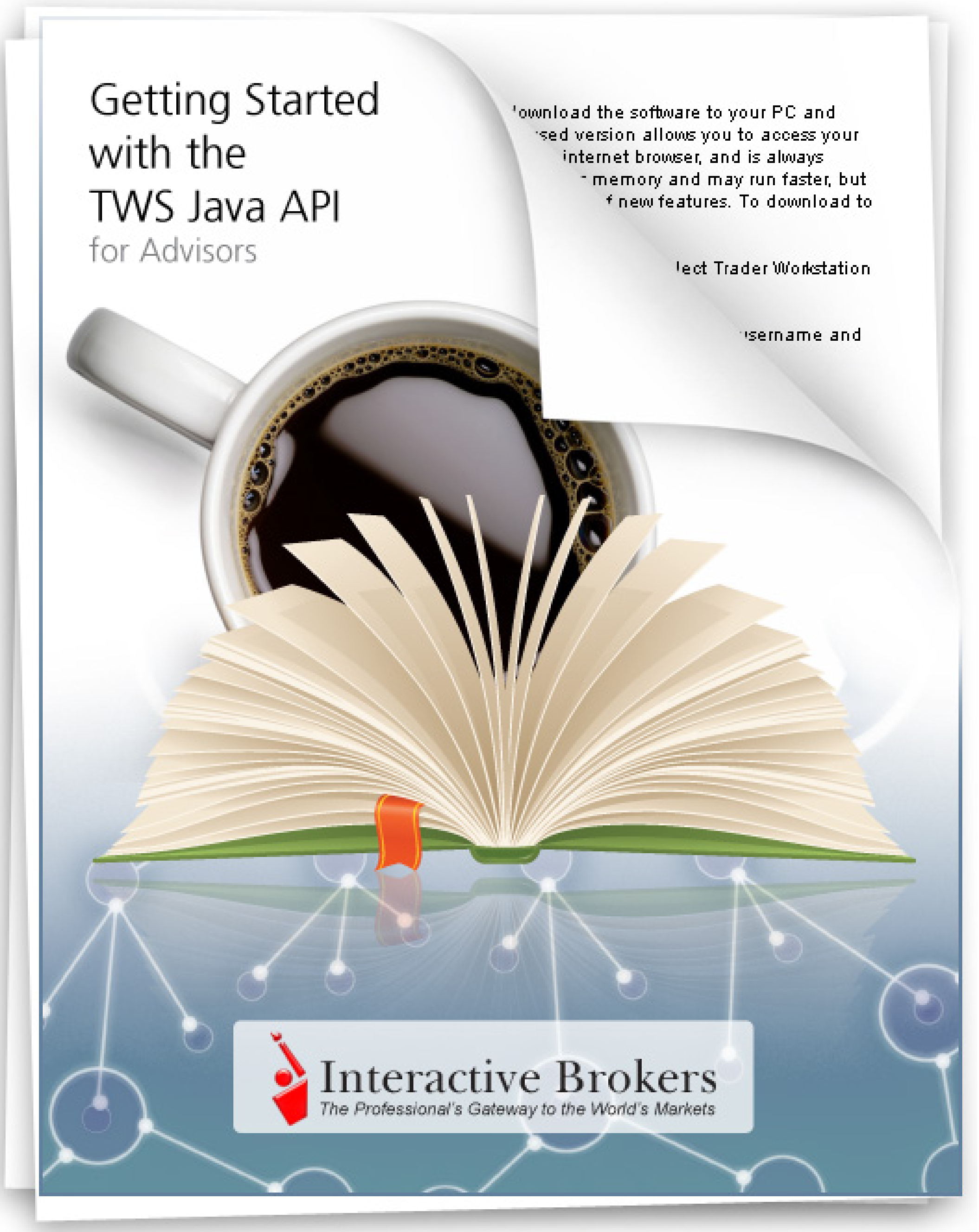
Getting Started with the TWS Java API

for Advisors


'ownload the software to your PC and
`sed version allows you to access your
internet browser, and is always
' memory and may run faster, but
f new features. To download to
lect Trader Workstation
'sername and

Interactive Brokers

The Professional's Gateway to the World's Markets

Getting Started with the TWS Java API for Advisors
March 2011
Supports TWS API Release 9.64

Contents

1

Introduction

You might be looking at this book for any number of reasons, including:

- You love IB's TWS, and are interested in seeing how using its API can enhance your trading.
- You use another online trading application that doesn't provide the functionality of TWS, and you want to find out more about TWS and its API capabilities.
- You never suspected that there was a link between the worlds of trading/financial management and computer programming, and the hint of that possibility has piqued your interest.

Or more likely you have a reason of your own. Regardless of your original motivation, you now hold in your hands a unique and potentially priceless tome of information. Well, maybe that's a tiny bit of an exaggeration. However, the information in this book, which will teach you how to access and manage the robust functionality of IB's Trader Workstation through our TWS Java API, could open up a whole new world of possibilities and completely change the way you manage your trading environment. Keep reading to find out how easy it can be to build your own customized trading application.

This book assumes that you are a Financial Advisor. To learn how to get started using the basic trading features of the TWS Java API, see the Getting Started with the TWS Java API guide.

How to Use this Book

Before you get started, you should read this section to learn how this book is organized, and see which graphical conventions are used throughout.

Our main goal is to give active traders and investors the tools they need to successfully implement a custom trading application (i.e. a trading system that you can customize to meet your specific needs), and that doesn't have to be monitored every second of the day. If you're not a trader or investor you probably won't have much use for this book, but please, feel free to read on anyway!

Throughout this book, we use the acronym "TWS" in place of "Trader Workstation." So when you see "TWS" anywhere, you'll know we're talking about Trader Workstation.

Before you read any further, we need to tell you that this book focuses on the TWS side of the Java API - we don't really help you to learn Java. If you aren't a fairly proficient Java programmer, or at least a very confident and bold beginner, this may be more than you want to take on. We suggest you start with a beginner's Java programming book, and come back to us when you're comfortable with Java.

Organization

We've divided this book into five major sections, each of which comprises a number of smaller subsections, and each of **those** have even smaller groupings of paragraphs and figures…well, you get the picture. Here's how we've broken things down:

Chapter 1: Introduction

This chapter (the one you're currently reading) tells you what you'll find in this guide and how to use it.

Chapter 2: Introducing the TWS Java API

This chapter helps you answer those important questions you need to ask before you can proceed - questions such as "What can TWS do for me?" and "Why would I use an API?" and "If I WERE to use an API, what does the Java platform have to offer me?" and even "What other API choices do I have?" If you already know you want to learn about the TWS API, just skip on ahead.

Chapter 3: Preparing to Use the TWS Java API

Chapter 3 walks you through the different things you'll need to do before your API application can effectively communicate with TWS. We'll help you download and install the API software, configure TWS and get the Java Test Client sample application up and running. A lot of this information is very important when you first get started, but once it's done, well, it's done, and you most likely won't need much from this section once you've completed it.

Chapter 4: The Java Test Client for Advisors

Chapter 4 gets you working with the Java Test Client: learning how to trade for and allocate shares among multiple client accounts. We'll tell you exactly what methods you need to use to send info to TWS, and just what TWS will send you back. We've already documented the method parameters, descriptions and valid values in the API Reference Guide, so instead of duplicating efforts and filling this book up with those important reference tidbits, we provide targeted links to different sections of the users' guide as we need them.

Chapter 5: Allocation Methods and Profiles

This chapter provides a handy reference for advisors who need to refresh their memory about the different allocation methods and profiles that can be set up in Trader Workstation.

Chapter 6: Where to Go from Here

After filling your head with boatfuls of API knowledge, we wouldn't dream of sending you off empty-handed! This chapter includes some additional information about linking to TWS using our Java API, then tells you how to keep abreast of new API releases (which of course means new features you can incorporate into your trading plan), how to navigate the Interactive Brokers website to find support and information, and what resources we recommend to help you answer questions outside the realm of IB support, questions such as "Why isn't my Java JDK working?"

Footnotes and References

[1]Any symbols displayed are for illustrative purposes only and are not intended to portray a recommendation.

Icons

TWS-Related	When you see this guy, you know that there is something that relates specifically to TWS: a new feature to watch for, or maybe something you're familiar with in TWS and are looking for in the API.
Java Tip	The Java tips are things we noted and think you might find useful. They don't necessarily relate only to TWS. We don't include too many of these, but when you see it you should check it out - it will probably save you some time.
Important!	This shows you where there is a particularly useful or important point being made.
Take a Peek!	You may want to take a peek, but it isn't the end of the world if you don't.
Go Outside!	This icon denotes references outside of this book that we think may help you with the current topic, including links to the internet or IB site, or a book title.

Document Conventions

Here's a list of document conventions used in the text throughout this book.

Convention	Description	Examples
Bold	Indicates: • menus • screens • windows • dialogs • buttons • tabs • keys you press • names of classes and methods	On the **Tickers** page, select a row by clicking the row number in the far left column… Press **Ctrl+C** to copy…
Italics	Indicates: • commands in a menu • objects on the screen, such as text fields, check boxes, and drop-down lists	To access the users' guide, under the **Software** menu, select *Trader Workstation*, then click *Users' Guide*.

In addition, Java code snippets appear in the following format:

EClientSocket constructor

```
EClientSocket   m_client = new EClientSocket(this);
```

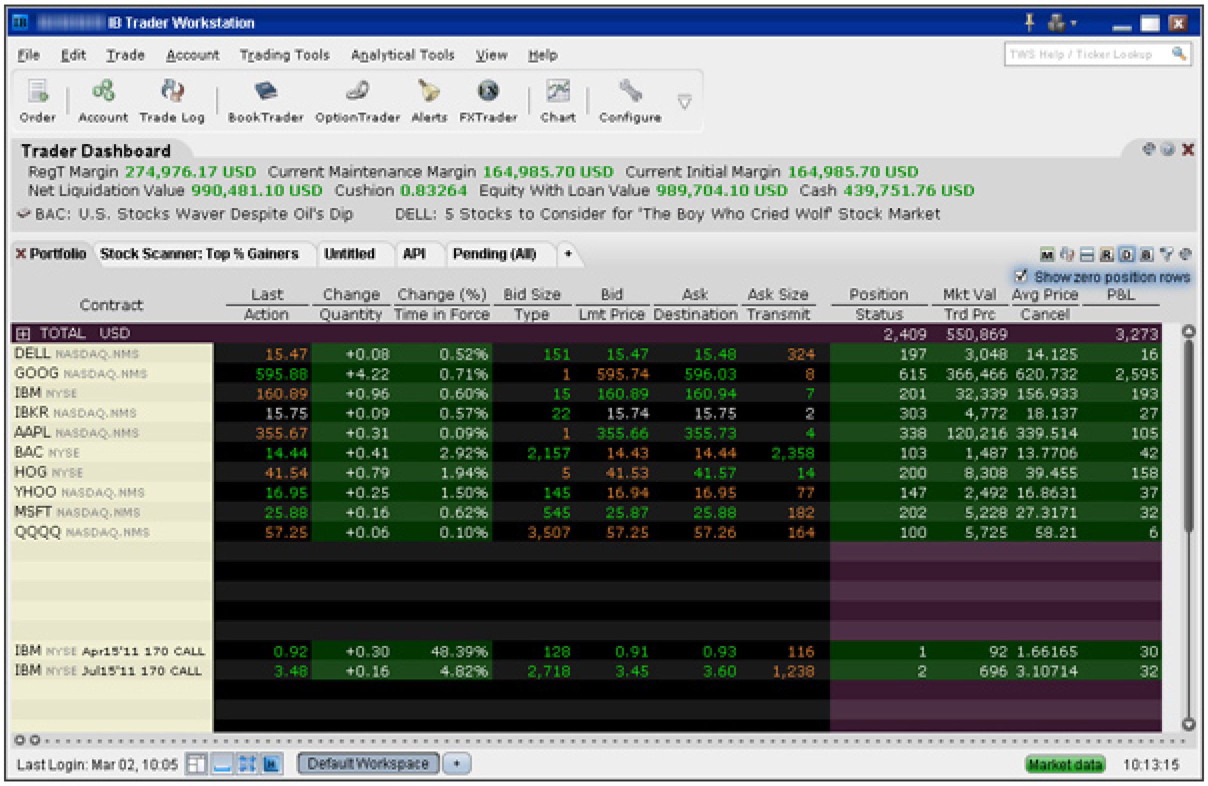
2 TWS and the Java API

The best place to start is by getting an idea of what Trader Workstation (TWS), is all about. In this section, first we'll describe TWS and some of its major features. Then we'll explain how the API can be used to enhance and customize your trading environment. Finally, we'll give you a summary of some of the things the Java API can do for you!

