EspressChart®

ユーザガイド

Version 6.6

株式会社クライム

Quadbase Systems Inc.

Version 5.3

First Edition (January 2007)

Quadbase Systems Inc.
2855 Kifer Road, Suite 203
Santa Clara, CA 95051 USA
Tel: (408) 982-0835
Fax: (408) 982-0838
Email: support@quadbase.com
Web site: www.quadbase.com

©日本語版マニュアル 2010 年 1 月
株式会社クライム
〒103-0014 東京都中央区日本橋蛎殻町 1-25-4
日本橋栄ビル 4F
TEL:03-3660-9336
FAX: 03-3660-9337
Email: tech-support@climb.co.jp
Web: www.climb.co.jp

Contents

1 概要

EspressChart はダイナミック（動的）なチャートを作成したり、デプロイ（配置、展開）することができる非常に優れたツールです。EspressChart は洗練されたチャートの作成を容易にし、あらゆる環境下、どのプラットフォームでも、どのようなコンフィグレーションでもそれを表現できます。データ接続ツールを持つ EspressChart では、ユーザーがデータソースをクエリー（照会）することもでき、ポイントとクリックによる簡単な操作でチャートを描くことができます。また、EspressChart はオブジェクト指向の API（アプリケーション プログラミング インタフェース）を備えているため、ユーザがダイナミック（動的）なチャートやチャートデザイナーをアプリケーションや JSP(JavaServer Pages),サーブレットに簡単に組み込むことができます。EspressChart は 100％Pure Java で 書かれている公認されたソフトであり、ネイティブライブラリを一切使っていません。そのためどのプラットフォームでもチャートを描写することができます。また、JDBC/ODBC データソース（Excel スプレッドシートを含む）に直接接続することができ、テキストファイルや XML ファイル、EJB からもデータを取り込むこともできます。更に、柔軟性のある ExpressChart では、ユーザがアーギュメントや配列（アレイ）として直接チャートにデータセットを渡すことができます。ランタイム時にチャートは GIF, JPEG, PNG, SVG & SWF などのよく使われるイメージフォーマットにして描写されたり、常時インターラクティブにアプレット上にディスプレイされます。

1.1 このガイドの活用法

EspressChart ユーザーズガイドは大きく 3 つのセクションに分かれています。最初のセクション（１章と 2 章）はインストール、構造や設計、セットアップそして全ての EspressChart コンポーネントのコンフィグレーションを含むバックグラウンドの情報を提供しています。２番目のセクション（３章から９章）はチャートデザイナーを使ったチャート作成の基礎と EspressChart のさまざまな機能を紹介しています。チャートの開発を全てプログラミングで行おうと思っている方も、このセクションを一読して、EspressChart の基本的な機能と用語に慣れておいてください。３番目のセクション（１０章から１３章）はチャートのプログラミングとデプロイ（配置、展開）について書かれています。ここではチャートビューワアプレット、チャート API、サーブレット/JSP(JavaServer Pages)への組込み、オプション/コンフィグレーションのデプロイメントについて説明しています。このガイドの API のセクションは概要、モードの使用法、API での EspressChart 機能の使い方が説明されています。また、インストレーションに含まれる以下の２つの追加リソースがあります。

1. **API How To:** これは API チュートリアルを提供します。特に、サンプルコードを用いることによって、API の基本的なチャート作成や操作の機能をどのように行うか説明します。ガイドは EspressChart インストレーションの次のロケーションにあります。: *EspressChartInstallDir*/help/api_HowTo/index.html

2. **API JavaDoc:** API JavaDoc は EspressChart インストレーションの次のロケーションにあります。: *EspressChartInstallDir/* help/apidocs/index.html

サンプルチャートと API コード（サーブレットと JSP のサンプルを含む）は次のディレクトリーにあります。 *EspressChartInstallDir*/help/examples directory.

1.2 EspressChart コンポーネント

EspressChart は次の４つの主なコンポーネントから成っています。

チャートデザイナー: チャートデザイナーはビジュアルにポイント、クリック環境でチャートを作成、編集できるインタラクティブなアプリケーションです。アプリケーションとしてローカルで実行することもできるし、Web ブラウザでアプレットとしてロードすることもできます。ほとんどのチャートプロパティはコード書かなくてもデザイナーで簡単に設定、編集することができます。完成したチャートは表示や印刷のためにスタティック（静的）なフォーマットに直接エクスポートすることができます。チャートはテンプレートフィイル（.tpl）として保存することもできます。テンプレートはチャートの属性（色、サイズなど）を保存していますが、チャートデータは持っていません。テンプレートがロードされる度にソースから最新データが読み込まれます。チャートはプログラミングすることなく継続的にアップデートされるということです。テンプレートは別のチャートやチャートオブジェクト（API）に適用することができ、適用することによって、あるチャートのフォーマット情報を別のチャートに取り入れることができます。

チャートビューワ: チャートビューワはチャートをリモートでビューすることができるアプレットです。チャートを回転させたり、リサイズ（サイズの変更）したり、ズームやパンなどができます。ビューワでは個々のデータをクリックすることができ、クリックすると、そのデータポイントに関連のあるデータを復元したり、別のチャートへリンクしたりすることができます。

