion(StringValue)
```

C

```
void = mdPresortSetMDHDRIDEAllianceVersion(object, *char)
```

COM

```
object.MDHDRIDEAllianceVersion = StringValue
```

SetMDHDRLicensedUsersJobNumber

Optional.

Input: String Value.

Sets the licensed user's internal job number.

Syntax

```
void = object->SetMDHDRLicensedUsersJobNumber(StringValue)
```

C

```
void = mdPresortSetMDHDRLicensedUsersJobNumber
   (object, *char)
```

COM

```
object.MDHDRLicensedUsersJobNumber = StringValue
```

SetMDHDRJobNameTitleIssue

Required.

Input: String Value.

Sets any applicable job, title-issue, campaign name, or description.

Syntax
`void = object->SetMDHDRJobNameTitleIssue(StringValue)`
C
`void = mdPresortSetMDHDRJobNameTitleIssue(object, *char)`
COM
`object.MDHDRJobNameTitleIssue = StringValue`

SetMDHDRFileSource

Required.

Input: String Value.

Sets the name of the originator supplying the files.

Syntax
`void = object->SetMDHDRFileSource(StringValue)`
C
`void = mdPresortSetMDHDRFileSource(object, *char)`
COM
`object.MDHDRFileSource = StringValue`

SetMDHDRUserLicenseCode

Optional.

Input: String Value.

Sets the User License Code for Mail.dat. This code is given to you by IDE Alliance.

Syntax

```
void = object->SetMDHDRUserLicenseCode(StringValue)
```

C

```
void = mdPresortSetMDHDRUserLicenseCode(object, *char)
```

COM

```
object.MDHDRUserLicenseCode = StringValue
```

SetMDHDRContactEmail

Required.

Input: String Value.

Sets the email of an individual for support at the originator of this file.

Syntax

```
void = object->SetMDHDRContactEmail(StringValue)
```

C

```
void = mdPresortSetMDHDRContactEmail(object, *char)
```

COM

```
object.MDHDRContactEmail = StringValue
```

SetMDHDRContactName

Required.

Input: String Value.

Sets the name of an individual for support at the originator of this file.

Syntax
`void = object->SetMDHDRContactName(StringValue)`
C
`void = mdPresortSetMDHDRContactName(object, *char)`
COM
`object.MDHDRContactName = StringValue`

SetMDHDRContactPhone

Required.

Input: String Value.

Sets the phone number of the individual listed in MDHDRContactName.

Syntax
`void = object->SetMDHDRContactPhone(StringValue)`
C
`void = mdPresortSetMDHDRContactPhone(object, *char)`
COM
`object.MDHDRContactPhone = StringValue`

SetMDHDReDocSenderCRID

Optional.

Input: String Value.

Sets the CRID of the document sender. This is the USPS ID.

Syntax

```
void = object->SetMDHDReDocSenderCRID(StringValue)
```

C

```
void = mdPresortSetMDHDReDocSenderCRID(object, *char)
```

COM

```
object.MDHDReDocSenderCRID = StringValue
```

SetMDHDRMailDatSoftwareVendorName

Required.

Input: String Value.

Sets the name of the author of this Mail.dat software as appended to this record.

Syntax

```
void = object->SetMDHDRMailDatSoftwareVendorName
   (StringValue)
```

C

```
void = mdPresortSetMDHDRMailDatSoftwareVendorName(object,
*char)
```

COM

```
object.MDHDRMailDatSoftwareVendorName = StringValue
```

SetMDHDRMailDatSoftwareProductsName

Required.

Input: String Value.

Sets the name of the product creating the Header and applicable data in associated records.

Syntax

```
void = object->SetMDHDRMailDatSoftwareProductsName
   (StringValue)
```

C

```
void = mdPresortSetMDHDRMailDatSoftwareProductsName
   (object, *char)
```

COM

