

e-gold[®]
www.e-gold.com

*A Programmers Guide to
Automating e-gold[®] Functions*

5 April, 2007

**Copyright 2000-2007 by e-gold[®] Ltd.
website: <http://www.e-gold.com>**

TABLE OF CONTENTS

1.	Introduction.....	1
1.1	Intended Audience	1
1.2	Overview of e-gold [®] Automation Tasks and Implementation.....	1
1.2.1	The e-gold [®] spend.	1
1.2.2	e-gold [®] account history.....	1
1.2.3	e-gold [®] account balance.....	1
1.2.4	e-gold [®] exchange rates.....	2
1.2.5	Implementation	2
2.	Automation Details.....	3
2.1	e-metal [®] spend	4
2.2	e-metal [®] spend preview/verification.....	6
2.3	e-metal [®] history.....	8
2.4	e-metal [®] balance	12
2.5	Exchange rates	14
3.	Example Scripts.....	17
3.1	Security notes	17
3.2	Disclaimer	18
3.3	Perl script (using cURL) to retrieve e-gold [®] account history	18
3.4	Perl script (using cURL) to preview e-gold [®] spend.....	22
3.5	Perl script (using cURL) to display e-gold [®] balance	25

e-gold[®] Programmers Guide

<u>Date</u>	<u>Description of Changes</u>	<u>Affected Sections</u>
April 5, 2007	Added two currencies to lists of currencies: XOF (CFA Franc BCEAO) and XAF (CFA Franc BEAC). Note trademark on e-metal [®]	2.1, 2.2, 2.3, 2.5,
September 23, 2004	Replaced contact email with URL.	Cover
December 23, 2001	Additional currencies for spend denominations including Euro.	2.1, 2.2, 2.3, 2.5
January 30, 2001	Correction of registered trademark symbol [®]	all
January 25, 2001	Inconsequential tweaks here and there.	
September 6, 2000	Initial document	all

e-gold[®] Programmers Guide

1. Introduction

This document describes how to programmatically interact with the e-gold[®] web site (located at <http://www.e-gold.com>). Interaction of this sort might be useful to those desiring to automate e-gold payments, verify a particular transaction has occurred, or pull account history into a third party program.

This document describes the fields and parameters required for interfacing to the e-gold system to perform these actions.

1.1 *Intended Audience*

This document is intended to be utilized by technical personnel with programming knowledge; specifically with a working knowledge of HTML forms.

1.2 *Overview of e-gold[®] Automation Tasks and Implementation*

1.2.1 The e-gold[®] spend.

The e-gold[®] spend is a transaction initiated by the owner of an e-gold[®] account to transfer e-metal[®] to another e-gold[®] account. (An e-gold[®] account holder can never “pull” e-metal[®] from another account; only the owner may initiate value transfer).

Normally, an e-gold[®] spend is performed interactively by the account holder using a web (or PCS/WAP phone) browser to submit payment information via HTML (HDML/WML) form input.

The user interested in automation of this task may require unattended spending, or repetitive spending without the manual data re-entry this would require using the browser method.

1.2.2 e-gold[®] account history.

The e-gold[®] account history for a particular account contains all the transactions for the account. (e.g. e-metal[®] payments received, made, storage fees assessed, etc).

Again, this act of viewing this history is normally performed interactively via a web browser.

The user interested in automation of this task may require verification of a particular transaction (possibly an automated spend, or receipt of payment via the e-gold shopping cart interface [SCI]).

1.2.3 e-gold[®] account balance.

The e-gold[®] account balance for a particular account is the current weight in ounces of each particular e-metal[®] held in the account.

The balance is normally viewed using a browser.

e-gold[®] Programmers Guide

The user interested in automation of balance may wish to display their balance or derived information on their web site.

1.2.4 e-gold[®] exchange rates.

Although e-gold Ltd. Does not make a market in exchange, it does maintain a set of exchange rates for denominating e-metal[®] spends in various currency amounts. (i.e. spend 1 USD worth of e-gold[®]).

Users normally view the exchange rates page with a web browser, or enter currency equivalents when performing an e-metal[®] spend not denominated in weight units.

The user interested in automation of this task may wish to perform calculations based on the current (or past) exchange rates used on the e-gold[®] system.

1.2.5 Implementation

The user wishing to implement e-gold[®] automation functionality has several options for implementing the task at hand. The language and platform chosen for the task must provide the following functionality:

1. Support for HTTPS form submission to a remote host.
2. Support for examining returned data for results.

Security Notes:

- 1. Do not embed your e-gold[®] account passphrase in your automation program. Rather, prompt the user for the passphrase when the program runs.**
- 2. Examine your automation program closely for problems. You are just as responsible for e-metal[®] transactions initiated by an automation program as if you had entered the transaction by hand. (i.e. e-metal[®] spends are not reversible regardless of the initiation method).**

2. Automation Details

This section provides detailed information to allow a programmer to interface his automation program to the e-gold[®] payment system.

In general, input fields are provided to the given URL via either an HTML form of the POST or GET variety. For instance a GET style request to retrieve the balance of e-metal[®] account 101574 might look like:

```
https://www.e-gold.com/acct/balance.asp?AccountID=101574&Passphrase=ThisIsNotReal
```

The history and exchange rate data is returned in comma delimited format to the requestor. The balance, spend, and preview URLs return result data in hidden fields embedded in normal HTML. For example, the above balance request might return:

```
...  
<td align=right><font face="Arial, Helvetica, sans-serif" size="2">225.7436</font></td>  
<input type=hidden name=Gold_Ounces value="7.257937">  
<input type=hidden name=Gold_Grams value="225.7436">  
<td align=right><font face="Arial, Helvetica, sans-serif" size="2">1,975.61</font></td>  
...
```

The caller can ignore all the HTML fields and pull out the return values from the hidden fields. (See the example scripts).