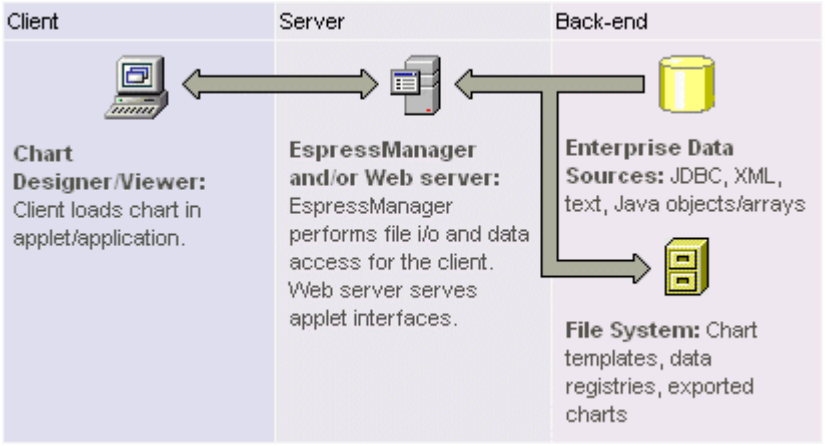
チャート API: チャート API は既存のアプリケーションやアプレットの中でチャートを作成し、カスタマイズし、デプロイ（配置、展開）することができる使いやすいアプリケーションプログラムインタフェースで、サーバサイドまたはクライアントサイドに置きます。Java パッケージによって高度な２D 及び３D のチャートライブラリ機能を使用することができます。たった１行のコードでチャートの作図が可能になります。

EspressManager: EspressManager は EspressChart の“バックエンド”です。要求に応じてデータベースにアクセスしたりクエリー（照会）を行う機能です。チャートデザイナーを使用している場合 EspressManager はクエリーユティリティとデータベースの間は シームレスコネクションになります。

また、EspressManager はデータバッファリング、ファイル I/O　及びユーザ認証も行います。チャートデザイナーの実行中に EspressManager が要求されても、チャートは API かチャートビューワを使ってデプロイするので EspressManager は使用されません。

1.3 EspressChart の構造

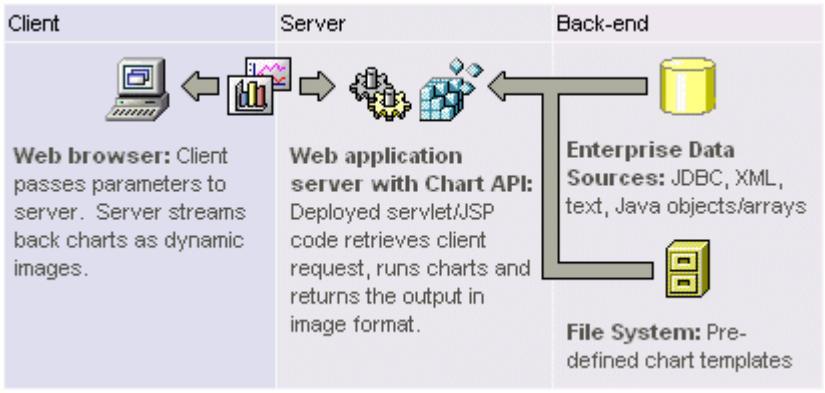
EspressChart は、設計時および実行時の両方で実行できる多種多様なコンフィグレーションが可能です。設計時、データアクセスツールで構成されるチャートデザイナーGUI インタフェースをアプリケーションとしてクライアントマシン上にロードするか、またはアプレットとしてクライアントサーバアーキテクチャ内にロードすることができます。次の図は、設計時にチャートデザイナーを使用して実行される EspressChart を示しています。



EspressChart Designer の構造

チャートデザイナーの実行中、サーバ側では EspressManager コンポーネントが実行されます。EspressManager は、セキュリティ制限のためにクライアントアプレットによって防止されているデータアクセスとファイルの I/O を実行します。EspressManager はさらに、データベース接続のための接続およびデータバッファリングと、マルチユーザ開発環境のための並行処理制御も行います。EspressManager は、チャートデザイナーと合わせて実行しなければなりません。

実行時、EspressManager を実行する必要はありません。EspressChart は、他のアプリケーション環境に組み込んで実行するように設計されているため、API クラスおよびチャートテンプレートファイルのデプロイメントは最小限で済みます。



サーブレット環境内の EspressChart

アプリケーションサーバ内で実行する場合、チャート API を使用して、サーブレットおよび JSP 内でチャートを作成し、作成したチャートをクライアント側のブラウザにイメージとして送ることができます。また、Web ページ内にチャートビューワをロードすることによって、クライアントがチャートを表示することもできます。

EspressChart は、クライアントサーバ環境内で実行できるだけでなく、シッククライアントアプリケーションでも実行することができます。チャート API をチャートビューワと組み合わせて使用すると、アプリケーションインタフェース内でチャートを表示/作成することができます。このコンフィグレーションでは、プロセス全体をクライアント上に入れることができるため、EspressManager を実行する必要はありません。上図は、考えられるすべてのデプロイメントコンフィグレーションを表していないことに注意してください。EspressChart のデプロイメントの詳細については、第 12 章を参照してください。