```
object.MDHDRMailDatSoftwareProductsName = StringValue
```

SetMDHDRMailDatSoftwareVersion

Required.

Input: String Value.

Sets the version of the software creating the transmitted Mail.dat.

Syntax

```
void = object->SetMDHDRMailDatSoftwareVersion(StringValue)
```

C

```
void = mdPresortSetMDHDRMailDatSoftwareVersion
   (object, *char)
```

COM

```
object.MDHDRMailDatSoftwareVersion = StringValue
```

SetMDHDRMailDatSoftwareVendorEmail

Required.

Input: String Value.

Sets the email address of the author of this software.

Syntax

```
void = object->SetMDHDRMailDatSoftwareVendorEmail
   (StringValue)
```

C

```
void = mdPresortSetMDHDRMailDatSoftwareVendorEmail
   (object, *char)
```

COM

```
object.MDHDRMailDatSoftwareVendorEmail = StringValue
```

SetMDSEGVerificationFacilityName

Optional.

Input: String Value.

Sets the name of the Mailing Facility where verification occurs.

Syntax

```
void = object->SetMDSEGVerificationFacilityName
   (StringValue)
```

C

```
void = mdPresortSetMDSEGVerificationFacilityName
   (object, *char)
```

COM

```
object.MDSEGVerificationFacilityName = StringValue
```

SetMDSEGVerificationFacilityZipPlus4

Optional.

Input: String Value.

Sets the ZIP + 4 of the Mailing Facility where verification occurs.

Syntax

```
void = object->SetMDSEGVerificationFacilityZipPlus4
   (StringValue)
```

C

```
void = mdPresortSetMDSEGVerificationFacilityZipPlus4
   (object, *char)
```

COM

```
object.MDSEGVerificationFacilityZipPlus4 = StringValue
```

SetMDSEGDescription

Required.

Input: String Value.

Sets the description of segmentation. The segmentation should be at single mail-stream level. Describe string, list, mail-stream characteristics which this particular set of names exhibits. For example:

Single List Segment: Spring - Re-mail, prospects, $100 Off

Selective Bind Segment: Spring - Re-mail, all versions

Syntax

```
void = object->SetMDSEGDescription(StringValue)
```

C

```
void = mdPresortSetMDSEGDescription(object, *char)
```

COM

```
object.MDSEGDescription = StringValue
```

SetMDMPUName

Required.

Input: String Value.

Sets the Mail Piece Unit Name. This name is used to identify a specific marketing version within a list, bind, and distribution environment.

Syntax

```
void = object->SetMDMPUName(StringValue)
```

C

```
void = mdPresortSetMDMPUName(object, *char)
```

COM

```
object.MDMPUName = StringValue
```

SetMDMPUDescription

Optional.

Input: String Value.

Sets the Mail Piece Unit Description. This is a unique name or code for each specific version being created within this mailing.

Syntax

```
void = object->SetMDMPUDescription(StringValue)
```

C

```
void = mdPresortSetMDMPUDescription(object, *char)
```

COM

```
object.MDMPUDescription = StringValue
```

SetMDMPADescription

Optional.

Input: String Value.

Sets the Mailer Postage Account Description.

Syntax
`void = object->SetMDMPADescription(StringValue)`
C
`void = mdPresortSetMDMPADescription(object, *char)`
COM
`object.MDMPADescription = StringValue`

SetMDMPAMailingAgentMailerID

Optional.

Input: String Value.

Sets the Mailer ID of the Mailing Agent.

Syntax
`void = object->SetMDMPAMailingAgentMailerID(StringValue)`
C
`void = mdPresortSetMDMPAMailingAgentMailerID(object, *char)`
COM
`object.MDMPAMailingAgentMailerID = StringValue`

SetMDCPTComDescription

Optional.

Input: String Value.

Sets the Component Description. This is a unique name or code for each specific portion of the mail piece. This property can carry an absolute reference to the component while the component ID is a shorthand for reference to the component's role within the mailing facilities' postage analysis. If used, it must have a value.

Syntax