Here's what you'll find in this chapter:

What is Trader Workstation?

Interactive Brokers' Trader Workstation, or TWS, is an online trading platform that lets you trade and manage orders for all types of financial products (including stocks, bonds, options, futures and Forex) on markets all over the world - all from a single spreadsheet-like screen.


IB Trader Workstation
File Edit Trade Account Trading Tools Analytical Tools View Help
TWS Help / Ticker Lookup
Order Account Trade Log BookTrader OptionTrader Alerts FXTrader Chart Configure
Portfolio
Stock Scanner: Top % Gainers
Untitled
API
Pending (All)
Show zero position rows
Default Workspace
Market data

Trader Dashboard

RegT Margin 274,976.17 USD Current Maintenance Margin 164,985.70 USD Current Initial Margin 164,985.70 USD
Net Liquidation Value 990,481.10 USD Cushion 0.83264 Equity With Loan Value 989,704.10 USD Cash 439,751.76 USD

BAC: U.S. Stocks Waver Despite Oil's Dip   DELL: 5 Stocks to Consider for 'The Boy Who Cried Wolf' Stock Market

Contract	Last / Action	Change / Quantity	Change (%) / Time in Force	Bid Size / Type	Bid / Lmt Price	Ask / Destination	Ask Size / Transmit	Position / Status	Mkt Val / Trd Prc	Avg Price / Cancel	P&L
TOTAL USD								2,409	550,869		3,273
DELL NASDAQ.NMS	15.47	+0.08	0.52%	151	15.47	15.48	324	197	3,048	14.125	16
GOOG NASDAQ.NMS	595.88	+4.22	0.71%	1	595.74	596.03	8	615	366,466	620.732	2,595
IBM NYSE	160.89	+0.96	0.60%	15	160.89	160.94	7	201	32,339	156.933	193
IBKR NASDAQ.NMS	15.75	+0.09	0.57%	22	15.74	15.75	2	303	4,772	18.137	27
AAPL NASDAQ.NMS	355.67	+0.31	0.09%	1	355.66	355.73	4	338	120,216	339.514	105
BAC NYSE	14.44	+0.41	2.92%	2,157	14.43	14.44	2,358	103	1,487	13.7706	42
HOG NYSE	41.54	+0.79	1.94%	5	41.53	41.57	14	200	8,308	39.455	158
YHOO NASDAQ.NMS	16.95	+0.25	1.50%	145	16.94	16.95	77	147	2,492	16.8631	37
MSFT NASDAQ.NMS	25.88	+0.16	0.62%	545	25.87	25.88	182	202	5,228	27.3171	32
QQQQ NASDAQ.NMS	57.25	+0.06	0.10%	3,507	57.25	57.26	164	100	5,725	58.21	6
IBM NYSE Apr15'11 170 CALL	0.92	+0.30	48.39%	128	0.91	0.93	116	1	92	1.66165	30
IBM NYSE Jul15'11 170 CALL	3.48	+0.16	4.82%	2,718	3.45	3.60	1,238	2	696	3.10714	32

Last Login: Mar 02, 10:05   10:13:15

To get a little bit of a feel for TWS, go to the IB website and try TWS demo application. Its functionality is slightly limited and it only supports a small number of symbols, but you'll definitely get the idea. Once you have an approved, funded account you'll also be able to use PaperTrader, our simulated trading tool, with paper-money funding in the amount of $100,000, which you can replenish at any time through TWS Account Management.

What Can You Do with TWS?

So, what can you do with TWS? For starters, you can:

- Send and manage orders for all sorts of products (all from the same screen!);
- Monitor the market through Level II, NYSE Deep Book and IB's Market Depth;
- Keep a close eye on all aspects of your account and executions;
- Use Technical, Fundamental and Price/Risk analytics tools to spot trends and analyze market movement;
- Completely customize your trading environment through your choice of modules, features, tools, fonts and colors, and user-designed workspaces.

Basically, almost anything you can think of TWS can do - or will be able to do soon. We are continually adding new features, and use the latest technology to make things faster, easier and more efficient. As a matter of fact, it was this faith in technology's ability to improve a trader's success in the markets (held by IB's founder and CEO Thomas Peterffy) that launched this successful endeavor in the first place. Since the introduction of TWS in 1995, IB has nurtured this relationship between technology and trading almost to the point of obsession!

A Quick Look at TWS

This section gives you a brief overview of the most important parts of TWS.

The TWS Quote Monitor

First is the basic TWS Quote Monitor. It's laid out like a spreadsheet with rows and columns. To add tickers to a page, you just click in the Underlying column, type in an underlying symbol and press Enter, and walk through the steps to select a product type and define the contract. Voila! You now have a live market data line on your trading window. It might be for a stock, option, futures or bond contract. You can add as many of these as you want, and you can create another window, or trading page, and put some more on that page. You can have any and all product types on a single page, maybe sorted by exchange, or you can have a page for stocks, a page for options, etc. Once you get some market data lines on a trading page, you're ready to send an order.

The Order Ticket

What? An order ticket? Sure, we have an order ticket if that's what you really want. But we thought you might find it easier to simply click on the bid or ask price and have us create a complete order line instantly, right in front of your eyes! Look it over, and if it's what you want click a button to transmit the order. You can easily change any of the order parameters right on the order line. Then just click the green Transmit guy to transmit your order! It's fast and it's easy, and you can even customize this minimal two-click procedure (by creating hotkeys and setting order defaults for example) so that you're creating and transmitting orders with just ONE click of the mouse.

Real-Time Account Monitoring

TWS also provides a host of real-time account and execution reporting tools. You can go to the Account Window at any time to see your account balance, total available funds, net liquidation and equity with loan value and more. You can also monitor this data directly from your trading window using the Trader Dashboard, a monitoring tool you can configure to display the last price for any contracts and account-related information directly on your trading window.

So - TWS is an all-inclusive, awesome powerful trading tool. You may be wondering, "Where does an API fit in with this?" Read on to discover the answer to that question.

For more information on TWS, see the TWS Users' Guide on our web site.

Why Use the TWS Java API?

OK! Now that you are familiar with TWS and what it can do, we can move on to the amazing API. If you actually read the last chapter, you might be thinking to yourself "Why would I want to use an API when TWS seems to do everything." Or you could be thinking "Hmmmm, I wonder if TWS can… fill in the blank?" OK, if you're asking the first question, I'll explain why you might need the API, and if you're asking the second, it's actually the API that can fill in the blank.

TWS has the capability to do tons of different things, but it does them in a certain way and displays results in a certain way. It's likely that our development team, as fantastic as they are, hasn't yet exhausted the number of features and way of implementing them that all of you collectively can devise. So it's very likely that you, with your unique way of thinking, will be or have been inspired by the power of TWS to say something like "Holy moly, I can't believe I can really do all of this with TWS! Now if I could only just (fill in the blank),my life would be complete!"

That's where the API comes in. Now, you can fill in the blank! It's going to take a little work to get there, but once you see how cool it is to be able to access functionality from one application to another, you'll be hooked.

TWS and the API

In addition to allowing you pretty much free reign to create new things and piece together existing things in new ways, the API is also a great way to automate your tasks. You use the API to harness the power behind TWS - in different ways.

Here's an analogy that might help you understand the relationship between TWS and the API. Start by imagining TWS as a book (since TWS is constantly being enhanced, our analogy imagines a static snapshot of TWS at a specific point in time). It's the reference book you were looking for, filled with interesting and useful information, a book with a beginning, middle and end, which follows a certain train of logic. You could skip certain chapters, read Chapter 10 first and Chapter 2 last, but it's still a book. Now imagine, in comparison, that the API is the word processing program in which the book was created with the text of the book right there. This allows you access to everything in the book, and most importantly, it lets you continually change and update material, and automate any tasks that you'd have to perform manually using just a book, like finding an index reference or going to a specific page from the table of contents.

The API works in conjunction with TWS and with the processing functions that run behind TWS, including IB's SmartRouting, high-speed order transmission and execution, support for over 40 orders types, etc. TWS accesses this functionality in a certain way, and you can design your API to take advantage of it in other ways.

Available API Technologies

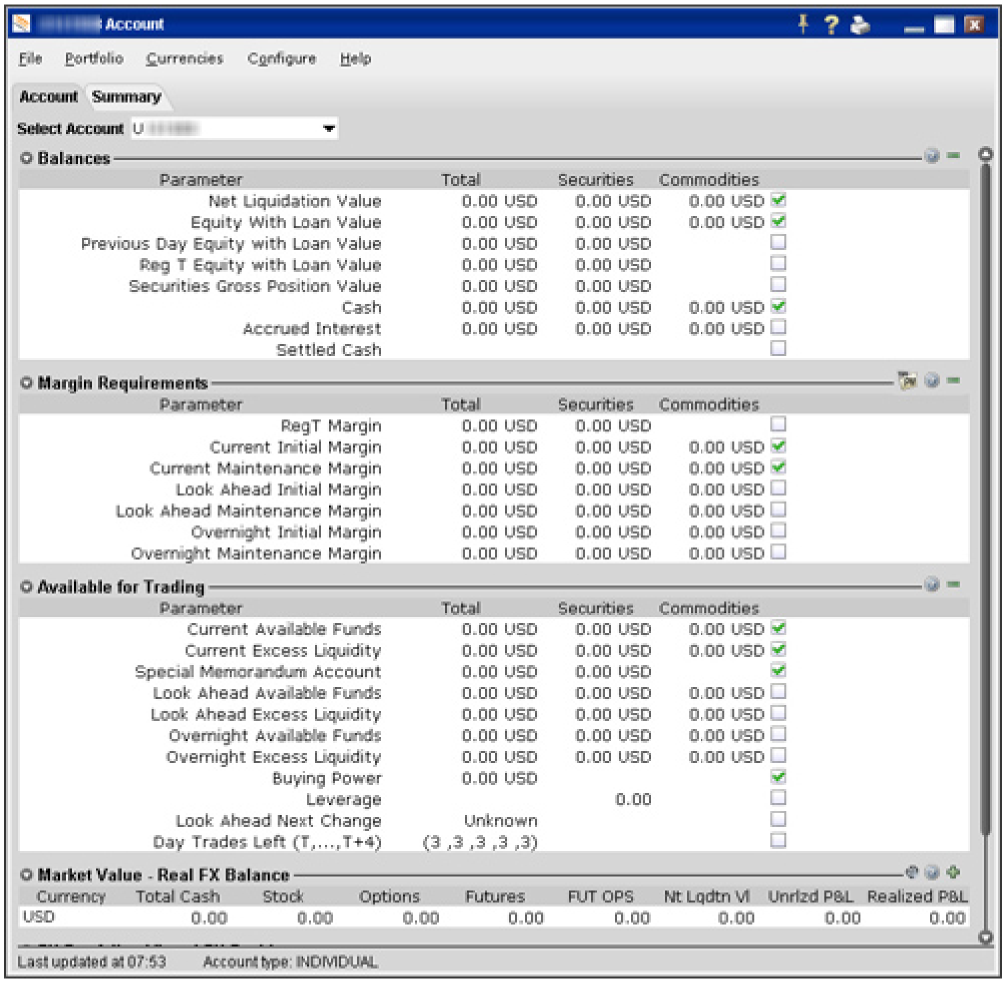
IB provides a suite of custom APIs in multiple programming languages, all to the same end. These include Java, C++, Active X for Visual Basic and .NET, and DDE for Excel (Visual Basic for Applications, of VBA). This book focuses specifically on just one, the Java version. Why would you use Java over the other API technologies? The main reason might be that you are a Java expert. If you don't know Java or any other programming language, you should take a look at the Excel/DDE API, which has a much smaller learning curve. But if you know Java, this platform offers more flexibility than the DDE for Excel, is supported on Windows, MAC, and Unix/Linux (the DDE is only supported in Windows), and provides very high performance.