2 インストールIコンフィギュレーション

EspressChart のインストーラーには Windows、Unix、Mac OS X、Pure Java に対応する 4 つのバージョンがあります。EspressChart 自体は Pure Java ですが、それぞれのプラットフォームで EspressChart のインストールを簡単にするために、これらの異なるバージョンを用意しています。

2.1 インストーラーの実行

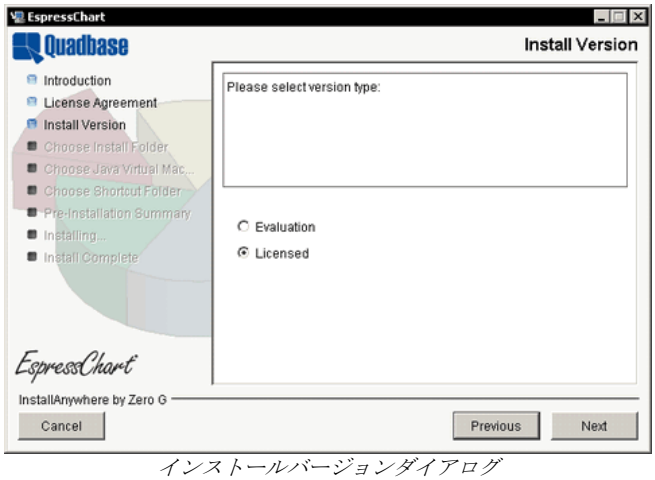
Windows バージョン: Windows のインストーラーヲ始動するには installEC.exe ファイルを実行し、インストーラーを開始させます。and the installer will launch.

Unix バージョン: Unix のインストーラーを始動するには installEC.bin ファイルを実行し、インストーラーを開始させます。.

Mac OS X バージョン: Mac インストーラーを始動するには InstallEC.zip ファイルをダブルクリックし、InstallEC.app ファイルを取り出します。InstallEC.app ファイルをダブルクリックし、インストーラーを開始させます。

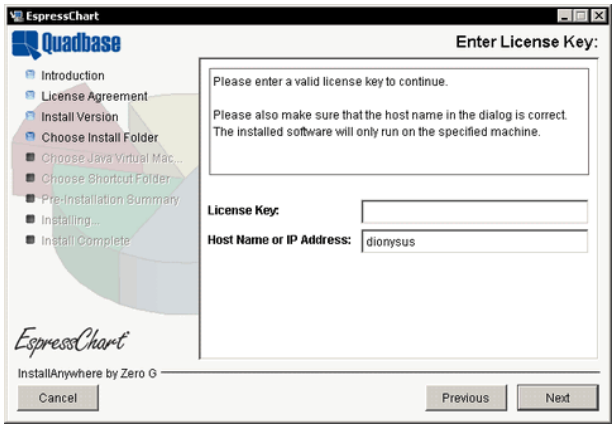
Pure Java バージョン: pure Java インストーラーを始動するには EspressChart がインストールされるマシーンに Java 1.2 もしくはそれ以上の Java Virtual Machine が既にインストートルされている必要があります。JMV が指定するパスにあることを確認してください。（あるいは、installEC.jar ファイルを JVM と同じディレクトリに移動します。）コマンドプロンプトから JVM ファイルを置いたディレクトリへナビゲートし、次のコマンドをタイプします。：java -jar installEC.jar　これによりインストーラーが開始します。

インストレーションプログラムが開始してライセンス契約に同意するとまず、評価版またはリリース版のどちらをインストールしたいかを質問するオプションが表示されます。



インストールバージョンダイアログ

リリース版のインストールを選択した場合、次のダイアログでライセンスキーの入力が要求され、EspressChart をインストールするマシーンのホスト名を検証します。

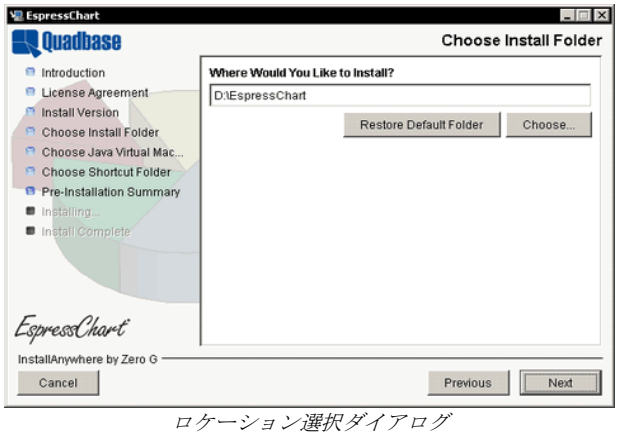


ライセンスキー入力ダイアログ

それらの情報を入力すると、インストーラーは Quadbase （クオードベース）にライセンスキーを登録します。登録に失敗すると EspressChart のリリース版のインストールを続行できません。ただし、評価版のインストールを続行するオプションが用意されているため、インストールが完了したら、http://www.quadbase.com/register.jsp にアクセスしてオンラインでキーを登録するか、または Quadbase 販売元に連絡してください。

*注意 – リリース版はインストール完了後、このダイアログで指定したホスト名でしか起動できないため、ダブルチェックしてホスト名が正しいかどうかを確認をしください。また、好みに応じてマシーンの IP アドレスを使用することもできます。