e-gold[®] Programmers Guide

2.1 e-metal[®] spend

An e-metal[®] spend may be performed by doing an HTML form post of GET or POST variety to the URL: <https://www.e-gold.com/acct/confirm.asp>

The following input fields should be used:

Input Fields for submit to https://www.e-gold.com/acct/confirm.asp		
Input Field Name	Description	Example Value
AccountID	e-metal [®] account number (payer)	100079
PassPhrase	e-metal [®] account passphrase	ThisIsNotReal
Payee_Account	e-metal [®] account to spend to	100997
Amount	Amount to be spent. Must be positive numerical amount. Also must equate to at least 0.000010 ounces at current exchange rates.	5.15
PAY_IN	Code defining units payment is to be denominated in. Must be one of 1, 2, 33, 41, 44, 49, 61, 81, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 8888, 9999 . 1=US Dollars, 2=Canadian Dollars, 33=French Francs, 41=Swiss Francs, 44=British Pounds, 49=Deutsche Marks, 61=Australian Dollars, 81=Japanese Yen, 85=Euro, 86=Belgian Franc, 87=Austrian Schilling, 88=Greek Drachma, 89=Spanish Peseta, 90=Irish Pound, 91=Italian Lira, 92=Luxembourg Franc, 93=Dutch Guilder, 94=Portuguese Escudo, 95=Finnish Markka, 96=Estonian Kroon, 97=Lithuanian Litas, 98=CFA Franc BEAC, 99=CFA Franc BCEAO, 8888=grams, 9999=ounces (troy).	1
WORTH_OF	e-metal [®] name to spend. Must be one of "Gold", "Silver", "Platinum", "Palladium" .	Gold
Memo	Up to 50 characters to be placed in memo section of transaction . The memo is visible to both payer and payee.	Thanks for dinner.
ACTUAL_PAYMENT_OUNCES	Numerical weight amount of payment as previewed. Formatted to 6 decimal places. Not required if IGNORE_RATE_CHANGE set. If present, must equate to computed weight of spend for spend to occur. (Used to detect updated exchange rates since spend was previewed).	0.017167
IGNORE_RATE_CHANGE	Spend will only occur if ACTUAL_PAYMENT_OUNCES input matches computed weight of spend. If IGNORE_RATE_CHANGE is present (value does not matter), spend will occur regardless of ACTUAL_PAYMENT_OUNCES input.	y
PAYMENT_ID	Optional merchant reference number. If present, this string of up to 50 characters is placed in the transaction. Payer and/or payee may search/query account history for this value.	ID 55789-Ab9@

e-gold[®] Programmers Guide

Output from the confirm.asp page will include HTML with embedded hidden form fields for retrieving results of the spend. These are:

<u>Output Fields from submit to https://www.e-gold.com/acct/confirm.asp</u>		
Output Field Name	Description	Example Value
ERROR	Spend did not occur if this field present. Text description of error.	The UNITS 555 specified is not valid.
Payee_Account	e-metal [®] account number of recipient of spend.	100997
ACTUAL_PAYMENT_OUNCES	Amount of e-metal [®] spent from spender. Formatted to 6 decimal places. (Note that the receiver of the e-metal [®] spend is charged a fee to receive the spend and will not net this complete amount. See PAYMENT_FEE_OUNCES field).	0.017167
PAYMENT_FEE_OUNCES	Fee charged to recipient of payment by e-gold Ltd in ounces of e-metal [®] . Formatted to 6 decimal places.	0.000172
PAYMENT_AMOUNT	Numerical amount of spend as entered. This is the same value as the Amount input field.	5.15
PAYMENT_UNITS	Code defining units payment was denominated in. Will be one of 1, 2, 33, 41, 44, 49, 61, 81, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 8888, 9999 and will match the PAY_IN input field. 1=US Dollars, 2=Canadian Dollars, 33=French Francs, 41=Swiss Francs, 44=British Pounds, 49=Deutsche Marks, 61=Australian Dollars, 81=Japanese Yen, 85=Euro, 86=Belgian Franc, 87=Austrian Schilling, 88=Greek Drachma, 89=Spanish Peseta, 90=Irish Pound, 91=Italian Lira, 92=Luxembourg Franc, 93=Dutch Guilder, 94=Portuguese Escudo, 95=Finnish Markka, 96=Estonian Kroon, 97=Lithuanian Litas, 98=CFA Franc BEAC, 99=CFA Franc BCEAO, 8888=grams, 9999=ounces (troy).	1
PAYMENT_METAL_ID	Code of e-metal [®] used in spend. Will be 1 for Gold, 2 for Silver, 3 for Platinum, 4 for Palladium.	1
PAYER_ACCOUNT	e-metal [®] account of spender	100079
USD_PER_OUNCE	Exchange rate of e-metal [®] used in this transaction in US dollars per ounce.	300.00
PAYMENT_BATCH_NUM	e-gold batch number generated for this transaction. Payer and/or payee may query/search account history by this number.	758094
PAYMENT_ID	Optional merchant reference number. If present on input, this string of up to 50 characters is returned on output. Payer and/or payee may search/query account history for this value.	ID 55789-Ab9@

e-gold[®] Programmers Guide

2.2 e-metal[®] spend preview/verification

An e-metal[®] spend preview may be performed by doing an HTML form post of GET or POST variety to the URL: <https://www.e-gold.com/acct/verify.asp>

This preview / verification function might be used to check the validity of a potential spend, or to determine the actual payment ounces of a potential spend prior to executing it. Note that posting to this location **does not** actually perform any transfer of e-metal[®].