For more information about our APIs, see the Application Programming Interfaces page on our web site.

An Example

It's always easier to understand something when you have a real life example to contemplate. What follows is a simple situation in which the API could be used to create a custom result.

TWS provides an optional field that shows you your position-specific P&L for the day as either a percentage or an absolute value. Suppose you want to modify your position based on your P&L value? At this writing, the only way to do this would be to watch the market data line to see if the P&L changed, and then manually create and transmit an order, but only if you happened to catch the value at the right point. Hmmmmm, I don't think so! Now, enter the API! You can instruct the API to automatically trigger an order with specific parameters (such as limit price and quantity) when the P&L hits a certain point. Now that's power! Another nice benefit of the API is that it gives you the ability to use the data in TWS in different ways. We know that TWS provides an extensive Account Information window that's chock-full of everything you'll ever want to know about your account status. The thing is, it's only displayed in a TWS window, The thing is, it's only displayed in a TWS window, like this:


[redacted] Account
File Portfolio Currencies Configure Help
Account Summary
Select Account U [redacted]

Balances

Parameter	Total	Securities	Commodities	
Net Liquidation Value	0.00 USD	0.00 USD	0.00 USD	✓
Equity With Loan Value	0.00 USD	0.00 USD	0.00 USD	✓
Previous Day Equity with Loan Value	0.00 USD	0.00 USD		☐
Reg T Equity with Loan Value	0.00 USD	0.00 USD		☐
Securities Gross Position Value	0.00 USD	0.00 USD		☐
Cash	0.00 USD	0.00 USD	0.00 USD	✓
Accrued Interest	0.00 USD	0.00 USD	0.00 USD	☐
Settled Cash				☐

Margin Requirements

Parameter	Total	Securities	Commodities	
RegT Margin	0.00 USD	0.00 USD		☐
Current Initial Margin	0.00 USD	0.00 USD	0.00 USD	✓
Current Maintenance Margin	0.00 USD	0.00 USD	0.00 USD	✓
Look Ahead Initial Margin	0.00 USD	0.00 USD	0.00 USD	☐
Look Ahead Maintenance Margin	0.00 USD	0.00 USD	0.00 USD	☐
Overnight Initial Margin	0.00 USD	0.00 USD	0.00 USD	☐
Overnight Maintenance Margin	0.00 USD	0.00 USD	0.00 USD	☐

Available for Trading

Parameter	Total	Securities	Commodities	
Current Available Funds	0.00 USD	0.00 USD	0.00 USD	✓
Current Excess Liquidity	0.00 USD	0.00 USD	0.00 USD	✓
Special Memorandum Account	0.00 USD	0.00 USD		✓
Look Ahead Available Funds	0.00 USD	0.00 USD	0.00 USD	☐
Look Ahead Excess Liquidity	0.00 USD	0.00 USD	0.00 USD	☐
Overnight Available Funds	0.00 USD	0.00 USD	0.00 USD	☐
Overnight Excess Liquidity	0.00 USD	0.00 USD	0.00 USD	☐
Buying Power	0.00 USD			✓
Leverage		0.00		☐
Look Ahead Next Change	Unknown			☐
Day Trades Left (T,...,T+4)	(3 ,3 ,3 ,3 ,3)			☐

Market Value - Real FX Balance

Currency	Total Cash	Stock	Options	Futures	FUT OPS	Nt Lqdtn Vl	Unrlzd P&L	Realized P&L
USD	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Last updated at 07:53 Account type: INDIVIDUAL

Lovely though it is, what if you wanted to do something else with this information? What if you want it reflected in some kind of banking spreadsheet where you log information for all accounts that you own, including your checking account, Interactive Brokers' account, 401K, ROIs, etc? Again - enter the API!

You can instruct the API to get any specific account information and put it wherever it belongs in a spreadsheet. The information is linked to TWS, so it's easy to keep the information updated by simply linking to a running version of TWS. With a little experimenting, and some help from the *API Reference Guide* and the *TWS Users' Guide*, you'll be slinging data like a short-order API chef in no time!

There are a few other things you must do before you can start working with the TWS Java API. The next chapter gets you geared up and ready to go.

3

Preparing to Use the Java API

Although the API provides great flexibility in implementing your automated trading ideas, all of its functionality runs through TWS. This means that you must have a TWS account with IB, and that you must have your TWS running in order for the API to work. This section takes you through the minor prep work you will need to complete, step by step.

Here's what you'll find in this chapter:

We want to tell you again that this book focuses on the TWS side of the Java API - we don't really help you to learn Java. Unless you are a fairly proficient Java programmer, or at least a very confident and bold beginner, this may be more than you want to take on. We suggest you start with a beginner's Java programming book, and come back to us when you're comfortable with Java.

Download the Java JDK and IDE

OK, well we've already said that you need to know Java before you can successfully implement your own TWS Java API application, and there's a good chance you already have the Java tools you'll need downloaded and installed. But in case you don't, we'll quickly walk you through what you need:

- The Java development kit (JDK)
- An integrated development environment (IDE).

We like the J2SE Development Kit and NetBeans IDE Bundle that's available (free!) from the Sun website. We're not including any version numbers of these Sun Java products, as they'll likely be different by the time you read this. You can use any IDE you're comfortable with.

In this book we use NetBeans as the IDE of choice, so if you're using another IDE you'll have to reinterpret our instructions to fit your development environment. If you're using NetBeans and aren't totally familiar with it, we recommend browsing through the Quick Start or the tutorial, both of which are available on the Help menu.

Anyway, I know we're not giving you too much here, but we are assuming you have enough savvy to find this stuff, download it, and install it. This is a tough line for us to walk, because we're really focusing on the TWS Java API for beginners, not on Java for beginners. If you're having trouble at this point, you should probably start with the TWS DDE for Excel API to get your feet wet!

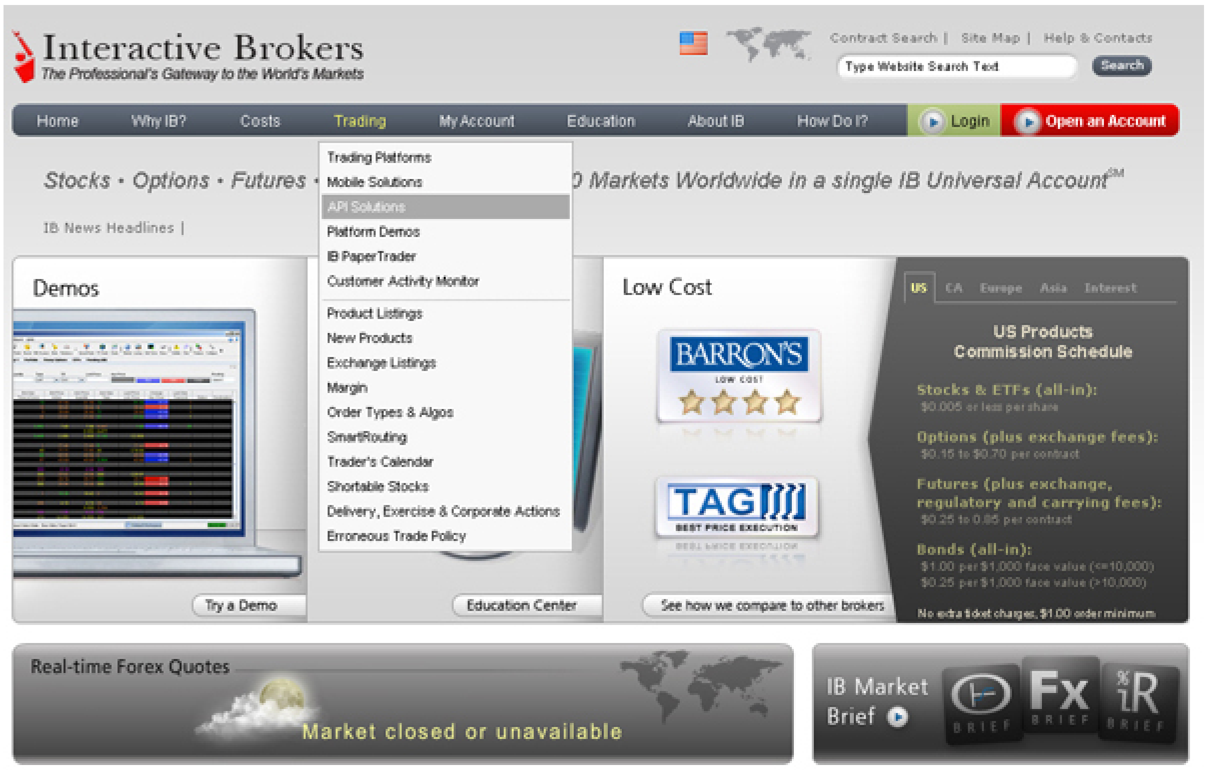
Once you have these pieces downloaded and installed, you can go to the IB website and download the TWS API software.

Download the API Software

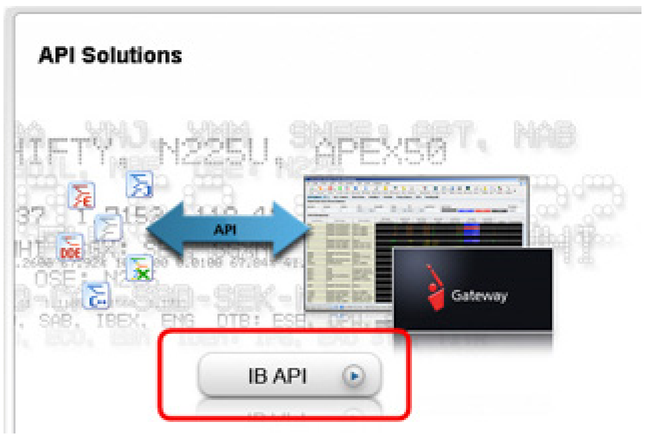
Next, you need to download the API software from the IB website.

Step 1: Download the API software.