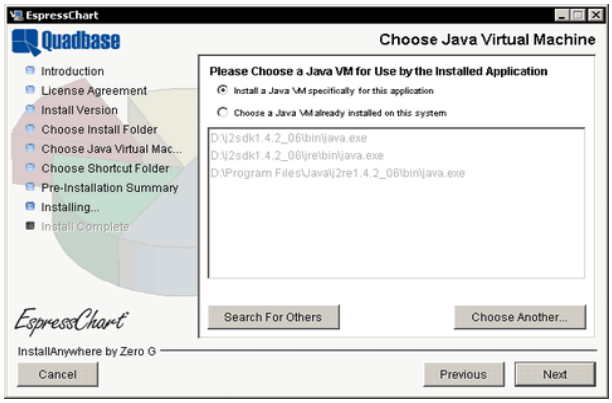
ライセンスキーが正しく登録されるか、または評価版のインストールを選択した場合、次のダイアログが表示され、インストール先を指定するように要求してきます。



ロケーション選択ダイアログ

デフォルトロケーションは \EspressChart\。新規ディレクトリを指定することもできます。 また、'Choose…' ボタンをクリックすることによってブラウズして選択指定することもできます。

次のオプションで EspressChart が実行するときに使用する Java バーチャルマシーン を指定します。（EspressChart は Java プログラムなので JVM（Java バーチャルマシーン） がその実行に必要となります。）

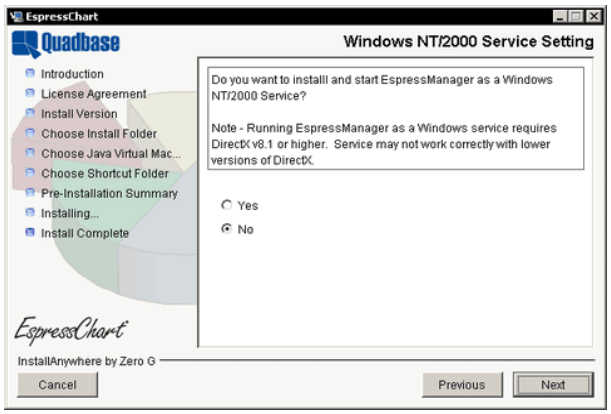


JVM 選択ダイアログ

EspressChart に搭載している JVM をインストールすることを選択できます。またシステムに既にインストールされている JVM を選択することもできます。インストーラーは JVM を検索し、全ての JVM をリストアップするのでそこから選択することができます。使用したい JVM がリストアップされていない場合は'Choose Another…' ボタンをクリックしてそれをブラウズさせます。

*注意 – インストーラーの pure Java バージョンを使っている場合はこのオプションは出てきません。JVM が必要な場合、プログラムを実行する JVM が備わっているインストーラーか確認してください。

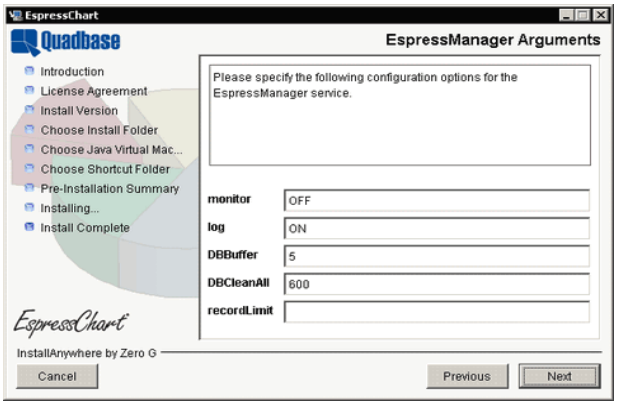
次のオプションは Windows リリースのためのものです。Unix バージョン、pure Java バージョンには利用不可であり、EspressChart の評価版でも利用不可です。このオプションでは、EspressManager を Windows NT/2000 の サービスとして実行させることができます。



NT/2000 サービスダイアログ

*注意 - Windows サービスとして EspressManager を正しく実行させるためには DirectX v8.1 もしくはそれ以上が Windows のシステムにインストールされている必要があります。

EspressManager をサービスとして開始するように指定すると、2 つの新たなオプションが続きます。まず先に、EspressManager のコンフィギュレーション情報を指定するプロンプトが表示されます。



EspressManage コンフィギュレーションダイアログ

これらのオプションは.bat ファイルまたは.sh ファイルを使って EspressManager を実行するときに利用するコマンドラインのアーギュメントと同じです。EspressManager のコンフィギュレーションに関する詳細は 2 章のセクション 2.3 を参照してください。

EspressManager を Windows サービスとして実行しようとする場合、インストーラーの次のオプションは、JDBC ドライバーのクラスを EspressManager Classpath に追加するかどうかです。これはデータベースに接続してチャートデータをリトリーブすることを可能にします。（.bat ファイルまたは.sh ファイルをクラスに追加するようにマニュアルで編集することができます。）ドライバーの追加を選択すると、次のオプションは追加するファイルの指定です。

インストーラーの最後のオプションはインストーラーの Windows または Mac OS X バージョン用です。プログラムのショートカットを作成して、スタートメニューに置くかディスクトップに置くか、または両方に置くかを指定するものです。デフォルトではショートカットが、スタートメニューに EspressChart という名前でプログラムグループに追加されます。