The following input fields should be used:

<u>Input Fields for submit to</u> <u>https://www.e-gold.com/acct/verify.asp</u>		
Input Field Name	Description	Example Value
AccountID	e-metal [®] account number (payer)	100079
PassPhrase	e-metal [®] account passphrase	ThisIsNotReal
Payee_Account	e-metal [®] account to spend to	100997
Amount	Amount to be spent. Must be positive numerical amount. Also must equate to at least 0.000010 ounces at current exchange rates.	5.15
PAY_IN	Code defining units payment is to be denominated in. Must be one of 1, 2, 33, 41, 44, 49, 61, 81, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 8888, 9999. 1=US Dollars, 2=Canadian Dollars, 33=French Francs, 41=Swiss Francs, 44=British Pounds, 49=Deutsche Marks, 61=Australian Dollars, 81=Japanese Yen, 85=Euro, 86=Belgian Franc, 87=Austrian Schilling, 88=Greek Drachma, 89=Spanish Peseta, 90=Irish Pound, 91=Italian Lira, 92=Luxembourg Franc, 93=Dutch Guilder, 94=Portuguese Escudo, 95=Finnish Markka, 96=Estonian Kroon, 97=Lithuanian Litas, 98=CFA Franc BEAC, 99=CFA Franc BCEAO, 8888=grams, 9999=ounces (troy).	1
WORTH_OF	e-metal [®] name to spend. Must be one of "Gold", "Silver", "Platinum", "Palladium".	Gold
Memo	Up to 50 characters to be placed in memo section of payment. The memo is visible to both payer and payee.	Thanks for dinner.
PAYMENT_ID	Optional merchant reference number. If present on input, this string of up to 50 characters is returned on output. Payer and/or payee may search/query account history for this value.	ID 55789-Ab9@

e-gold[®] Programmers Guide

Output from the verify.asp page will include HTML with embedded hidden form fields for retrieving results of the spend. These are:

<u>Output Fields from submit to https://www.e-gold.com/acct/verify.asp</u>		
Output Field Name	Description	Example Value
ERROR	Verification is unsuccessful and did not occur if this field present. Text description of error.	The UNITS 555 specified is not valid.
Payee_Account	e-metal [®] account number of recipient of spend.	100997
ACTUAL_PAYMENT_OUNCES	Actual amount of e-metal [®] that would be spent from spender. Formatted to 6 decimal places. (Note that the receiver of the e-metal [®] spend is charged a fee to receive the spend and will not net this complete amount. See PAYMENT_FEE_OUNCES field).	0.017167
PAYMENT_AMOUNT	Numerical amount of spend as entered. This is the same value as the Amount input field.	5.15
PAYMENT_UNITS	Code defining units payment is denominated in. Will be one of 1, 2, 33, 41, 44, 49, 61, 81, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 8888, 9999 and will match the PAY_IN input field. 1=US Dollars, 2=Canadian Dollars, 33=French Francs, 41=Swiss Francs, 44=British Pounds, 49=Deutsche Marks, 61=Australian Dollars, 81=Japanese Yen, 85=Euro, 86=Belgian Franc, 87=Austrian Schilling, 88=Greek Drachma, 89=Spanish Peseta, 90=Irish Pound, 91=Italian Lira, 92=Luxembourg Franc, 93=Dutch Guilder, 94=Portuguese Escudo, 95=Finnish Markka, 96=Estonian Kroon, 97=Lithuanian Litas, 98=CFA Franc BEAC, 99=CFA Franc BCEAO, 8888=grams, 9999=ounces (troy).	1
PAYMENT_METAL_ID	Code of e-metal [®] used in preview. Will be 1 for Gold, 2 for Silver, 3 for Platinum, 4 for Palladium.	1
PAYER_ACCOUNT	e-metal [®] account of spending account.	100079
USD_PER_OUNCE	Exchange rate of e-metal [®] currently in effect in US dollars per ounce.	300.00
PAYMENT_ID	Optional merchant reference number. If present on input, this string of up to 50 characters is returned on output. Payer and/or payee may search/query account history for this value.	ID 55789-Ab9@

e-gold[®] Programmers Guide

2.3 e-metal[®] history

The transaction history for an e-gold[®] account can be retrieved in CSV (Comma Separated Value) format using the URL: <https://www.e-gold.com/acct/historycsv.asp>

The following input fields should be used:

<u>Input Fields for submit to</u> <u>https://www.e-gold.com/acct/historycsv.asp</u>		
Input Field Name(s)	Description	Example Value(s)
AccountID	e-metal [®] account number (payer)	100079
PassPhrase	e-metal [®] account passphrase	ThisIsNotReal
startmonth, startday, startyear	These three fields define the <u>starting</u> day to gather account history from. Month should be a number from 1 to 12, day should be a number from 1 to 31 and year should be a 4 digit number from 1996 onward.	12 1 1999
endmonth, endday, endyear	These three fields define the <u>ending</u> day to gather account history from. Month should be a number from 1 to 12, day should be a number from 1 to 31 and year should be a 4 digit number from 1996 onward. History will be provided for records from the beginning of the start date fields to the end of the end date fields. For instance, the example values would retrieve one complete month of data for the month of December 1999.	12 31 1999
paymentsmade	If present on input, e-metal [®] payments made are included in the history display. To not include e-metal [®] payments made, do not include this as an input field.	1
paymentsreceived	If present on input, e-metal [®] payments received are included in the history display. To not include e-metal [®] payments received, do not include this as an input field.	1
inexchanges	If present on input, InExchange transactions are included in the history display. To not include InExchange transactions, do not include this as an input field.	1
outexchanges	If present on input, OutExchange transactions are included in the history display. To not include OutExchange transactions, do not include this as an input field.	1
bailments	If present on input, Direct Bailment transactions are included in the history display. To not include Direct Bailment transactions, do not include this as an input field.	1
redemptions	If present on input, Redemption transactions are included in the history display. To not include Redemption transactions, do not include this as an input field.	1
fees	If present on input, fees assessed on the account are included in the history display. To not include fees, do not include this as an input field. Examples of fee transactions are payment receive fees and monthly e-metal [®] storage charges.	1