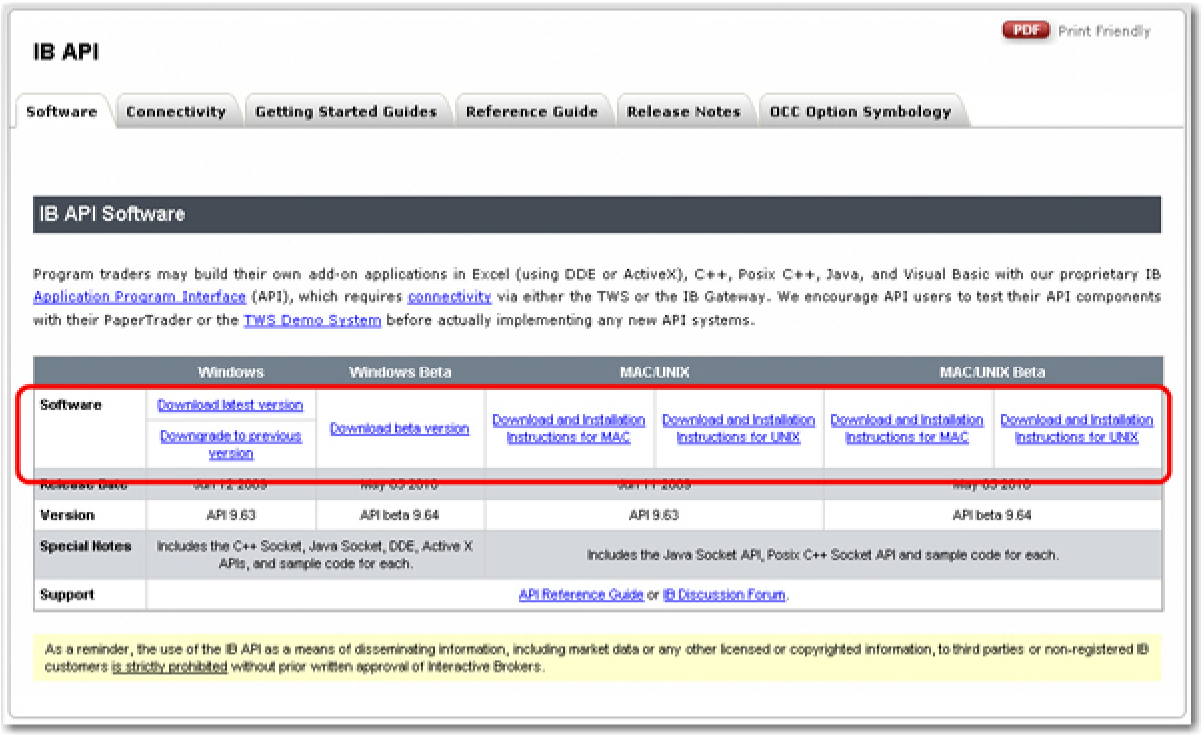
This step takes you out to the IB website at http://individuals.interactivebrokers.com/en/p.php?f=programInterface&p=a&ib_entity=lic. The menus are along the top of the homepage. Hold your mouse pointer over the Trading menu, then click *API Solutions*.


Interactive Brokers
The Professional's Gateway to the World's Markets
Contract Search | Site Map | Help & Contacts
Type Website Search Text
Search
Home
Why IB?
Costs
Trading
My Account
Education
About IB
How Do I?
Login
Open an Account
Trading Platforms
Mobile Solutions
API Solutions
Platform Demos
IB PaperTrader
Customer Activity Monitor
Product Listings
New Products
Exchange Listings
Margin
Order Types & Algos
SmartRouting
Trader's Calendar
Shortable Stocks
Delivery, Exercise & Corporate Actions
Erroneous Trade Policy
Stocks • Options • Futures •
0 Markets Worldwide in a single IB Universal Account
IB News Headlines |
Demos
Try a Demo
Education Center
Low Cost
BARRON'S
TAG
See how we compare to other brokers
US CA Europe Asia Interest
US Products Commission Schedule
Stocks & ETFs (all-in):
$0.005 or less per share
Options (plus exchange fees):
$0.15 to $0.70 per contract
Futures (plus exchange, regulatory and carrying fees):
$0.25 to 0.85 per contract
Bonds (all-in):
$1.00 per $1,000 face value (<=10,000)
$0.25 per $1,000 face value (>10,000)
No extra ticket charges, $1.00 order minimum
Real-time Forex Quotes
Market closed or unavailable
IB Market Brief

On the API Solutions page, click the **IB API** button on the left side of the page.



This displays the IB API page which shows a table with links to software downloads that are compatible with Windows, MAC or Unix platforms. When available, there will also be a Windows Beta version of the software. Look across the top of the table and find the OS you need.



IB API

PDF Print Friendly

Software | Connectivity | Getting Started Guides | Reference Guide | Release Notes | OCC Option Symbology

IB API Software

Program traders may build their own add-on applications in Excel (using DDE or ActiveX), C++, Posix C++, Java, and Visual Basic with our proprietary IB Application Program Interface (API), which requires connectivity via either the TWS or the IB Gateway. We encourage API users to test their API components with their PaperTrader or the TWS Demo System before actually implementing any new API systems.

	Windows	Windows Beta	MAC/UNIX		MAC/UNIX Beta	
Software	Download latest version Downgrade to previous version	Download beta version	Download and Installation Instructions for MAC	Download and Installation Instructions for UNIX	Download and Installation Instructions for MAC	Download and Installation Instructions for UNIX
Release Date	Jun 12 2009	May 03 2010	Jun 11 2009		May 03 2010	
Version	API 9.63	API beta 9.64	API 9.63		API beta 9.64	
Special Notes	Includes the C++ Socket, Java Socket, DDE, Active X APIs, and sample code for each.		Includes the Java Socket API, Posix C++ Socket API and sample code for each.			
Support	API Reference Guide or IB Discussion Forum.					

As a reminder, the use of the IB API as a means of disseminating information, including market data or any other licensed or copyrighted information, to third parties or non-registered IB customers is strictly prohibited without prior written approval of Interactive Brokers.

For this book, we assume that you are using Windows. If you're using a different operating system (Mac, Unix), be sure to adjust the instructions accordingly!

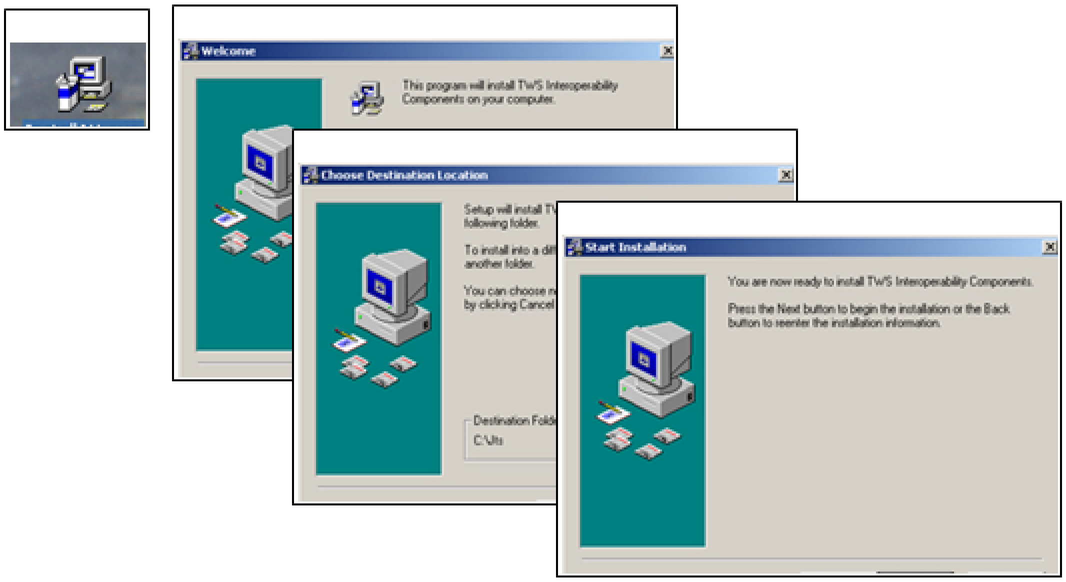
In the Windows column, click *Download Latest Version*. This opens a File Download box, where you can decide whether to save the installation file, or open it. We recommend you choose *Save* and then select a place where you can easily find it, like your desktop (you choose the path in the Save in field at the top of the Save As box that opens up). Once you've selected a good place to put it, click the **Save** button. It takes seconds to download the executable file. Note that the API installation file is named for the API version; for example, InstallAX_960.

We'll usually be stressing just the opposite, but at this point, you need to make sure TWS is not running. If it is, you won't be able to install the API software.

Step 2: Install the API software.

Next, go to the place where you saved the file (for example, your desktop or some other location on your computer), and double-click the API software installation file icon. This starts the installation wizard, a simple process that displays a series of dialogs with questions that you must answer.

Remember where the installation wizard installs the application. You'll need this information later when you open the API application in Excel.



This program will install TWS Interoperability Components on your computer.

Setup will install TV
following folder.

To install into a dif
another folder.

You can choose n
by clicking Cancel

Destination Folde
C:\Jts

You are now ready to install TWS Interoperability Components.

Press the Next button to begin the installation or the Back button to reenter the installation information.

Once you have completed the installation wizard, the sample application installs, and you're ready to open the Java Test Client, connect to TWS, and get started using the Java API sample application!

Connect to the Java Test Client

OK, you've got all the pieces in place. Now that we're done with the prep work, it's time to get down to the fun stuff.

Although the API provides great flexibility in implementing your automated trading ideas, all of its functionality runs through TWS. This means that you must have a TWS account with IB, and you must have TWS running in order for the API to work. This section describes how to enable TWS to connect to the Java API. Note that if you don't have an account with IB, you can use the Demo TWS system to check things out.. If you DO have an account, we recommend opening a linked PaperTrader test account, which simulates the TWS trading environment, and gives you $100,000 in phantom cash to play with.

Enabling TWS to support the API is probably the simplest step you'll encounter in this book. It's probably more difficult to actually remember to log into TWS before you run the API!

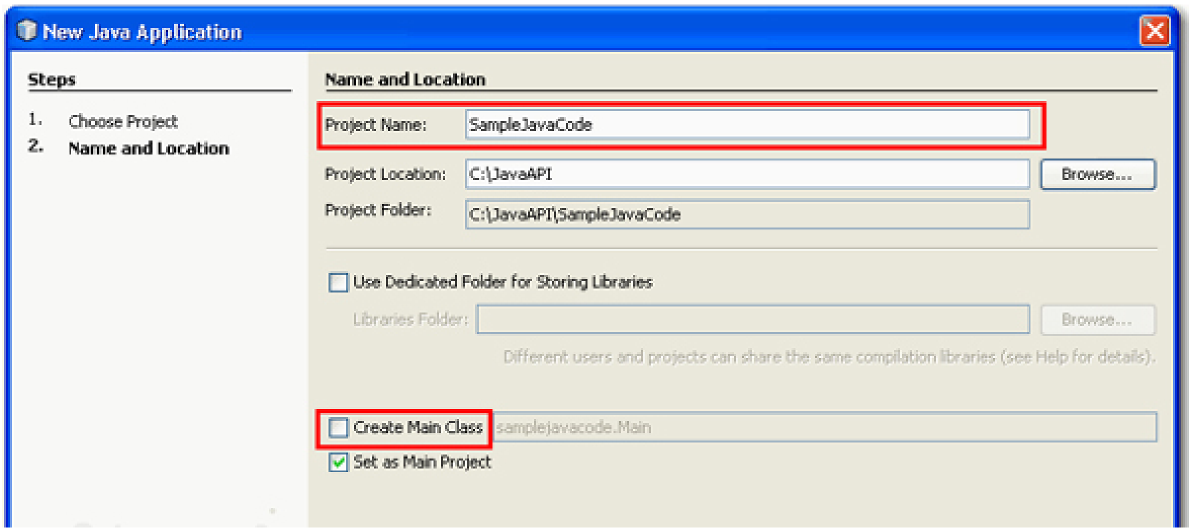
Step 1: Log into TWS.

OK, log into TWS, or run the Demo available on the **Demo** tab of the Trader Workstation page on our website.

Now look up at the top of the trading window, and you'll see the menu bar. Click the **Edit** menu, and then click *Global Configuration*. In the Configuration window, click *API* in the left pane, then click *Settings*, which reveals several options on the right side of the window. Check the *Enable ActiveX and Socket Clients* check box and click **OK**.

Step 2: Set Up the Java Test Client.