```
void = object->SetMDCPTComDescription(StringValue)
```

C

```
void = mdPresortSetMDCPTComDescription(object, *char)
```

COM

```
object.MDCPTComDescription = StringValue
```

SetMDCPTMailOwnerID

Optional.

Input: String Value.

Sets the Mail Owner ID in the component record. This is the USPS ID.

Syntax

```
void = object->SetMDCPTMailOwnerID(StringValue)
```

C

```
void = mdPresortSetMDCPTMailOwnerID(object, *char)
```

COM

```
object.MDCPTMailOwnerID = StringValue
```

SetMDMPAMailOwnerPermitNumber

Optional.

Input: String Value.

Sets the Mail Owner Permit Number in the component record.

Syntax
`void = object->SetMDMPAMailOwnerPermitNumber(StringValue)`
C
`void = mdPresortSetMDMPAMailOwnerPermitNumber(object, *char)`
COM
`object.MDMPAMailOwnerPermitNumber = StringValue`

SetMDMPAMailOwnerPermitType

Optional.

Input: String Value.

Sets the Mail Owner Permit Type in the component record.

Syntax
`void = object->SetMDMPAMailOwnerPermitType(StringValue)`
C
`void = mdPresortSetMDMPAMailOwnerPermitType(object, *char)`
COM
`object.MDMPAMailOwnerPermitType = StringValue`

SetMDCPTOwnerCRID

Optional.

Input: String Value.

Sets the CRID of the Mail Owner in the component record. This is the USPS ID.

Syntax

```
void = object->SetMDCPTOwnerCRID(StringValue)
```

C

```
void = mdPresortSetMDCPTOwnerCRID(object, *char)
```

COM

```
object.MDCPTOwnerCRID = StringValue
```

SetMDCPTMailOwnersMailingRefID

Optional.

Input: String Value.

Sets the Mail Owner's Mailing Reference ID. This is the Mail Owner's chosen value to represent mailing to the USPS.

Syntax

```
void = object->SetMDCPTMailOwnersMailingRefID(StringValue)
```

C

```
void = mdPresortSetMDCPTMailOwnersMailingRefID
   (object, *char)
```

COM

```
object.MDCPTMailOwnersMailingRefID = StringValue
```

SetMDCPTPostalPriceIncID

Optional.

Input: String Value.

Sets the Postal Price Incentive ID. This is a USPS provided incentive ID.

Syntax

```
void = object->SetMDCPTPostalPriceIncID(StringValue)
```

C

```
void = mdPresortSetMDCPTPostalPriceIncID(object, *char)
```

COM

```
object.MDCPTPostalPriceIncID = StringValue
```

SetMDCPTPostalPriceIncType

Optional.

Input: String Value.

Sets the Postal Price Incentive type. The Postal Price Incentive types are:

Value	Description
A	Reply Rides Free
B	Saturation/ HD Total
C	Saturation/ HD SCF
D	Summer Sale
MB	Mobile Barcode

Syntax

```
void = object->SetMDCPTPostalPriceIncType(StringValue)
```

C

```
void = mdPresortSetMDCPTPostalPriceIncType(object, *char)
```

COM

```
object.MDCPTPostalPriceIncType = StringValue
```

SetMDCPTContentOfMail

Optional.

Input: String Value.

Sets the Content of the Mail. This identifies unique products within the mailing.

Value	Description
A	Reply Envelope or Reply Card
B	Contents NOT required to be mailed FCM
C	DVD/CD or other disk
E	Product Sample
F	Round Trip ONLY: One DVD/CD or other disk (Can be LT or FL)

For First-Class Mail, any possible combination of A, B, and C includes:

Value	Description
A1	A and B
A2	A and C
A3	A, B, and C
B1	B and C

For Standard Mail, any possible combination of C and E includes:

Value	Description
C1	C and E

Syntax