e-gold[®] Programmers Guide

incentives	If present on input, payer and payee Incentive transactions are included in the history display. To not include Incentive transactions, do not include this as an input field.	1
batchfilter	If present on input, only transactions that have this value as a batch number are displayed.	598034
counterfilter	If present on input, only transactions that have this e-gold account number as a counter part (payments to/from this account number) are displayed.	100079
paymentidfilter	If present on input, only transactions that have this merchant reference number recorded with the transactions are displayed. Merchant reference numbers are defined using the PAYMENT_ID field of an automated e-metal [®] spend or the e-gold Shopping Cart Interface (SCI). They can be up to 50 characters.	ID 55789-Ab9@
metalfilter	If present on input, only transactions that use the given e-metal [®] are displayed. Legal values are 1, 2, 3, and 4 corresponding to Gold, Silver, Platinum, and Palladium.	2
desc	If present, sort records in descending order instead of the default ascending order.	1
oldsort	If present, defines the column to sort the history transactions on. (Default is timestamp). Choices are tstamp , transactioncode , batch_num , metal_name , weight , counteraccount_id , dollaramt , and spot_price corresponding to the available columns returned.	weight

e-gold[®] Programmers Guide

Upon submission to historycsv.asp, a set of data matching the requested input information is returned. The report returned is of ContentType application/csv. If no records are found based on the input parameters, a single line containing:

No Records Found.

Is returned. Otherwise, the first line returned contains 15 column names:

Time, Type, Batch, e-metal, Weight, To/From Number, To/From Name, Entered Amount, Entered Currency, Rate, Memo, PayeeName, PayeeAddress, Redemption Info, Merchant Ref Number

A further description of each of these items follows:

Columns returned from https://www.e-gold.com/acct/historycsv.asp		
ColumnName	Description	Example
Time	Time of transaction in GMT.	09/08/2000 00:03
Type	Transaction Type. Possible values are: InExchange OutExchange Payment Received Payment Made Redemption InEx. Commission OutEx. Commission Redemption S/H Incentive Payment Payment Receive Fee Direct Bailment	Payment Made
e-metal	Name of e-metal [®] transaction occurred in. Possible values are: Gold Silver Platinum Palladium	Gold
Weight	Weight in ounces of the transaction amount. May be positive or negative. Formatted to 6 decimal places.	-0.160760
To/From Number	For transactions involving another account (e.g. Payments Made or Received) the other account involved in the transaction.	101574
To/From Name	The e-gold account name of the To/From Number described above. This is the name currently set by that account holder on the e-gold system. It may be changed by that user at any time.	Jim Doe's e-gold Account
Entered Amount	The actual numerical value entered by the person performing the transaction.	5
Entered Currency	Symbol for the currency used to enter the transaction. Possible values are: g, oz, US\$, CAD, FRF, CHF, GBP, DEM, AUD, JPY, EUR, BEF, ATS, GRD, ESP, IEP, ITL, LUF, NLG, PTE, FIM, EEK, LTL, XAF, XOF.	g

e-gold[®] Programmers Guide

Rate	Exchange rate in effect for this transaction. In US\$ per ounce for Entered Currencies of US\$, g, oz. In same currency for other values of Entered Currency.	276.25
Memo	Memo corresponding to this transaction.	Thanks for dinner.
PayeeName	For OutExchange transactions, the name of the party the OutExchange was made to.	Tampa Power and Light
PayeeAddress	For OutExchange transactions, the address of the party the OutExchange was made to.	1234 Maple View, Tampa FL 33445
Redemption Info	For Redemptions, a description of the specie involved.	Qty: 3 Specie: 1 oz gold American Eagle (premium: 1%)
Merchant Ref Number	Merchant reference number (if any) associated with this transaction. Merchant reference numbers are defined using the PAYMENT_ID field of an automated e-metal [®] spend or the e-gold Shopping Cart Interface (SCI). They can be up to 50 characters.	ID 55789-Ab9@

e-gold[®] Programmers Guide

2.4 e-metal[®] balance

The e-metal[®] for a given account may be retrieved doing an HTML form post of GET or POST variety to the URL: <https://www.e-gold.com/acct/balance.asp>

The following input fields should be used:

<u>Input Fields for submit to</u> <u>https://www.e-gold.com/acct/balance.asp</u>		
Input Field Name	Description	Example Value
AccountID	e-metal [®] account number (payer)	100079
PassPhrase	e-metal [®] account passphrase	ThisIsnotReal

e-gold[®] Programmers Guide

Output from the balance.asp page will include HTML with embedded hidden form fields for retrieving balance information. These are:

<u>Output Fields from submit to https://www.e-gold.com/acct/balance.asp</u>		
Output Field Name	Description	Example Value
Gold_Ounces	Current available gold balance in troy ounces. Formatted to 6 decimal places.	6.004512
Gold_Grams	Current available gold balance in grams. A conversion of the Gold_Ounces field made by multiplying it by 31.103 and formatting to 4 places. The Gold_Ounces value is the definitive balance; this field is present for convenience.	186.7583
Silver_Ounces	Current available silver balance in troy ounces. Formatted to 6 decimal places.	6.004512
Silver_Grams	Current available silver balance in grams. A conversion of the Silver_Ounces field made by multiplying it by 31.103 and formatting to 4 places. The Silver_Ounces value is the definitive balance; this field is present for convenience.	186.7583
Platinum_Ounces	Current available platinum balance in troy ounces. Formatted to 6 decimal places.	6.004512
Platinum_Grams	Current available platinum balance in grams. A conversion of the Platinum_Ounces field made by multiplying it by 31.103 and formatting to 4 places. The Platinum_Ounces value is the definitive balance; this field is present for convenience.	186.7583
Palladium_Ounces	Current available palladium balance in troy ounces. Formatted to 6 decimal places.	6.004512
Palladium_Grams	Current available palladium balance in grams. A conversion of the Palladium_Ounces field made by multiplying it by 31.103 and formatting to 4 places. The Palladium_Ounces value is the definitive balance; this field is present for convenience.	186.7583
ERROR	Access is unsuccessful and did not occur if this field present. Text description of error.	Could not login to account. Either Automation Access not enabled, IP Attribute does not match, or PassPhrase is incorrect.