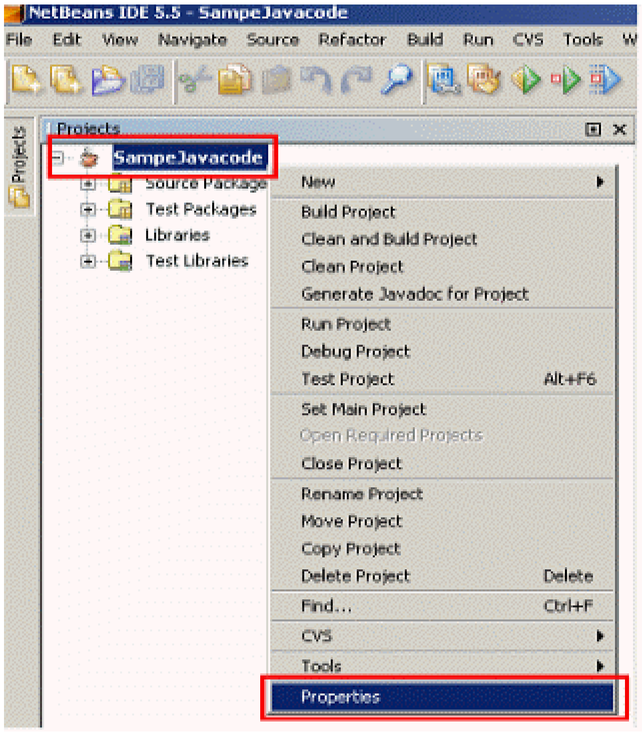
Now, open NetBeans and click *New Project*. This starts the project wizard. In the Projects area, select *Java Application* and click **Next**. You'll see a screen like the one below.



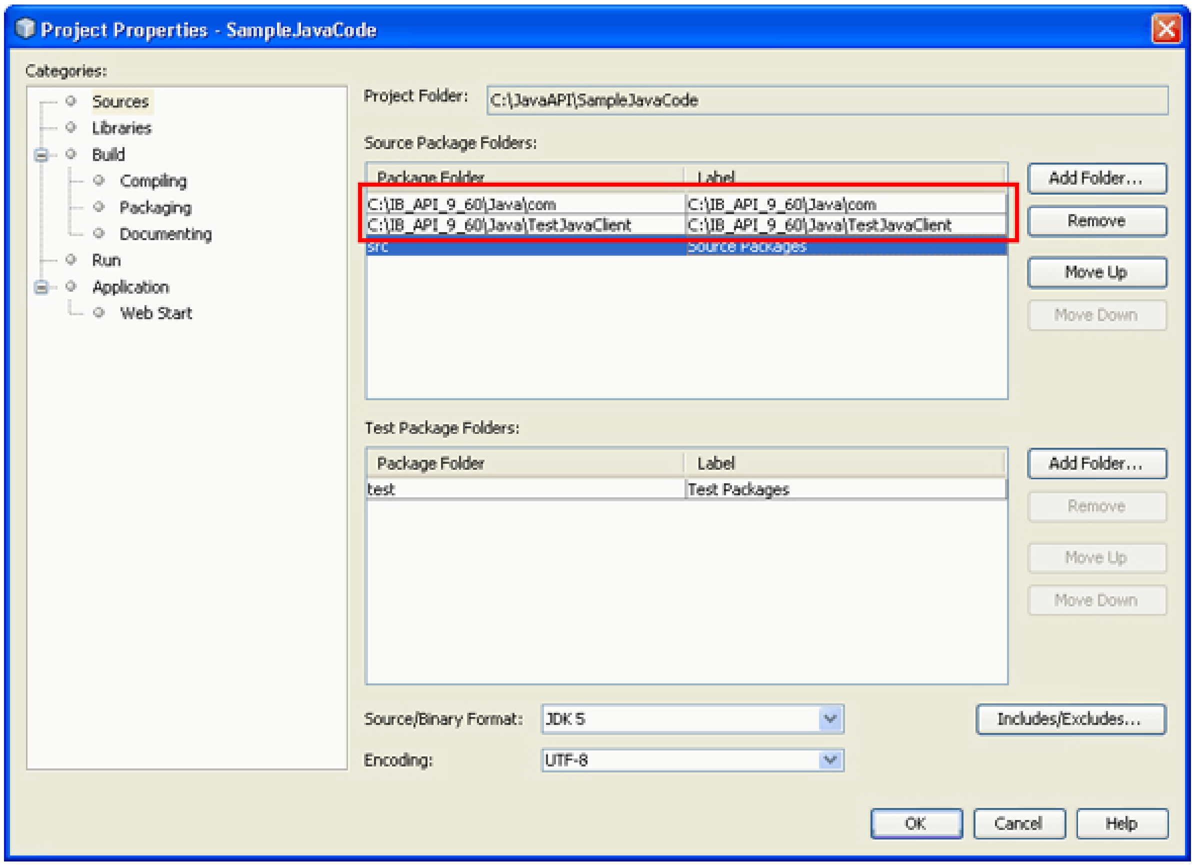
Project Name: SampleJavaCode
Project Location: C:\JavaAPI
Project Folder: C:\JavaAPI\SampleJavaCode
Use Dedicated Folder for Storing Libraries
Libraries Folder:
Different users and projects can share the same compilation libraries (see Help for details).
Create Main Class samplejavacode.Main
Set as Main Project

Enter a project name and project location. Uncheck the box for *Create Main Class* and click **Finish**.

Now right-click your new *SampleJavaCode* project from the *Projects* list and select *Properties*.



In the Source Package Folders area, click **Add Folder** and navigate to the directory where you installed the API sample program. Add two folders: the com folder and the *TestJavaClient* folder. Then click **OK**.



Project Folder: C:\JavaAPI\SampleJavaCode

Source Package Folders:

Package Folder	Label
C:\IB_API_9_60\Java\com	C:\IB_API_9_60\Java\com
C:\IB_API_9_60\Java\TestJavaClient	C:\IB_API_9_60\Java\TestJavaClient
src	Source Packages

Test Package Folders:

Package Folder	Label
test	Test Packages

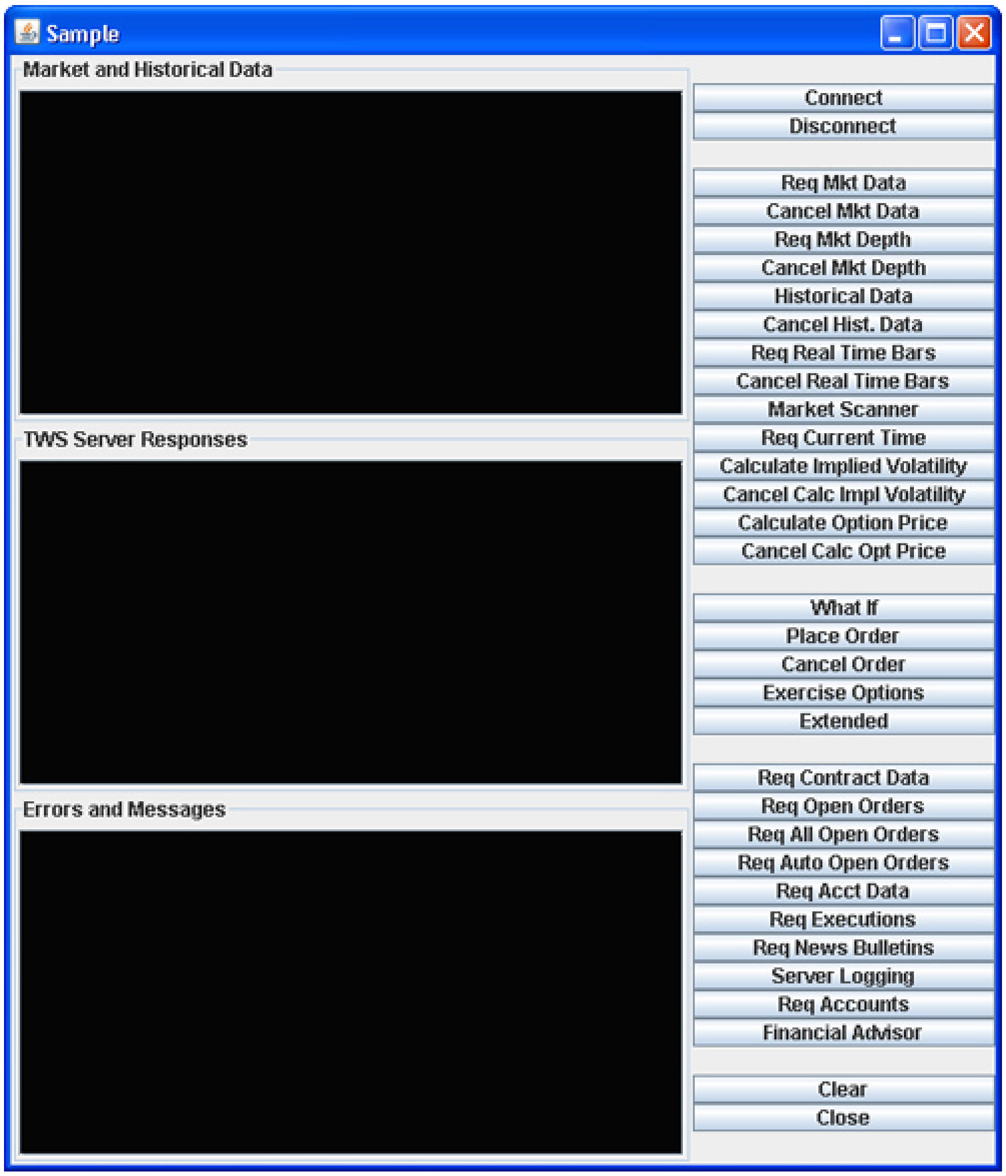
Source/Binary Format: JDK 5

Encoding: UTF-8

Step 3: Run the Java Test Client.

Now it's time to run the application. Press **F6** to run. When the system prompts you to select *TestJavaClient.Main* as the main class, click **OK** (recall that earlier, you had to uncheck the *Create Main Class* box when you first set up the project; now is when you assign the main class). And of course, click **OK** again.

Now press **F6** to run again. You're looking at the java test client, and you should see something like this thing below:



Here you are. What now? Part II focuses on performing the trading tasks defined by the action buttons in the sample client. We'll take a quick, general look at what's going on behind the GUI. Then we'll walk through the basics of the TWS API, in the order defined by the buttons in the Java Test Client layout, pictures above.

The TWS API does not have to be written as a GUI program, but to completely understand how the Java Test Client works, you should have some general understanding of Java Swing. We recommend taking a look at The Swing Tutorial on java.sun.com.

4

The Java API for Financial Advisors

The content in this book applies to financial advisors who handle multiple clients. This chapter focuses on the features in the Java Test Client (and the Java API) that are only available to multi-client account advisors. We discuss the methods and parameters used by the Java Test Client for Financial Advisors (or FA as we like to call it) accounts.

Here's what you'll find in this chapter:

Before you go any further, we feel it's only fair to tell you that the material in this chapter will be meaningful to you only if you actually have a TWS FA account; that is, you should be a professional financial advisor who routinely handles multiple clients in TWS, and is familiar with the financial advisor features in TWS. If you'd like to learn more about TWS's Financial Advisor features, see the TWS Users' Guide, available on our web site.

To learn how to get started using the basic trading features of the TWS Java API, see the Getting Started with the TWS Java API guide.

Viewing Account Data for Managed Accounts

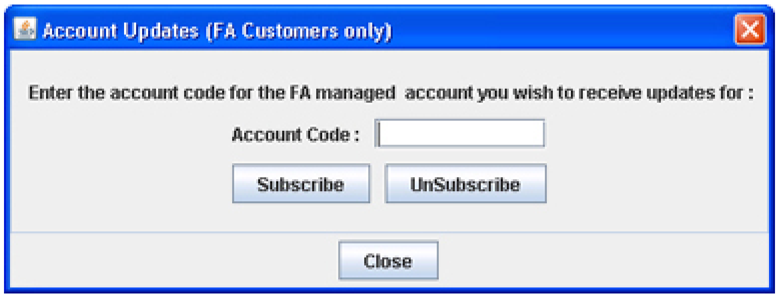
There are three buttons on the Java Test Client that apply to FA accounts:

- **Req Acct Data**: This button lets you get account information for one of the accounts that you manage.
- **Req Accounts**: This button displays a list of account codes for all of the accounts that you manage.
- **Financial Advisor**: This button lets you view/manage your FA Allocation Account Groups, Allocation Profiles, and Account Aliases.