Mac OS X ではディスクトップ、ドックまたはフォルダーにエイリアスを作成することができます。

最後のオプションを完了すると選択したオプション全てのサマリーが表示されます。それからプログラムがインストールされます。

2.2 コンフィギュレーション

EspressChart のコンフィギュレーションは既に設定されています。そのコンフィギュレーションを変更しなくても EspressChar は実行できます。しかし次のような場合は、コンフィギュレーションの設定を変更する必要があります。

- EspressManager が使用するポート番号を変更したい。
- CTRL-J または CTRL-P を使っているアプレットからチャートを印刷したい。
- チャートデザイナーユーザを追加/削除/編集したい。

これらの設定は config.txt というファイルに保存されています。格納場所は次の通りです。: *EspressChartInstallDir*/userdb/ config.txt. 以下にサンプルコンフィギュレーションファイルを示します。

```
[port]
22071

[webroot]
C:\Inetpub\wwwroot

[users]
guest
Name1 Password1
Name2 Password2
```

[port] セクションの下に EspressManager に使いたいポート番号を設定します。デフォルトでは、この番号は 22071 になっています。また、ここに IP アドレスを設定して、マシーンに IP アドレスを問い合わさせるのではなく、EspressManager が使用

する IP アドレスを与えることもできます。IP アドレスを設定する場合はポート番号の後に IP アドレスを追加します。例えば次のようになります。

```
[port]
22071
```

to

```
[port]
22071, 204.147.182.31
```

このように変更することによって、EspressManager は起動時に常にこの IP アドレスを使用します。config.txt ファイルの [port]セクションでは、EspressManager が接続を確立するのに使用するドメインを複数指定できます。ドメインは[port] セクションの IP アドレスの後に追加します。

```
[port]
port number, ip address, domain1, domain2, ..., domain_n
```

サーチシーケンス（検索順）は IP アドレス、それからドメインの `domain1, domain2,. . .` となります。

[webroot]セクションは Web サーバの webroot（ウェブルート）へのパスが入っています。（CTRL-J または CTRL-P を使って印刷する場合）、チャートがチャートビューワからエクスポートされるとき、EspressManager ユーザはこれを使います。このオプションの詳細は 10 章に記述があります。

[user]セクションは各行がユーザ名とパスワードから成っています。ユーザ名”guest”でパスワード無しがデフォルトで設定されています。この行を削除して必要なだけユーザを追加することができます。各ユーザ名とパスワードは 1 つまたはそれ以上のスペースで区切られます。（つまり、ユーザ名パスワードにスペースを含めることができません。）また、ユーザー名もパスワードも大文字と小文字は区別します。

2.3 EspressManager のスタート

デザイナーをスタートさせる前に EspressManager が実行されていなければなりません。ほとんどの場合、インストールするときは EspressManager が Web サーバマシーンで始動されている必要があります。EspressManager とチャートデザイナの両方がローカルで実行されながら、EspressChart をスタンドアローンマシーンで使用することは可能です。この場合、割り当てられる IP 番号は 127.0.0.1 になります。これはログファイル *EspressChartInstallDir*/EspressManager.log を作成し、このログファイルはモニタリングとして、問題が起きたときの診断などに役立ちます。各チャートデザイナーユーザのマシーンに EspressManager がインストールされている必要も、あるいはスターとされている必要もありません。

EspressManager をスタートするには EspressChart のルートディレクトリにある EspressManager の.bat または.sh ファイルを実行します。（このファイルはインストール時に自動的に作成されます。）Windows インストレーションではインストー

ル時に指定したオプションによりますが、スタートメニューやディスクトップに追加されたプログラムのショートカットを使用することができます。そうするとEspressManager が自動的にデフォルトプロパティでスタートします。

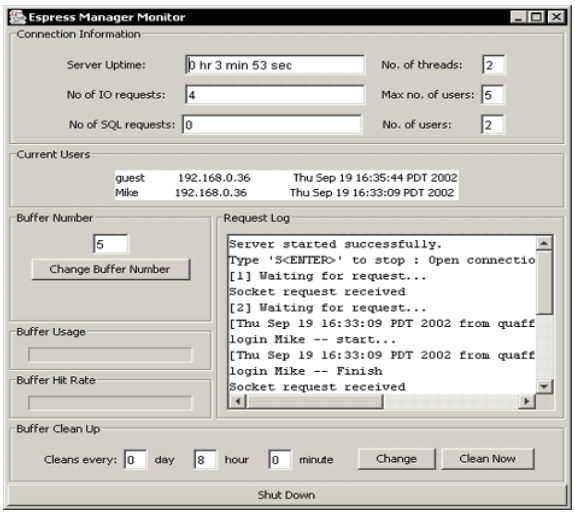
また、アーギュメントを使って EspressManager のコンフィギュレーションを設定することもできます。アーギュメントはダッシュ“ー”の後にコマンドセットが続く形式をとっています。各アーギュメントにスペースは不要ですが、異なるアーギュメント間を区切るために、アーギュメントとアーギュメントの間に最低ひとつのスペースが必要です。

EspressManager を起動する際にコマンドラインにこれらのアーギュメントを入力することができます。また、espressmanager.bat/.sh ファイルを修正してアーギュメントを追加することもできます。あるいは、EspressChart をインストールする/userdb/ディレクトリ内にある ServerCommands.txt ファイルにアーギュメントを入れることもできます。