If no transactions have ever occurred for a particular e-metal[®] type, the output fields for that e-metal[®] will not be present.

e-gold[®] Programmers Guide

2.5 Exchange rates

The current and historical exchange rates for the various e-metal[®]s vs. several national currencies may be retrieved doing an HTML form post of GET or POST variety to the URL: <http://www.e-gold.com/unsecure/metaldata.asp>

The following input fields should be used:

Input Fields for submit to http://www.e-gold.com/unsecure/metaldata.asp		
Input Field Name	Description	Example Value
LATEST	If this input field is present (value unimportant) then only the current exchange rates will be returned. When it is not present, the past 3 months of exchange rate data is returned.	1
StartDate	Optional starting date formatted in seconds since GMT to retrieve data from. If present, EndDate must also be specified.	913930691
EndDate	Optional ending date formatted in seconds since GMT to retrieve data through. If present, StartDate must also be specified.	914248432
GOLD	Optional field indicating retrieval of Gold exchange rates. Default is to return Gold, Silver, Platinum, Palladium exchange rates in that order. Other combinations may be specified using the input fields GOLD , SILVER , PLATINUM , PALLADIUM . Order of output data is always gold, then silver, then platinum, then palladium.	1
SILVER	See above	1
PLATINUM	See above	1
PALLADIUM	See above	1
CUR	Optional input field indicating display of exchange rates in this currency. Default is US Dollars. Choices are: AUD = Australian Dollars, DEM = Deutsche Marks JPY = Japanese Yen, GBP = British Pounds CHF = Swiss Francs, FRF = French Francs CAD = Canadian Dollars, EUR = Euro BEF = Belgian Franc, ATS = Austrian Schilling GRD = Greek Drachma, ESP = Spanish Peseta IEP = Irish Pound, ITL = Italian Lira LUF = Luxembourg Franc, NLG = Dutch Guilder PTE = Portuguese Escudo, FIM = Finnish Markka EEK = Estonian Kroon, LTL = Lithuanian Litas XAF = CFA Francs (BEAC), XOF = CFA Franc (BCEAO)	AUD

Output from the metaldata.asp page will be a set of comma delimited ASCII data. Each set of data will begin with a date/time at which the set of exchange rates took effect. This date/time will be in GMT and will be formatted like "Month/Day/Year"

e-gold[®] Programmers Guide

Hour:Minutes/Seconds AM/PM” example: “6/2/00 9:36:45 PM“ corresponding to June 2nd 2000 21:36:45 GMT. The date/time will be followed by one to four values (depending on GOLD, SILVER, PLATINUM, PALLADIUM inputs) giving the exchange rate per troy ounce for the selected e-metal[®]s in the currency selected (US Dollars default).

Some examples of input and output:

Example 1 “Current exchange rates for all 4 e-metal[®] types in US dollars”

`http://www.e-gold.com/unsecure/metaldata.asp?LATEST=1`

gives:

9/28/99 12:24:50 PM, 294.500 , 5.430 , 397.000 , 366.000

Which provides the gold exchange rate of 294.50 US dollars per ounce at the given time (GMT). The other three numbers are the silver, platinum, and palladium rates per ounce in US dollars.

Example 2 “Historical rates for all 4 e-metal[®] types in US dollars”

`http://www.e-gold.com/unsecure/metaldata.asp?StartDate=913930691&EndDate=914248432`

gives:

12/17/98 9:38:11 PM, 291.500 , 4.930 , 350.000 , 326.000
12/18/98 2:24:30 PM, 292.000 , 4.940 , 349.000 , 310.000
12/18/98 8:24:11 PM, 289.500 , 4.910 , 345.000 , 310.000
12/18/98 9:43:28 PM, 289.500 , 4.910 , 345.000 , 310.000
12/21/98 11:41:46 AM, 289.500 , 4.960 , 345.000 , 310.000
12/21/98 1:53:52 PM, 289.500 , 4.900 , 346.000 , 308.000

Again, the values are US dollars per troy ounce in effect at the given date/time (GMT).

Example 3 “Single type of e-metal[®] in US dollars”

`http://www.e-gold.com/unsecure/metaldata.asp?LATEST=1&GOLD=1`

gives:

9/28/99 12:47:24 PM, 296.000

The single value returned is the exchange rate of gold in US dollars per troy ounce starting at the given time (GMT).

Example 4 “Two types of e-metal[®] in US dollars”

`http://www.e-gold.com/unsecure/metaldata.asp?LATEST=1&SILVER=1&GOLD=1`

gives:

9/28/99 12:47:24 PM, 296.000 , 5.440

The two values returned are the exchange rates of gold and silver in US dollars per troy ounce starting at the given time (GMT). Note that the order of returned rates does not

e-gold[®] Programmers Guide

depend on the input order. The returned values are **always** gold, silver, platinum, palladium.

Example 5 “Exchange rates in currencies other than US dollars”

`http://www.e-gold.com/unsecure/metaldatas.asp?LATEST=1&CUR=JPY`

gives:

`9/28/99 12:47:24 PM, 31409.168 , 577.250 , 42126.486 , 38837.012`

This example retrieves the current exchange rates in Japanese Yen per troy ounce for the four e-metal[®] types.