```
void = object->SetMDCPTContentOfMail(StringValue)
```

C

```
void = mdPresortSetMDCPTContentOfMail(object, *char)
```

COM

```
object.MDCPTContentOfMail = StringValue
```

SetMDCPTStandParcelType

Optional.

Input: String Value.

Sets the Standard Parcel type. The Standard Parcel types are:

Value	Description
M	Marketing
F	Fulfillment

Syntax	
	`void = object->SetMDCPTStandParcelType(StringValue)`
C	
	`void = mdPresortSetMDCPTStandParcelType(object, *char)`
COM	
	`object.MDCPTStandParcelType = StringValue`

SetMDCPTStandFlatType

Optional.

Input: String Value.

Sets the Standard Flat type. The Standard Flat types are:

Value	Description
C	Catalog
N	Not a Catalog

Syntax	
	`void = object->SetMDCPTStandFlatType(StringValue)`
C	
	`void = mdPresortSetMDCPTStandFlatType(object, *char)`
COM	
	`object.MDCPTStandFlatType = StringValue`

SetMDCPTUserOptField

Optional.

Input: String Value.

Sets the User Option field. This is not to be used for communication between parties. It is specifically for user usage.

Syntax
`void = object->SetMDCPTUserOptField(StringValue)`
C
`void = mdPresortSetMDCPTUserOptField(object, *char)`
COM
`object.MDCPTUserOptField = StringValue`

SetMDCSMCSAAgreementID

Optional.

Input: String Value.

Sets the CSA Agreement ID generated by the USPS.

Syntax
`void = object->SetMDCSMCSAAgreementID(StringValue)`
C
`void = mdPresortSetMDCSMCSAAgreementID(object, *char)`
COM
`object.MDCSMCSAAgreementID = StringValue`

Palletization

SetProducePallets

Optional.

Input: Boolean Value.

Enables palletization. This method cannot be called when mailing Flat pieces with Cosack or Cotrayed disabled.

Syntax

```
BooleanValue = object->SetProducePallets()
```

C

```
int = mdPresortSetProductPallets(object)
```

COM

```
BooleanValue = object.ProducePallets()
```

GetPalletLabelLine1

Optional.

Input: String Value.

Returns the pallet label line 1.

Syntax

```
StringValue = object->GetPalletLabelLine1()
```

C

```
StringValue = mdPresortGetPalletLabelLine1(object)
```

COM

```
StringValue = object.PalletLabelLine1
```

GetPalletLabelLine2

Optional.

Input: String Value.

Returns the pallet label line 2.

Syntax
`StringValue = object->GetPalletLabelLine2()`
C
`StringValue = mdPresortGetPalletLabelLine2(object)`
COM
`StringValue = object.PalletLabelLine2`

GetPalletNumber

Optional.

Input: Long Integer.

Returns the pallet number.

Syntax
`LongInteger = object->GetPalletNumber()`
C
`long int = mdPresortGetPalletNumber(object)`
COM
`LongInteger = object.PalletNumber`

GetPalletSortLevel

Optional.

Input: String Value.

Returns the pallet sort level.

Syntax

```
StringValue = object->GetPalletSortLevel()
```

C

```
StringValue = mdPresortGetPalletSortLevel(object)
```

COM

```
StringValue = object.PalletSortLevel
```

GetPalletsTotal

Optional.

Input: Long Integer.

Returns the total number of pallets.

Syntax