Viewing Account Data for a Managed Account

The **Req Acct Data** button in the sample application lets you view account information for one of your managed accounts.

So, what happens when you click this button? As you might have expected, the attached Action Listener in SampleFrame.java calls the **onReqAcctData()** method, also in SampleFrame.java, and displays the Account Updates (FA Customers only) dialog, which is shown below.


Account Updates (FA Customers only)

Enter the account code for the FA managed account you wish to receive updates for :

Account Code :

As it says in the dialog, you simply type the account code for the managed account information you want to view in the *Account Code* box, then click **Subscribe**.

onReqAcctData() Method in SampleFrame.java

```
void  onReqAcctData() {
   AcctUpdatesDlg dlg = new AcctUpdatesDlg(this);
   dlg.show();
   m_client.reqAccountUpdates( dlg.m_subscribe, dlg.m_acctCode);
   if ( m_client.isConnected() && dlg.m_subscribe) {
      m_acctDlg.reset();
      m_acctDlg.setVisible(true);
   }
}
```

When you click **Subscribe**, **onReqAcctData()** calls the Java API EClientSocket **reqAccountUpdates()** method.

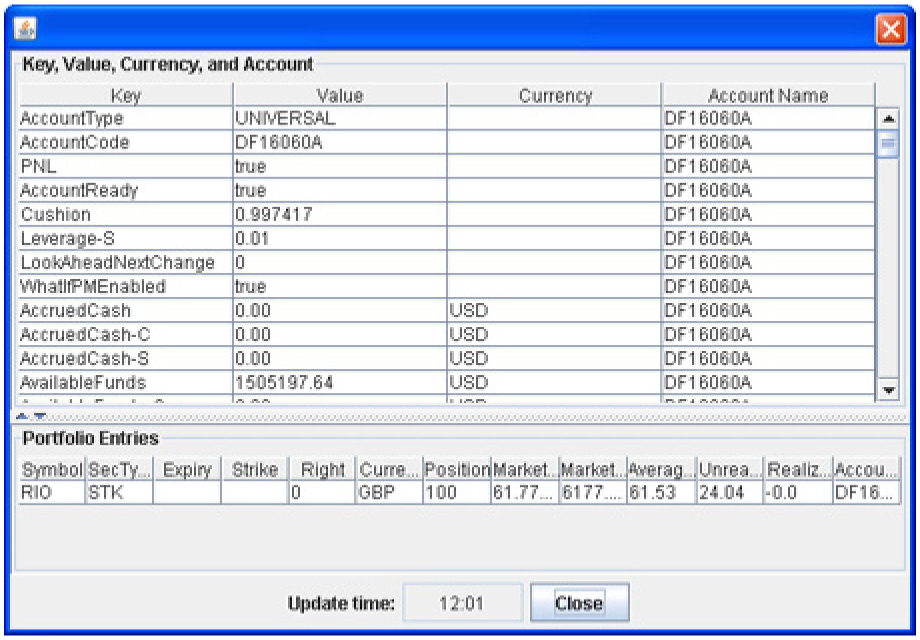
The reqAccountUpdates() Method

```
public synchronized void reqAccountUpdates(boolean subscribe,
String acctCode)
```

The **reqAccountUpdates()** EClientSocket method has two parameters that correspond to your actions in the Account Updates dialog:

- *subscribe* - If set to TRUE, the Java Test Client will start receiving account and portfolio updates. If set to FALSE, the client will stop receiving this information. This parameter corresponds to you clicking either the **Subscribe** button (parameter is set to TRUE), or the **UnSubscribe** button (parameter is set to FALSE).
- *acctCode* - This parameter holds the account code of the managed account for which you want to display account and portfolio information (the account code you typed in the Account Updates dialog in the Java Test Client).

Back in the sample application, once you click **Subscribe**, the information you requested is displayed in its own dialog (shown below). The data includes account values, portfolio, and last update time information for the specified account code.



Key, Value, Currency, and Account

Key	Value	Currency	Account Name
AccountType	UNIVERSAL		DF16060A
AccountCode	DF16060A		DF16060A
PNL	true		DF16060A
AccountReady	true		DF16060A
Cushion	0.997417		DF16060A
Leverage-S	0.01		DF16060A
LookAheadNextChange	0		DF16060A
WhatIfPMEnabled	true		DF16060A
AccruedCash	0.00	USD	DF16060A
AccruedCash-C	0.00	USD	DF16060A
AccruedCash-S	0.00	USD	DF16060A
AvailableFunds	1505197.64	USD	DF16060A

Portfolio Entries

Symbol	SecTy...	Expiry	Strike	Right	Curre...	Position	Market...	Market...	Averag...	Unrea...	Realiz...	Accou...
RIO	STK			0	GBP	100	61.77...	6177....	61.53	24.04	-0.0	DF16...

Update time: 12:01

This information is received from TWS via the methods **updateAccountTime()**, **updateAccountValue()** and **updatePortfolio()** in the Java API EWrapper interface.

The updateAccountTime() Method

```
void updateAccountTime(String timeStamp)
```

The EWrapper **updateAccountTime()** method returns the last update time for the specified account. This method has just one parameter, *timeStamp*, which indicates the last update time of the account information. This time is displayed at the bottom of the account information window pictured on the previous page.

The updateAccountValue() Method

```
void updateAccountValue(String key, String value, String currency,
String accountName)
```

The EWrapper **updateAccountValue()** method returns the actual account data for the specified account code. The parameters that are returned via this method are listed in the following table.

Parameter	Description
key	A string that indicates one type of account value. There is a long list of possible keys that can be sent, here are just a few examples: • CashBalance - account cash balance • DayTradesRemaining - number of day trades left • EquityWithLoanValue - equity with Loan Value • InitMarginReq - current initial margin requirement • MaintMarginReq - current maintenance margin • NetLiquidation - net liquidation value
value	The value associated with the key.
currency	Defines the currency type, in case the value is a currency type.
account	States the account the message applies to. Useful for Financial Advisor sub-account messages.

Tables are for illustrative purposes only and are not intended to represent valid API information.

The updatePortfolio() Method

```
void updatePortfolio(Contract contract, int position, double
marketPrice, double marketValue, double averageCost, double
unrealizedPNL, double realizedPNL, String accountName)
```

The EWrapper **updatePortfolio()** method returns portfolio data for the specified account code. The parameters that are returned via this method are listed in the following table.

Parameter	Description
contract	This structure contains a description of the contract which is being traded. The exchange field in a contract is not set for portfolio update.
position	This integer indicates the position on the contract. If the position is 0, it means the position has just cleared.
marketPrice	The unit price of the instrument.
marketValue	The total market value of the instrument.
averageCost	The average cost per share is calculated by dividing your cost (execution price + commission) by the quantity of your position.
unrealizedPNL	The difference between the current market value of your open positions and the average cost, or Value - Average Cost.
realizedPNL	Shows your profit on closed positions, which is the difference between your entry execution cost (execution price + commissions to open the position) and exit execution cost ((execution price + commissions to close the position)
accountName	The name of the account the message applies to. Useful for Financial Advisor sub-account messages.

Tables are for illustrative purposes only and are not intended to represent valid API information.

Viewing a List of Managed Accounts

The **Req Accounts** button lets you view a list of account codes for all of the accounts that you manage.

When you click this button, the attached Action Listener in SampleFrame.java calls the **onReqManagedAccts()** method, also in SampleFrame.java, and calls the EClientSocket method **reqManagedAccts()**.

onReqManagedAccts() Method in SampleFrame.java

```
void onReqManagedAccts() {
   // request the list of managed accounts
   m_client.reqManagedAccts();
}
```

The reqManagedAccts() Method

```
public synchronized void reqManagedAccts()
```

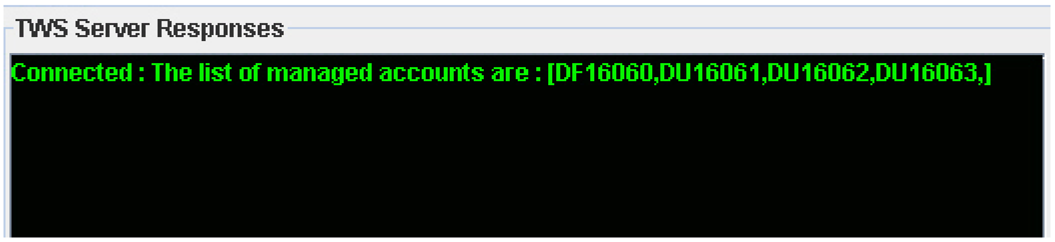
The EClientSocket r**eqManagedAccts()** method requests a list of managed accounts. It has no parameters. The list is returned by the method **managedAccounts()** in the Java API EWrapper interface.

The managedAccounts() Method

```
void managedAccounts(String accountsList)
```

The EWrapper **managedAccounts()** method has one parameter, *accountsList*, that returns the list of managed accounts. Note that this method is also called when a successful connection is made to a TWS FA account from the Java Test Client.

The list of accounts is displayed in the *TWS Server Responses* text panel in the sample application, as shown below.



```
Connected : The list of managed accounts are : [DF16060,DU16061,DU16062,DU16063,]
```

Viewing and Modifying Configuration Information

The **Financial Advisor** button lets you view and modify your FA configuration information in the Java Test Client sample application, including:

- FA Account Groups - These are groups of accounts and an associated allocation method that you create in TWS.
- FA Allocation Profiles - These are profiles you create in TWS that distribute shares to each account based on a percentage, ratio or absolute number.
- FA Account Aliases - These are recognizable names that you assign to accounts in TWS.

So, what happens when you click the **Financial Advisor** button? Well of course, the attached Action Listener in SampleFrame.java calls the method **onFinancialAdvisor()**, also in SampleFrame.java.

onFinancialAdvisor() Method in SampleFrame.java

```
void onFinancialAdvisor() {
   faGroupXML = faProfilesXML = faAliasesXML = null ;
   faError = false ;
   m_client.requestFA(EClientSocket.GROUPS) ;
   m_client.requestFA(EClientSocket.PROFILES) ;
   m_client.requestFA(EClientSocket.ALIASES) ;
}
```

The requestFA() Method

```
public synchronized void requestFA( int faDataType )
```

onFinancialAdvisor() calls the EClientSocket method **requestFA()**, which contains a single parameter, *faDataType*. This parameter specifies the type of Financial Advisor configuration information to return:

- 1 = GROUPS
- 2 = PROFILE
- 3 = ACCOUNT ALIASES

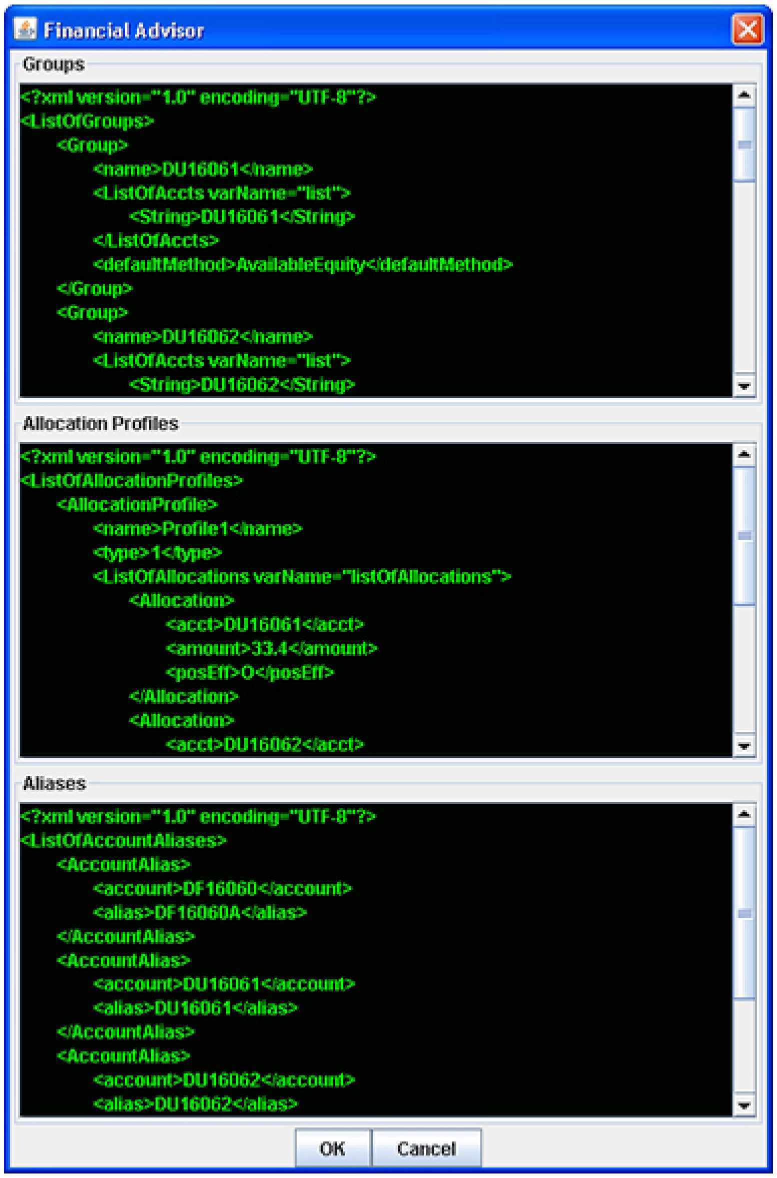
These values correspond to the three main pieces of FA information described earlier in this section.

The receiveFA() Method

```
void receiveFA(int faDataType, String xml)
```

The actual FA configuration information is returned as XML strings from the **receiveFA()** method in the Java API EWrapper interface. The **receiveFA()** method has two parameters, *fadataType*, which specifies the type of Financial Advisor configuration information to receive (1 = GROUPS, 2 = PROFILE, 3 = ACCOUNT ALIASES), and *xml*, which is the XML string containing the FA configuration information saved on the TWS server.

Finally, the requested FA configuration information is displayed in the Financial Advisor dialog shown below. This dialog has three text panels that display the three XML strings returned.



```
<?xml version="1.0" encoding="UTF-8"?>
<ListOfGroups>
    <Group>
        <name>DU16061</name>
        <ListOfAccts varName="list">
            <String>DU16061</String>
        </ListOfAccts>
        <defaultMethod>AvailableEquity</defaultMethod>
    </Group>
    <Group>
        <name>DU16062</name>
        <ListOfAccts varName="list">
            <String>DU16062</String>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<ListOfAllocationProfiles>
    <AllocationProfile>
        <name>Profile1</name>
        <type>1</type>
        <ListOfAllocations varName="listOfAllocations">
            <Allocation>
                <acct>DU16061</acct>
                <amount>33.4</amount>
                <posEff>O</posEff>
            </Allocation>
            <Allocation>
                <acct>DU16062</acct>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<ListOfAccountAliases>
    <AccountAlias>
        <account>DF16060</account>
        <alias>DF16060A</alias>
    </AccountAlias>
    <AccountAlias>
        <account>DU16061</account>
        <alias>DU16061</alias>
    </AccountAlias>
    <AccountAlias>
        <account>DU16062</account>
        <alias>DU16062</alias>
```

Keep in mind, all this action occurs behind the scenes and the dialog appears immediately.

How Do I Modify the FA Configuration Information in the Java Test Client?

Once you have the FA configuration information displayed in the Financial Advisor dialog shown on the previous page, you can modify the XML strings directly in the text panel. Simply click in a text panel and modify the information. When you click the **OK** button after making whatever changes you require, your new FA configuration is sent back to TWS.

So how does this work behind the scenes?

When you click the **OK** button after making your changes to the FA configuration information in the Java Test Client's Financial Advisor dialog, **onFinancialAdvisor()** in SampleFrame.java calls the EClientSocket method **replaceFA().** In fact, even if you don't make any changes to the FA configuration information in the Financial Advisor dialog, **replaceFA()** is STILL called!

The replaceFA() Method

The **replaceFA()** method basically takes the configuration information in the Financial Advisor dialog, whether you've changed it or not, and sends it along to TWS as "new" configuration information (hence the name "replaceFA"). This information is saved on the TWS server. So, whether or not you actually make changes in the Financial Advisor dialog, the FA configuration information displayed as XML strings is sent to the TWS server via **replaceFA()**. Whenever you view this information in the Java Test Client, the XML strings are returned via the EWrapper method **receiveFA().** Thus, the XML strings are being "updated" via replaceFA() every time you click **OK** in the Financial Advisor dialog.

```
public synchronized void replaceFA( int faDataType, String xml )
```

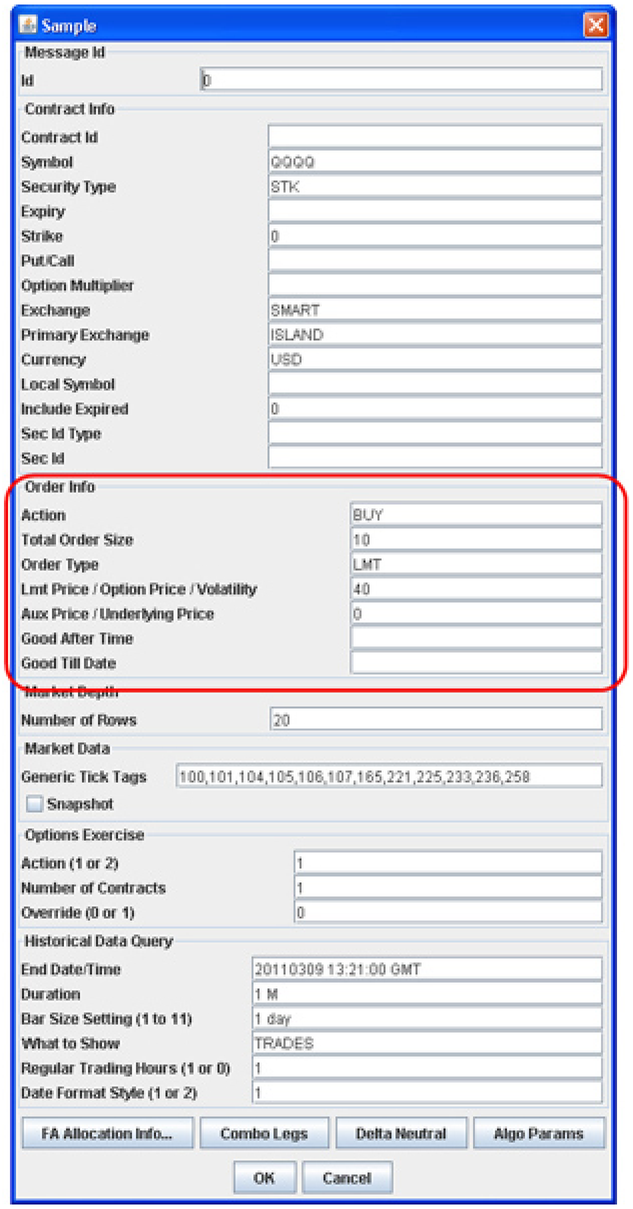
The **replaceFA()** method, whose header is shown above, has two parameters, *fadataType*, which specifies the type of Financial Advisor configuration information to receive (1 = GROUPS, 2 = PROFILE, 3 = ACCOUNT ALIASES), and *xml*, which is the XML string containing the FA configuration information. Yes, these are the same parameters that are in receiveFA().

If you know how to write XML, you can actually create your FA Groups, Allocation Profiles and Account Aliases directly in the Java Test Client's Financial Advisor dialog. However, it is much easier to set up your FA configuration in TWS first.

Placing Orders Using FA Accounts

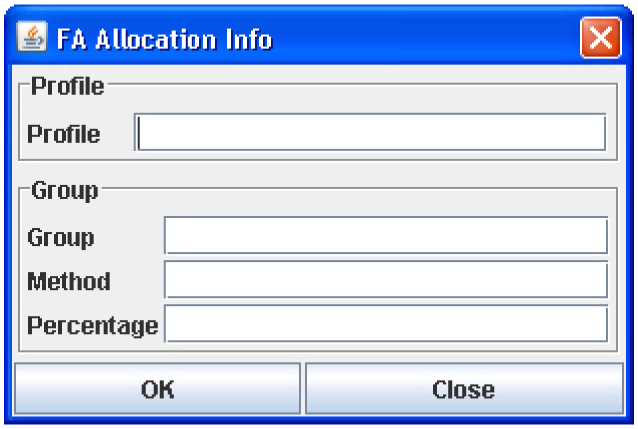
In this section, we take a look at how to set up an order for an advisor using a single account group, an account group or an allocation profile. When placing an order as an advisor, you'll use the same order fields in the same Order dialog pictured below and described back in Chapter 13. If you are an advisor however, there are a couple of extra steps you take in order to enter your allocation profile or account group information.

To place an order using a single account, an account group, and an allocation profile in the Java Test Client, first click the **Place Order** button. In the Sample dialog, enter the contract information in the *Contract Info* section and the order information in the *Order Info* section, as shown below.



Message Id	
Id	0
Contract Info	
Contract Id	
Symbol	QQQQ
Security Type	STK
Expiry	
Strike	0
Put/Call	
Option Multiplier	
Exchange	SMART
Primary Exchange	ISLAND
Currency	USD
Local Symbol	
Include Expired	0
Sec Id Type	
Sec Id	
Order Info	
Action	BUY
Total Order Size	10
Order Type	LMT
Lmt Price / Option Price / Volatility	40
Aux Price / Underlying Price	0
Good After Time	
Good Till Date	
Market Depth	
Number of Rows	20
Market Data	
Generic Tick Tags	100,101,104,105,106,107,165,221,225,233,236,258
Snapshot	☐
Options Exercise	
Action (1 or 2)	1
Number of Contracts	1
Override (0 or 1)	0
Historical Data Query	
End Date/Time	20110309 13:21:00 GMT
Duration	1 M
Bar Size Setting (1 to 11)	1 day
What to Show	TRADES
Regular Trading Hours (1 or 0)	1
Date Format Style (1 or 2)	1

This is where placing orders as an advisor is different from placing orders for a normal account. Next, click the **FA Allocation Info...** button at the bottom of the Sample dialog. The FA Allocation Info dialog appears. This is where you will enter your allocation profile or account group information.



In the *Profile* box, type the name of the allocation profile you want to use for this order.

- If you enter an allocation profile, you don't have to enter the group information in the *Group* section of the dialog.
- If, however, you want to place the order for a specific account group, type the name of the account group, method and percentage if required by the method in the appropriate boxes.
- If you enter a *Profile*, you don't have to enter the *Group* information; conversely, if you fill in the *Group* fields, you don't have to enter a *Profile*. When you have filled in the correct boxes, click **OK**.

Click **OK** to close the Sample dialog and place your order. You can always check any open orders and executions in the sample application by clicking the appropriate button.

That's all folks! Of course, you can always refer to Chapter 5 for more details and examples on allocation methods and profiles.

For detailed information about setting up Financial Advisor Allocation Profiles, Account Groups and Account Aliases, see the TWS Users' Guide, available on our web site, or use the online help available from within TWS itself.