3. Example Scripts

The following scripts may be used as examples to guide you in your own use of the e-gold[®] system. These examples use the free tool cURL (see <http://curl.haxx.se/>) to perform form posts to the e-gold URLs. cURL is useful in that it supports the SSL required by e-gold (using OpenSSL) and is available for several different platforms. You will need the SSL capable cURL along with the OpenSSL libraries for your platform (most likely you can obtain these at <http://curl.haxx.se/download.html>) but you may also wish to visit <http://www.openssl.org>.

For perl scripts, you'll need to have perl installed ☺. If you are trying to run these scripts on windows, a good choice might be the activestate perl distribution available at <http://www.activestate.com/>. Note that any password stty non-echo lines won't work on Windows platforms.

3.1 Security notes

If you are placing an automation script in a situation where it accepts input data from untrusted sources, you'll want to try and protect your script from malicious attempts to make it do something other than it should. Some hints in this regard:

1. **Do not embed your e-gold account passphrase in your automation program. Rather, prompt the user for the passphrase when the program runs.**
2. **Examine your automation program closely for problems. You are just as responsible for e-metal[®] transactions initiated by an automation program as if you had entered the transaction by hand. (i.e. e-metal[®] spends are not reversible regardless of the initiation method).**
3. **Escape any system arguments. If you are using PHP take a look at the EscapeShellCmd and EscapeShellArg functions. If you are doing SQL commands, replace all delimiters in input data.**

e-gold[®] Programmers Guide

3.2 Disclaimer

Disclaimer

The Author [e-gold Ltd.] accepts no responsibility for damages to persons, property or data incurred through the use of these script(s). To the maximum extent permitted by law, in no event shall the Author [e-gold Ltd.] be liable for any damages whatsoever (including, without limitation, damages for loss of business profits, business interruption, loss of business information, or other pecuniary loss) arising out of the use or inability to use this software, even if the Author has been advised of the possibility of such damages.

This product is supplied as-is, with no warranties express or implied.

Use this software at your own risk.

3.3 Perl script (using cURL) to retrieve e-gold[®] account history

This script accepts a variety of command line arguments allowing retrieval of specified transactions from an e-gold account. The script prompts the user to enter the passphrase for the given account. Result data is displayed directly on stdout as it is received from the e-gold system.

```
#!/usr/bin/perl
#
# perl script to retrieve e-gold account history
# from maintenance release e-gold system
# version 1.0
# free for use, modification, redistribution
#
#Disclaimer
#The Author [e-gold Ltd.] accepts no
#responsibility for damages to persons, property or data incurred
#through the use of this script(s). To the maximum extent permitted
#by law, in no event shall the Author [e-gold Ltd.] be
#liable for any damages whatsoever (including, without limitation,
#damages for loss of business profits, business interruption, loss of
#business information, or other pecuniary loss) arising out of the use
#or inability to use this software, even if the Author has been advised
#of the possibility of such damages.
#
#This product is supplied as-is, with no warranties express or implied.
#Use this software at your own risk.
#
if ($#ARGV < 0) {
    &command_help;
}
# pick up account number we will be operating on...
$account = $ARGV[0];
shift;
require "getopts.pl";
# parse the arguments...
&Getopts ("s:e:b:f:m:r:t:") || &command_help;
# default dates are today...
($todaysec,$todaymin,$todayhour,$todaydom,$todaymonth,$todayyear)
    = localtime(time);
# convert to one relative numbers the e-gold site uses
# and set up our default time frame of today -> today
$startday = $todaydom;
$startmonth = $todaymonth + 1;
$startyear = $todayyear + 1900;
$endday = $todaydom;
$endmonth = $todaymonth + 1;
$endyear = $todayyear + 1900;
# if dates provided, use them instead...
if ($opt_s){
    ($startmonth, $startday, $startyear) = split(/\//, $opt_s, 3);
}
}
```

e-gold[®] Programmers Guide

```
if ($opt_e){
    ($endmonth, $endday, $endyear) = split(/\\/, $opt_e, 3);
}
# get batch number filter if present
$batchfilter = "";
if ($opt_b){
    $batchfilter = $opt_b;
}
# get to/from account number number filter if present
$counterfilter = "";
if ($opt_f){
    $counterfilter = $opt_f;
}
# get to/from account number number filter if present
$metalfilter = "";
if ($opt_m){
    if(index($opt_m,"gold") > 0){
        $metalfilter = "1";
    }
    if(index($opt_m,"silver") > 0){
        $metalfilter = "2";
    }
    if(index($opt_m,"platinum") > 0){
        $metalfilter = "3";
    }
    if(index($opt_m,"palladium") > 0){
        $metalfilter = "4";
    }
}
# get merchant reference number if present
$paymentidfilter = "";
if ($opt_r){
    $paymentidfilter = $opt_r;
}
# get list of transaction types we check
$paymentsreceived = 1;
$paymentsmade = 1;
$inexchanges = 1;
$outexchanges = 1;
$bailments = 1;
$redemptions = 1;
$fees = 1;
$incentives = 1;
if ($opt_t){
    if(index($opt_t, "paymentsreceived") < 0){
        $paymentsreceived = 0;
    }
    if(index($opt_t, "paymentsmade") < 0){
        $paymentsmade = 0;
    }
    if(index($opt_t, "inexchanges") < 0){
        $inexchanges = 0;
    }
    if(index($opt_t, "outexchanges") < 0){
        $outexchanges = 0;
    }
    if(index($opt_t, "bailments") < 0){
        $bailments = 0;
    }
    if(index($opt_t, "redemptions") < 0){
        $redemptions = 0;
    }
    if(index($opt_t, "fees") < 0){
        $fees = 0;
    }
    if(index($opt_t, "incentives") < 0){
        $incentives = 0;
    }
}
#print "Days: $days, Newest: $newest, Trans: $trans\n";

sub command_help
{
    print "Usage: $0 account_number [-s startingdate] [-e endingdate] [-b batch_number] [-f
account_number] [-m metal_name] [-r ref_number] [-t'transactiontype']\n";
    print "    -s starting date, default = today\n";
}
```