-help: "-help"とタイプすると EspressManager は利用可能なアーギュメントと各アーギュメントの意味に関する情報を提供します。"-h" または "-?"でも同じオンラインヘルプの情報を得ることができます。

-log: "-log" アーギュメントをタイプすると、ログファイルが作成され、全てのクライアント/サーバネットワーク情報が記録されます。アドミニストレータ（管理者）が各ユーザの要求を評価するのにログファイルをオープンするかもしれません。ログは EspressManager.log というファイル名で保存されます。

-monitor:ON/OFF: "-monitor"アーギュメントを指定すると、ステータスを表示するグラフィックユーザーインターフェースの EspressManager モニターが表示されます。また、このモニターで EspressManager の設定を変更するもできます。



EspressManager モニター

-runInBackground:ON/OFF: このアーギュメントでは、EspressManager をバックグラウンドプロセスとして実行するかどうかを指定することができます。バックグラウンドプロセスとして実行すると、ユーザはシャットダウンを入力できません。EspressManager はスクリプトから実行されます。このオプションを正しく動作させるためには、"-monitor:OFF" アーギュメントと組み合わせて使用しなければなりません。この方法で EspressManager を実行した場合、シャットダウンするにはプロセスを終了させるしかありません。

-recordLimit:nn: クエリー実行時に EspressManager がデータベースから読み出すレコード数の最大値を設定します。最大値に 達すると、EspressManager はクエリーの実行を停止します。この機能は、ユーザがメモリにストアできる以上のレコードを読み出すことを防止し、メモリ不足エラーのために発生するクラッシュを防止することができます。

-queryTimeout:sss: チャートクエリーのタイムアウト（中断）インターバル（間隔）を秒単位で設定します。クエリー実行時間がタイムウトアーギュメントを経過すると、EspressChart はクエリーをアボート（中止）します。この機能はユーザが偶発的にランナウェイクエリーを作成するのを防止します。

-DBBuffer:nnn: チャートのデータソースとしてデータベースが使われている場合、このアーギュメントを使って、データベースコネクションとチャートに使われるデータソースの両方をバッファリングすることができます。これはつまり、DBBuffer が 0 に設定されていれば、毎回チャートが作成され、実行され、それはデータベースへ接続を再確立し、クエリーの再実行を行います。しかし、DBBuffer をオンにすれば、コネクションとデータはキャッシュにストアされます。ストアされるコネクションとクエリーの数が DBBuffer アーギュメントで指定する 0 から 999 の数値によって設定されます。

-DBCleanAll:ddhhmm: データソースのデータは定期的に更新されるかもしれません。そこで、データバッファーオプションが使用されると（つまり DBBuffer の値が 0 でない）、最新のデータを得るためにバッファーをリフレッシュしたくなるかもしれません。"-DBCleanAll" アーギュメントはバッファーをクリアし、データソースからデータを取ってくる時間の間隔を示すのに使われます。ddhhmm の値は"dd" が days（何日）、"hh" が hours（何時間）、"mm" が minutes（何分）です。また、EspressManager は省略フォーマットをサポートしています。つまり、数字を省略すると、高い方の数字が省かれます。
例:

DBCleanAll: 10 日間と 10 時間 10 分ごとにバッファーをクリア
DBCleanAll: 10 時間 10 分ごとにバッファーをクリア
DBCleanAll: 10 分ごとにバッファーをクリア

このアーギュメントは次のような場合無効です。

"-DBBuffer" アーギュメントが 0 に設定されている場合（バッファーなし）
"-DBCleanAll" アーギュメントが 0 に設定されてる場合（常にメモリがクリア）

EspressManager のスタート時にリフレッシュするインターバル（間隔）が出力表示されます。

-RequireLogin:API を利用してチャートデザイナーを起動する際にセキュリティを適用するには、このアーギュメントを使用します。通常、API を経由してチャートデザイナーを呼び出す場合、ユーザ認証は行われません。このアーギュメントを使用することにより、ユーザは自分自身の認証をプログラムに適用できますが、無認可のユーザもサーバにアクセスできます。これを防止するために、ユーザはこのアーギュメントをオンに設定して（値は true/false）、API からチャートデザイナーを呼び出す際にチャートデザイナーに対する認証を強制的に実行させることができます。このアーギュメントをオンにした場合、Qb チャートデザイナー オブジェクトが表示されるたびに、API メソッドを呼び出して（config.txt ファイルに定義されている）正しいユーザ名とパスワードも指定しなければなりません。API からチャートデザイナーを呼び出すための詳細については、セクション 11.5.6 を参照してください。

-QbDesignerPassword: このアーギュメントでは、-RequireLogin アーギュメントをオンにしたときに使用するパスワードを設定できます。config.txt ファイルからのログインを使用する代わりに、Qb チャートデザイナー オブジェクトが表示されたときにメソッドを呼び出してサーバーに提示しなければならない特定のパスワードを設定することができます。API からチャートデザイナーを呼び出すための詳細についてはセクション 11.5.6 を参照してください。

-MaxRecordInMemory:nnn: メモリーに読み込み込むことのできる最大レコード数を設定します。クエリーを停止させるためのレコード制限とは異なり、この機能は、しきい値に達すると、システム上のテンポラリファイルに対するデータのページングを開始します。チャートの表示やエクスポートは続行されますが、データはページファイル から読み込まれます。この機能は、チャート実行中にシステムがメモリ不足になるのを防ぎます。