```
LongInteger = object->GetPalletsTotal()
```

C

```
long int = mdPresortGetPalletsTotal(object)
```

COM

```
LongInteger = object.PalletsTotal
```

GetPalletZipCode

Optional.

Input: String Value.

Returns the pallet ZIP Code.

Syntax
`StringValue = object->GetPalletZipCode()`
C
`StringValue = mdPresortGetPalletZipCode(object)`
COM
`StringValue = object.PalletZip`

Order of Operations

The following is pseudo-code for coding a basic presort operation. You will find more information on the listed page number for each call.

Step 1: Create Object

Step 2: Set License string:

Step 3: Set Path To Presort Data Files:

Step 4: Initialize Data Files:

Step 5: Sort settings (Pass the Enumeration of the sort that you want to use):

Step 6: Set Length, Height, Weight, and Piece Thickness:

Step 7: Set the Postage Statement fields:

Set the Permit Holder Information:

Set the Move Update Method (Choose One of the following):

Step 8: Set the Produce reports path:

Step 9: Pass Input fields to the object depending on what sort you are trying to qualify for.

For Standard and First Class Non-Automation Sorts:

Pass the Zip Code to the Object.

For Standard and First Class Automation Sorts:

Pass the Zip Code and Plus4 to the Object.

For Standard Sorts with Enhanced Carrier Route, Saturation, and High-Density:

Pass the Zip Code, Plus4, Carrier Route, Walk Sequence, Delivery Point Code, and Business Residential Indicator to the Object.

The only time you can do this sort is if you buy a list in walk sequence order or have DES append it the walk sequence number to your list.