e-gold[®] Programmers Guide

```
print " -e ending date, default = today\n";
print " -b match batch number, default = all batches\n";
print " -f to/from given account number, default = all\n";
print " -m only given metal type, default = all\n";
print " -r match given merchant reference number, default = all\n";
print " -t display given transaction types, default = all\n";
print " enter dates in format mm/dd/yyyy\n";
print " metal names are: gold, silver, platinum, palladium\n";
print " transactiontypes:\n";
print "   paymentsreceived\n";
print "   paymentsmade\n";
print "   inexchange\n";
print "   outexchange\n";
print "   bailments\n";
print "   redemptions\n";
print "   fees\n";
print "   incentives\n";
print " example: $0 100666 -s 1/1/2000 -e 12/31/2000 -t
paymentsreceived,paymentsmade\n";
  exit 1;
}
# prompt user for password
print "\nPassphrase for $account?";
if ( system('stty -echo') != 0) {
    die "Error setting terminal to not echo\n";
}
$pp = <STDIN>;
chop($pp);
system('stty echo');
print "\n";
#build up the arguments for the history command
$curlargs = "initial=1&startmonth=$startmonth";
$curlargs .= "&startday=$startday";
$curlargs .= "&startyear=$startyear";
$curlargs .= "&endmonth=$endmonth";
$curlargs .= "&endday=$endday";
$curlargs .= "&endyear=$endyear";
if($paymentsmade > 0 ){
    $curlargs .= "&paymentsmade=1";
}
if($paymentsreceived > 0 ){
    $curlargs .= "&paymentsreceived=1";
}
if($inexchange > 0 ){
    $curlargs .= "&inexchange=1";
}
if($outexchange > 0 ){
    $curlargs .= "&outexchange=1";
}
if($bailments > 0 ){
    $curlargs .= "&bailments=1";
}
if($redemptions > 0 ){
    $curlargs .= "&redemptions=1";
}
if($fees > 0 ){
    $curlargs .= "&fees=1";
}
if($incentives > 0 ){
    $curlargs .= "&incentives=1";
}
if(length($batchfilter) > 0 ){
    $curlargs .= "&batchfilter=$batchfilter";
}
if(length($counterfilter) > 0 ){
    $curlargs .= "&counterfilter=$counterfilter";
}
if(length($paymentidfilter) > 0 ){
    $curlargs .= "&paymentidfilter=$paymentidfilter";
}
if(length($metalfilter) > 0 ){
    $curlargs .= "&metalfilter=$metalfilter";
}
}
$curlargs .= "&PassPhrase=$pp";
$curlargs .= "&AccountID=$account";
# print "\n$curlargs\n";
```

e-gold[®] Programmers Guide

```
$sysstring = "curl -s -d ";
$sysstring .= '';
$sysstring .= $curlargs;
$sysstring .= "&CURPAGE=0";
$sysstring .= '';
$sysstring .= " https://www.e-gold.com/acct/historycsv.asp";
#print $sysstring;
open(foo, "$sysstring|");
while(<foo>){
    $lineout = $_;
    # if we get back some kind of HTML, the e-gold server is unhappy with us...
    if($lineout =~ /<html>/){
        print "\nError occurred.\n";
        close foo;
    }
    else {
        print $lineout;
    }
}
close foo;
# all done
print "\n";
```

3.4 Perl script (using cURL) to preview e-gold[®] spend

This script accepts a fixed number of command line arguments allowing a preview of a spend. The script prompts the user to enter the passphrase for the given account. Result data is displayed on stdout and is either an error string indicating something incorrect in the preview, or the actual weight in ounces that would be spent.

```
#!/usr/bin/perl
#
# perl script to preview a spend
# from maintenance release e-gold system
# version 1.0
# free for use, modification, redistribution
#
#
#Disclaimer
#The Author [e-gold Ltd.] accepts no
#responsibility for damages to persons, property or data incurred
#through the use of this script(s). To the maximum extent permitted
#by law, in no event shall the Author [e-gold Ltd.] be
#liable for any damages whatsoever (including, without limitation,
#damages for loss of business profits, business interruption, loss of
#business information, or other pecuniary loss) arising out of the use
#or inability to use this software, even if the Author has been advised
#of the possibility of such damages.
#
#This product is supplied as-is, with no warranties express or implied.
#Use this software at your own risk.
#
if ($#ARGV < 4) {
    &command_help;
}
# pick up account number we will be spending from...
$accountfrom = $ARGV[0];
shift;
# pick up account number we will be spending to...
$accountto = $ARGV[0];
shift;
# pick up amount
$amount = $ARGV[0];
shift;
# pick up units
$units = uc($ARGV[0]);
shift;
$unitsok = 0;
if(index($units,"USD") == 0){
    $units = 1;
    $unitsok = 1;
}
if(index($units,"OZ") == 0){
    $units = 9999;
    $unitsok = 1;
}
if(index($units,"G") == 0){
    $units = 8888;
    $unitsok = 1;
}
if(index($units,"CAD") == 0){
    $units = 2;
    $unitsok = 1;
}
if(index($units,"FFR") == 0){
    $units = 33;
    $unitsok = 1;
}
if(index($units,"CHF") == 0){
    $units = 41;
    $unitsok = 1;
}
if(index($units,"GBP") == 0){
    $units = 44;
```