Allocation Methods and Profiles

This chapter describes the allocation methods and allocation profiles available to Financial Advisors in Trader Workstation and the API.

Here's what you'll find in this chapter:

Allocation Methods for Account Groups

Note that you must type the method name in exactly as appears here, or your order won't work.

EqualQuantity Method

Requires you to specify an order size. This method distributes shares equally between all accounts in the group.

Example: You transmit an order for 400 shares of stock ABC. If your Account Group includes four accounts, each account receives 100 shares. If your Account Group includes six accounts, each account receives 66 shares, and then 1 share is allocated to each account until all are distributed.

NetLiq Method

Requires you to specify an order size. This method distributes shares based on the net liquidation value of each account. The system calculates ratios based on the Net Liquidation value in each account and allocates shares based on these ratios.

Example: You transmit an order for 700 shares of stock XYZ. The account group includes three accounts, A, B and C with Net Liquidation values of $25,000, $50,000 and $100,000 respectively. The system calculates a ratio of 1:2:4 and allocates 100 shares to Client A, 200 shares to Client B, and 400 shares to Client C.

AvailableEquity Method

Requires you to specify an order size. This method distributes shares based on the amount of equity with loan value currently available in each account. The system calculates ratios based on the Equity with Loan value in each account and allocates shares based on these ratios.

Example: You transmit an order for 700 shares of stock XYZ. The account group includes three accounts, A, B and C with available equity in the amounts of $25,000, $50,000 and $100,000 respectively. The system calculates a ratio of 1:2:4 and allocates 100 shares to Client A, 200 shares to Client B, and 400 shares to Client C.

PctChange Method

This method only works when you already hold a position in the selected instrument. Do not specify an order size. Since the quantity is calculated by the system, the order size is displayed in the Quantity field after the order is acknowledged. This method increases or decreases an already existing position. Positive percents will increase a position, negative percents will decrease a position.

Example 1: Assume that three of the six accounts in this group hold long positions in stock XYZ. Client A has 100 shares, Client B has 400 shares, and Client C has 200 shares. You want to increase their holdings by 50%, so you enter "50" in the percentage field. The system calculates that your order size needs to be equal to 350 shares. It then allocates 50 shares to Client A, 200 shares to Client B, and 100 shares to Client C.

Example 2: You want to close out all long positions for three of the five accounts in a group. You create a sell order and enter "-100" in the Percentage field. The system calculates 100% of each position for every account in the group that holds a position, and sells all shares to close the positions.

These handy charts make it easy to see how negative and positive percent values will affect long and short positions for both buy and sell orders. Phew, that was a mouthful!

BUY ORDER	**Positive Percent**	**Negative Percent**
Long Position	**Increases position**	**No effect**
Short Position	**No effect**	**Decreases position**

SELL ORDER	**Positive Percent**	**Negative Percent**
Long Position	**No effect**	**Decreases position**
Short Position	**Increases position**	**No effect**

Allocation Profiles

Percentages

This method will split the total number of shares in the order between listed accounts based on the percentages you indicate.

Example: An order for 1000 shares using a profile with four accounts at 25% each would allocate 250 shares to each listed account in the profile.

Ratios

This method calculates the allocation of shares to the listed accounts based on the ratios you indicate.

Example: An order for 1000 shares using a profile with four accounts set to a ratio of 4, 2, 1, 1 would allocate 500, 250, 125 and 125 shares to the listed accounts, respectively.

Shares

This method allocates an absolute number of shares to each account listed. If you use this method, you don't need to enter an order quantity. The order size is determined by adding together the number of shares allocated to each account in the profile.

6 Where to Go from Here

If you've come this far and actually read the book, you now have a pretty decent grasp on what the Java API can do, and how to make it do some of the things you want. Now we give you a bit more information about how to link to TWS with our Java API, and we suggest some helpful outside resources you can use to help you move forward.

This chapter contains the following sections:

Linking to TWS using the TWS Java API

If you have the skill and confidence to handle Java on your own, you can build your own Java application to link to TWS, using the following steps as a guide.

1. Import **com.ib.client.*** into your source code file. This is the package that contains the TWS Java API classes and methods.
2. Implement the **EWrapper** interface. This class will receive messages from the socket.
3. Override the following methods:

EWrapper Method	Description
tickPrice()	Handles market data.
tickSize()	
tickOptionComputation()	
tickGeneric()	
tickString()	
tickEFP()	
orderStatus()	Receives order status.
openOrder()	Receives open orders.
error()	Receives error information.
connectionClosed()	Notifies you when TWS terminates the connection.
updateAccountValue()	Receives current account values.
updateAccountTime()	Receives the last time account information was updated.
updatePortfolio()	Receives current portfolio information.
nextValidId()	Receives the next valid order ID upon connection.
contractDetails()	Receives contract information.
contractDetailsEnd()	Identifies the end of a given contract details request.
bondContractDetails()	Receives bond contract information.
exectDetails()	Receives execution report information.
updateMktDepth()	Receives market depth information.
updateMktDepthL2()	Receives Level II market depth information.
updateNewsBulletin()	Receives IB news bulletins.
managedAccounts()	Receives a list of Financial Advisor (FA) managed accounts.

EWrapper Method	Description
receiveFA()	Receives FA configuration information.
historicalData()	Receives historical data results.
scannerParameters()	Receives an XML document that describes the valid parameters of a scanner subscription.
scannerData()	Receives market scanner results.
scannerDataEnd()	Called when the scanner snapshot is received and marks the end of one scan.
realTimeBar()	Receives real-time bars.
currentTime()	Receives the current system time on the server.
fundamentalData()	Receives Reuters global fundamental market data.

4 Instantiate the **EClientSocket** class. This object will be used to send messages to TWS.

5 Call the following EClientSocket methods:

EClientSocket Method	Description
eConnect()	Connects to TWS.
eDisconnect()	Disconnects from TWS.
reqMktData()	Requests market data.
cancelMktData()	Cancels market data.
reqMktDepth()	Requests market depth.
cancelMktDepth()	Cancels market depth.
reqContractDetails()	Requests contract details.
placeOrder()	Places an order.
cancelOrder()	Cancels an order.
reqAccountUpdates()	Requests account values, portfolio, and account update time information.
reqExecutions()	Requests a list of the day’s execution reports.
reqOpenOrders()	Requests a list of current open orders for the requesting client and associates TWS open orders with the client. The association only occurs if the requesting client has a Client ID of 0.
reqAllOpenOrders()	Requests a list of all open orders.

EClientSocket Method	Description
reqAutoOpenOrders()	Automatically associates a new TWS with the client. The association only occurs if the requesting client has a Client ID of 0.
reqNewsBulletin()	Requests IB news bulletins.
cancelNewsBulletins()	Cancels IB news bulletins.
setServerLogLevel()	Sets the level of API request and processing logging.
reqManagedAccts()	Requests a list of Financial Advisor (FA) managed account codes.
requestFA()	Requests FA configuration information from TWS.
replaceFA()	Modifies FA configuration information from the API.
reqScannerParameters()	Requests an XML document that describes the valid parameters of a scanner subscription.
reqScannerSubscription()	Requests market scanner results.
cancelScannerSubscription()	Cancels a scanner subscription.
reqHistoricalData()	Requests historical data.
cancelHistoricalData()	Cancels historical data.
reqRealTimeBars()	Requests real-time bars.
cancelRealTimeBars()	Cancels real-time bars.
exerciseOptions()	Exercises options.
reqCurrentTime()	Requests the current server time.
serverVersion()	Returns the version of the TWS instance to which the API application is connected.
TwsConnectionTime()	Returns the time the API application made a connection to TWS.
reqFundamentalData()	Requests Reuters global fundamental data. There must be a subscription to Reuters Fundamental set up in Account Management before you can receive this data.
cancelFundamentalData()	Cancels Reuters global fundamental data.

Additional Resources

There are many resources out there that will be adequate in getting you where you need to go. If you have some books or places that you like, feel free to stick with them. The following are the resources we find most helpful, and perhaps they'll be good to you, too!

Help with Java Programming

While this book is intended for users with Java programming experience, we understand that even experienced Java programmers need help every once in a while.

The best place to go to find additional help with all things Java is the Sun web site. Just type http://java.sun.com in your browser's address line and check out the list of links under *Resources* on the right side of the page. Sun has many online resources available for Java programmers, including documentation, tutorials, and code samples.

If you simply want to look up information about the actual Java API (as opposed to our TWS Java API), you can go directly to Sun's API Specifications Reference page. There you will find links to documentation, Javadocs, technical articles and a whole host of useful information.

There are literally hundreds of additional printed and web-based resources for Java programmers. We encourage you to investigate these on your own.

Help with the Java API

For help specific to the Java TWS API, the one best place to go, really the ONLY place to go, is the Interactive Brokers website. Once you get there, you have lots of resources. Just type www.interactivebrokers.com in your browser's address line. Now that you're there, let me tell you where you can go.

As of this writing in March of 2011, the IB website looks as we're describing. IB has a tendency to revamp the look and organization of their site every year or two, so have a little patience if it looks slightly different from what's described here.

The API Reference Guide

The API Reference Guide includes sections for each API technology, including the DDE for Excel. The upper level topics which are shown directly below the main book are applicable across the board to all or multiple platforms.

To access the API Reference Guide from the IB web site, select *API Solutions* from the **Trading** menu, then click the **IB API** button, then click the **Reference Guide** tab. Click the **Online API Reference Guide** button to open the online guide, which contains a section devoted entirely to the DDE for Excel API.

The API Beta and API Production Release Notes

The beta notes are in a single page file, and include descriptions of any new additions to the API (all platforms) that haven't yet been pushed to production. The API Release Notes opens an index page that includes links to all of the past years' release notes pages. The index provides one-line titles of all the features included in each release.

To access these notes from the IB web site, select *API Solutions* from the **Trading** menu, then click the **IB API** button, then click the **Release Notes** tab and select a link to the latest API production release notes. You can also access the release notes for the latest API Beta release from this page.

The TWS API Webinars

IB hosts free online webinars through WebEx to help educate their customers and other traders about the IB offerings. They present the API webinar about once per month, and have it recorded on the website for anyone to listen to at any time.

- To register for the API webinar, from the IB web site click **Education**, then select *Webinars*. Click the **Live Webinars** button, then click the **API** tab.
- To view the recorded version of the API webinar, from the **Live Webinars** page click the **Watch Previously Recorded Webinars** button. Links to recorded versions of previously recorded webinars are listed on the page.

API Customer Forums

You can trade ideas and send out pleas for help via the IB customer base accessible through both the IB Bulletin Board and the Traders' Chat. The bulletin board includes a thread for the API, and thus provides an ongoing transcript of questions and answers in which you might find the answer to your question. The Traders' Chat is an instant-message type of medium and doesn't retain any record of conversations.

- "To view or participate in the IB Bulletin Board, go to the **Education** menu and click *Bulletin Boards & Chats*. Click the **Bulletin Board** tab, then click the **Launch IB Discussion Forum button** to access all of our bulletin boards, including the TWS API bulletin board.
- To participate in the Traders' Chat, you need to click the **Chat** icon from the menu bar on TWS. Note that both of these customer forums are for IB customers only.

IB Customer Service

IB customers can also call or email customer service if you can't find the answer to your question. However, IB makes it clear that the APIs are designed for use by programmers and that their support in this area is limited. Still, the customer service crew is very knowledgeable and will do their best to help resolve your issue. Simply send an email to:

api@interactivebrokers.com

IB Features Poll

The IB Features Poll lets IB customers submit suggestions for future product features, and vote and comment on existing suggestions.

From the IB web site, click **About IB**, then select *New Features Poll*. Suggestions are listed by category; click a plus sign next to a category to view all feature suggestions for that category. To submit a suggestion, click the *Submit Suggestion* link.

Index