-MaxCharForRecordFile:nnn: レコードファイルの保存を呼び出したときに生成されるページングファイルの 1 フィールド当たりの最大文字数を設定します。データ内のレコードの文字数がこのアーギュメントより長い場合、完成されるチャートでは文字が切り捨てられます。

-FileRecordBufferSize:nnn: レコードの保存を呼び出したときに一度にページングされる最大レコード数を設定します。バッファのサイズはパフォーマンスに影響を与え、バッファのサイズ大きいほど、チャートの作成速度が早くなります。

-singleTableForDistinctParamValue "-singleTableForDistinct ParamValue" アーギュメントを使用する場合、パラメータ化されたチャートのパラメータダイアログは異なって描画されます。パラーメータをデータベースのカラムにマッピングすると、 個別のリストが描画され、選択された個別リストはマッピングされたフィールドに対してのみ実行されます。デフォルトの動作では （このアーギュメントを使用しない場合）、オリジナルのクエリー内の結合および条件を使用して個別パラメータリストを制約します。パラメータ化されたクエリーの詳細については、セクション 5.2.2.2 を参照してください。

Mac OS X 上で実行し、インストール時にエイリアスを作成するように選択した場合、espressmanager.app パッケージを修正して EspressManager の設定を変更する必要があります。このためには、espressmanager.app 上で右クリック（CTRL+Click ）し、ポップアップメニューから 'Show Package Contents'を選択します。次に、Contents フォルダに進みます。このフォルダに'info.plist'という名前のファイルがあります。テキストエディター内でこのファイルを開いて、Java プロパティ "lax.command.line.args" にこのアーギュメントを追加します。

2.3.1 サーブレットとしての EspressManager のスタート

EspressManager はアプリケーション プロセスとして実行されるだけでなく、アプリケーションサーバ/サーブレットランナー内のサーブレットとしても実行することができます。この場合、EspressManager はソケットの代わりに http を使用してクライアントと通信します。このコンフィギュレーションの利点は、EspressManager がアプリケーションサーバと同一のポートを共用できるため、ファイアウォールの背後からの配備が容易になるという点です。さらに、次のように EspressManager を実行すれば、ユーザはリモートアドミニストレーション（遠隔地管理）を実行できます。

EspressManager をサーブレットとして配備するには、以下の手順を実行します。

1. EspressManager.jsp、MenuError.jsp およびアプリケーションサーバルート（もしくは仮想ディレクトリ）など http 経由でアクセス可能なディレクトリ下にある/WebComponent/ディレクトリをコピーします。

2. EspressChart/classes ディレクトリの内容をコピーし、サーブレットコンテキストにより利用可能なクラスを作成します。

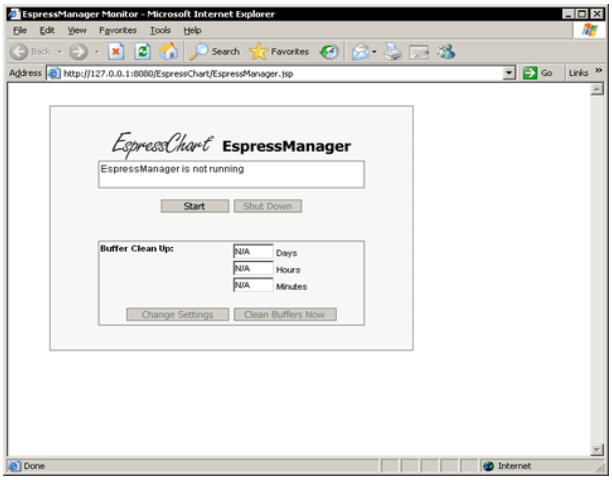
3. QuadbaseDirectory.cfg ファイルを EspressChart インストレーションディレクトリからアプリケーションサーバのワーキングディレクトリにコピーします。ワーキングディレクトリを検出するには、EspressManager サーブレットクラスをコピーしたサーブレットコンテキストから whatIsMyWorkingDirectory を実行します。サーブレットを呼び出すと、サーブレットはワーキングディレクトリのパスをブラウザに出力表示します。

4. EspressManager.jar を EspressChart インストレーションの/lib/ ディレクトリからアプリケーションサーバの CLASSPATH に追加します。実行中のアプ

リケーションサーバに応じて、/lib/ ディレクトリから parser.jar と jaxp.jar を追加 しなければならない場合があります (たとえば、トムキャットにはすでに XML パーサーが装備されています)。また、EspressManager からデータベースに接続するために JDBC ドライバのクラスを追加しなければならない場合もあります。

5. アプリケーションサーバをリスタートします。

これら全てのステップを完了したら、ステップ１で説明したとおり、Web ブラウザで、アプリケーションサーバ内で利用可能な EspressManager.jsp ページを指示して EspressManager ページをロードします。



EspressManager JSP モニター

ロードされたこのページで'Start'ボタンをクリックし、EspressManager を起動します。EspressManager が起動されたら、バッファの設定項目の設定または変更を行ったあと、このウィンドウからシャットダウンすることができます。EspressManager を実行したままこのウィンドウを閉じることもできます。この場合、EspressManager は、JSP を再度呼び出して EspressManager をシャットダウンするまで実行を継続します。