e-gold[®] Programmers Guide

```
        $unitsok = 1;
    }
    if(index($units,"DEM") == 0){
        $units = 49;
        $unitsok = 1;
    }
    if(index($units,"AUD") == 0){
        $units = 61;
        $unitsok = 1;
    }
    if(index($units,"JPY") == 0){
        $units = 81;
        $unitsok = 1;
    }
    if($unitsok == 0) {
        die "$units is Invalid, choose oz, g, usd, gbp, aud, jpy, dem, cad, ffr, chr";
    }
    # pick up e-metal type
    $metal = uc($ARGV[0]);
    shift;
    $metalok = 0;
    if(index($metal,"GOLD") == 0){
        $metal = "Gold";
        $metalok = 1;
    }
    if(index($metal,"SILVER") == 0){
        $metal = "Silver";
        $metalok = 1;
    }
    if(index($metal,"PLATINUM") == 0){
        $metal = "Platinum";
        $metalok = 1;
    }
    if(index($metal,"PALLADIUM") == 0){
        $metal = "Palladium";
        $metalok = 1;
    }
    if($metalok == 0) {
        die "$metal is invalid, choose gold, silver, platinum, or palladium";
    }
    sub command_help
    {
        print "Usage: $0 from to amount units metal\n";
        print " units can be one of oz, g, usd, aud, cad, dem, gbp, jpy, chf, ffr\n";
        print " metal can be one of gold, silver, platinum, palladium\n";
        print "\nextample:\n";
        print "$0 101574 100998 1 usd gold\n";
        print "to preview a spend of 1 US dollar equivalent of gold from\n";
        print " account 101574 to account 100998\n";
        exit 1;
    }
    # prompt user for password
    print "\nPassphrase for $accountfrom?";
    if ( system('stty -echo') != 0) {
        die "Error setting terminal to not echo\n";
    }
    $pp = <STDIN>;
    chop($pp);
    system('stty echo');
    print "\n";

    #build up the arguments for the preview command
    $curlargs = "Payee_Account=$accountto";
    $curlargs .= "&Amount=$amount";
    $curlargs .= "&PAY_IN=$units";
    $curlargs .= "&WORTH_OF=$metal";
    $curlargs .= "&Memo=automatic spend";
    $curlargs .= "&PassPhrase=$pp";
    $curlargs .= "&AccountID=$accountfrom";
    # print "\n$curlargs\n";
    $sysstring = "curl -s -d ";
    $sysstring .= "'";
    $sysstring .= $curlargs;
    $sysstring .= "'";
    $sysstring .= " https://www.e-gold.com/acct/verify.asp";
```

e-gold[®] Programmers Guide

```
#print $sysstring;
# read everything into one buffer
undef $/;
open(foo, "$sysstring|");
while(<foo>){
  # pull out the hidden fields...the pattern starts with "hidden name=" and
  # ends with ">"
  while( /hidden name=(.*?)>/gs ) {
    # see if there was some kind of error
    if(index($1, "ERROR") == 0) {
      $errstring = $1;
      $errstring =~ ( /value="(.)"/ );
      print "Error on preview is: $1\n";
      exit -1;
    }

    # print out the information we care about from the preview
    if(index($1, "ACTUAL_PAYMENT_OUNCES") == 0) {
      $ozstring = $1;
      $ozstring =~ ( /value="(.)"/ );
      print "Successful Preview.\n";
      print "Weight of this payment would be: $1 oz.\n";
    }
  }
}
close foo;
# all done
print "\n";
```

3.5 Perl script (using cURL) to display e-gold[®] balance

This script accepts an e-gold account number on the command line and displays an e-gold account balance for that account. The script prompts the user to enter the passphrase for the given account. Result data is displayed on stdout.

```
#!/usr/bin/perl
#
# perl script to display account balances
# from maintenance release e-gold system
# version 1.0
# free for use, modification, redistribution
#
#Disclaimer
#The Author [e-gold Ltd.] accepts no
#responsibility for damages to persons, property or data incurred
#through the use of this script(s). To the maximum extent permitted
#by law, in no event shall the Author [e-gold Ltd.] be
#liable for any damages whatsoever (including, without limitation,
#damages for loss of business profits, business interruption, loss of
#business information, or other pecuniary loss) arising out of the use
#or inability to use this software, even if the Author has been advised
#of the possibility of such damages.
#
#This product is supplied as-is, with no warranties express or implied.
#Use this software at your own risk.
#
if ($#ARGV < 0) {
    &command_help;
}
# pick up account number we will be getting balance on...
$account= $ARGV[0];
sub command_help
{
    print "Usage: $0 account_number\n";
    print "    will display e-gold account balance for given account.\n";
    print "\nextample:\n";
    print "$0 101574\n";
    exit 1;
}
# prompt user for password
print "\nPassphrase for $accountfrom?";
if ( system('stty -echo') != 0) {
    die "Error setting terminal to not echo\n";
}
$pp = <STDIN>;
chop($pp);
system('stty echo');
print "\n";
#build up the arguments for the preview command
$curlargs = "PassPhrase=$pp";
$curlargs .= "&AccountID=$account";
$sysstring = "curl -s -d ";
$sysstring .= "'";
$sysstring .= $curlargs;
$sysstring .= "'";
$sysstring .= " https://www.e-gold.com/acct/balance.asp";
# read everything into one buffer
undef $/;
open(foo, "$sysstring|");
while(<foo>){
    # pull out the hidden fields...the pattern starts with "hidden name=" and
    # ends with ">"
    while( /hidden name=(.*?)>/gs ) {
        # see if there was some kind of error
        if(index($1, "ERROR") == 0) {
            $errstring = $1;
            $errstring =~ ( /value="(.*?)"/ );
            print "Error on balance is: $1\n";
            exit -1;
        }
    }
    # print out the information we care about from the preview
```

e-gold[®] Programmers Guide

```
    if(index($1, "_Ounces") > 0) {
        $matching = $1;
        $ozstring = $matching;
        $ozstring =~ ( /value="(.)"/ );
        $ozstring = $1;
        $namestring = $matching;
        $namestring =~ ( /(.)_Ounces value/ );
        print "$1 balance of $account: $ozstring oz.\n";
    }
}
close foo;
# all done
print "\n";
```