For Standard Sorts with Enhanced Carrier Route Basic:

Pass the Zip Code, Plus4, Carrier Route, Delivery Point Code, Lot Number, and Lot Order to the Object.

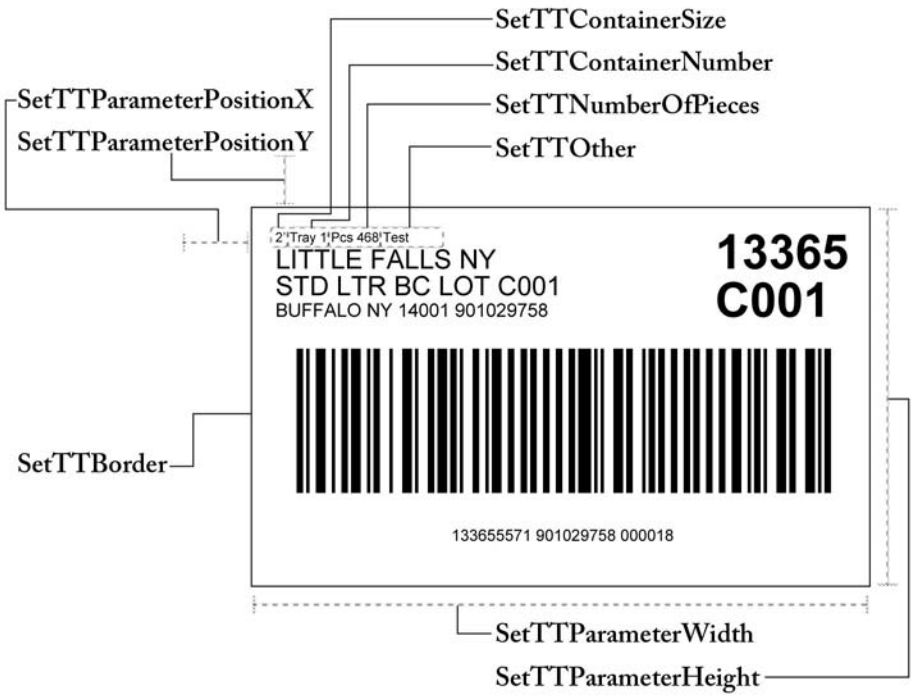
Step 10: Update the Parameters:

Step 11: Run the Presort:

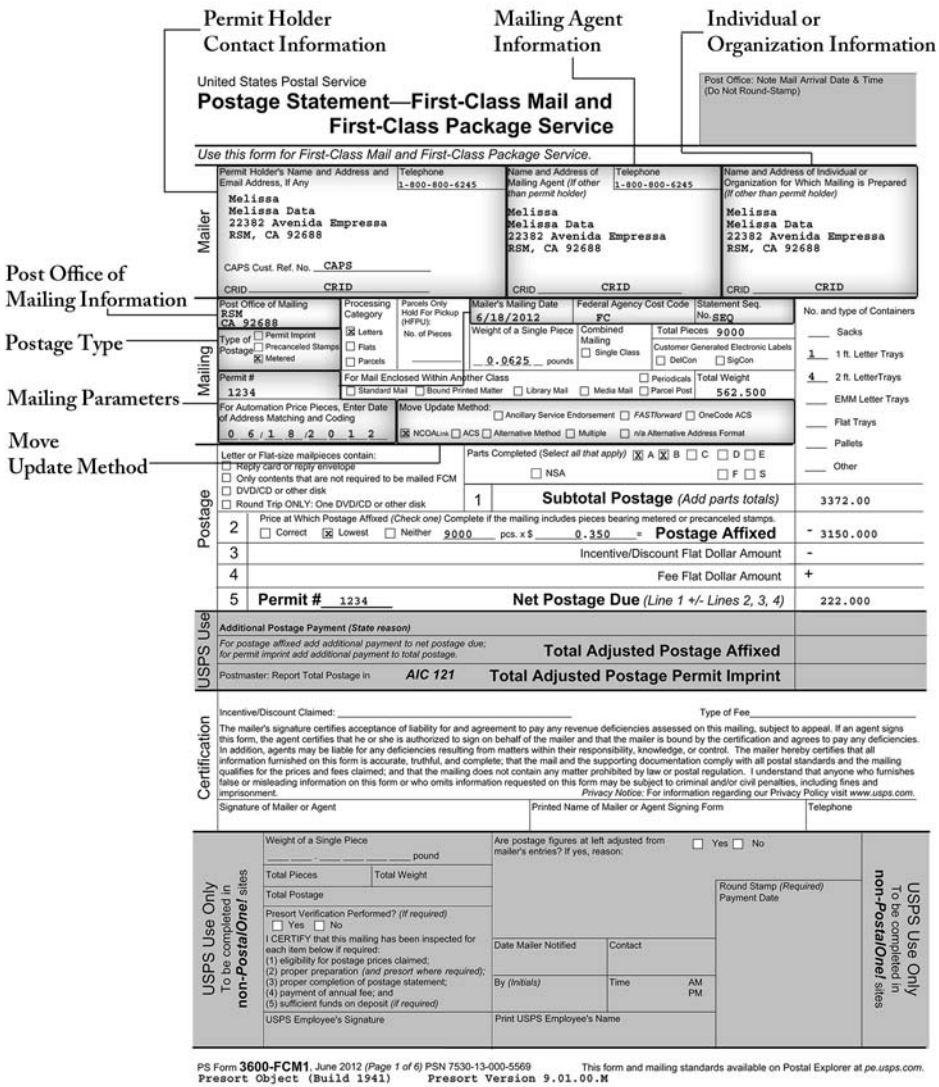
Step 12: Output the Tray Number, Sequence Number, Endorsement line, and Bundle number:

Sample Documents

Tray Tags



Postage Statement



Intelligent Mail barcode

The Intelligent Mail barcode can be returned either as a numeric or alpha string.

Numeric string example:

```
94702901029758000000055107193
```

Alpha string example:

```
TTDDFTAAAAAFFTAATDATFATTTTFAFTTATFTFDFTTDAFADAFTATDDAFTDFDTDFDTFD
```

When using the Intelligent Mail barcode alpha string with the required font (usps4cb.ttf) and required font-height (16pt), you will recieve a barcode like the following:

The following details what the Intelligent Mail barcode is composed of:

6-digit Mailer Identifier

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Barcode ID [2N]		Service Type ID [3N]			Mailer ID [6N]						Serial Number [9N]										Routing Code [None, 5, 9, or 11N] [ZIP + 4, Delivery Point]										

9-digit Mailer Identifier

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Barcode ID [2N]		Service Type ID [3N]			Mailer ID [9N]									Serial Number [6N]						Routing Code [None, 5, 9, or 11N] [ZIP + 4, Delivery Point]												

Presort Object will append all zeros for the Serial Number section of the barcode. These are used for Full Service only.