EspressManager をサーブレットとして実行する場合、EspressManager に接続しているすべてのコンポーネントが修正を必要とするため、それを考慮しておいてください。このようなコンポーネントとしては、チャートデザイナー、スケジューラー、チャートビューワ、ページビューワおよびチャート API を使用するすべてのアプレリケーション/サーブレット/JSP があります。EspressChart の配備の詳細については、１３章を参照してください。

2.3.1.1 トムキャット4.x 内へのサーブレットとしての EspressManage の配備

以下では、トムキャット 4.x 内で EspressManager をサーブレットとしてセットアップして実行する方法について説明します。手順は、前のセクションで説明した手順と同じです。この例では、ポート 8080 上でローカルで（IP 127.0.0.1）トムキャットを実行しているものとします。ご使用のコンフィギュレーションが異なる場合は、以下の手順の IPO およびポートをご使用の IP およびポートに置き換えてください。

1. トムキャットのルート(<トムキャットのインストレーションディレクトリ>/webapps/ROOT)の下に"EspressChart"という名前のディレクトリを作成します。 EspressManager.jsp 、 MenuError.jsp ファイルおよび /Web_Component/ ディレクトリを EspressChart インストレーション(<EspressChart インストレーションディレクトリ>)からトムキャットのルート下に新しく作成した/EspressChart/ ディレクトリにコピーします。

2. <EspressChart インストレーショディレクトリ>/classes ディレクトリの内容を <トムキャットインストレーションディレクトリ>/webapps/ROOT/WEB-INF/classes ディレクトリ にコピーします。ここで、呼び出し元が<Tomcat Installation Directory>/conf/web.xml file でコメントになっていないことを確認する必要があります（トムキャットではデフォルトにより、呼び出し元がコメントになっているため、"/servlet/"コンテキストを利用できません）。

3. QuadbaseDirectory.cfg ファイルを EspressChart ディレクトリから<トムキャットインストレーションディレクトリ>/bin ディレクトリにコピーします（通常、これはトムキャットのワーキングディレクトリです。ただし、トムキャットを別の場所から起動した場合、ワーキングディレクトリは異なる可能性があります。ワーキングディレクトリを確認するには、Web ブラウザを開いて、トムキャットを実行させた状態で”http://127.0.0.1:8080/servlet/whatIsMyWorkingDirectory"と入力します。このサーブレットはワーキングディレクトリのパスを表示します）。ワーキングディレクトリが /bin/ ディレクトリ以外のディレクトリであれば、QuadbaseDirectory.cfg ファイルをそのワーキングディレクトリに格納します。

4. (<EspressChart Installation Directory>/lib の下にある)EspressManager.jar ファイルをトムキャットの CLASSPATH に追加します。通常、こうするためには、<トムキャットインストールディレクトリ>/bin ディレクトリ内にある setclasspath.bat または .sh ファイルを編集します。

5. トムキャットサーバをリスタートします。

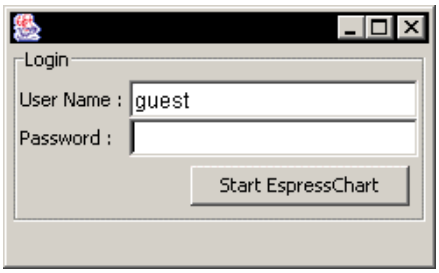
以上で EspressManager は正しく配備されます。EspressManager を起動するには、Web ブラウザを開いてアドレス

http://127.0.0.1:8080/EspressChart/EspressManager.jsp。に進みます。モニタがロードされます。'Start'ボタンをクリックすると、EspressManager が起動されます。

2.4 チャートデザイナーのスタート

チャートデザイナを実行するには、スタンドアローンのアプリケーションとして実行する場合と Web ブラウザを通してロードするアプレットとして実行する場合と 2 つの方法があります。どちらの場合もチャートデザイナーをスタートさせるために EspressManager が実行されていなければなりません。前章に EspressManager のスタートに関する説明がありますので参考にしてください。

アプリケーションとして実行: チャートデザイナをスタートさせるには EspressChart インストレーションのルートディレクトリに置かれている Designer の.bat もしくは sh ファイルを実行します。Windows インストレーションでは、インストレーション時に指定したオプションによりますが、スタートメニューまたはディスクトップに追加されたショートカットを使用することができます。ユーザ名とパスワードの入力を要求するダイアログが出力されます。config.txt ファイルにユーザの設定をしてある場合はそのユーザ名とパスワードを入力します。ユーザを設定していない場合はユーザ名"guest"と入力してパスワードは何も入れないでおきます。"Start EspressChart"をクリックするとアプリケーションがスタートします。



チャートデザイナのログインウィンドウ

ブラウザからの実行: チャートデザイナをリモート（遠隔）で実行している場合 EspressManager がそのリモートマシーンで実行されているか確認し、それからブラウザで EspressChart の URL をタイプします。例 http://machinename/espresschart/ index.html. 最初のページには EspressManager へログオンするためのダイアログプロンプトが含まれています。config.txt ファイルを編集してユーザ設定を行っている場合はユーザ名とパスワードを入力します。config.txt ファイルの編集をしていない場合はユーザ名"guest" といれてパスワードは何もいれないでおきます。"Start EspressChart"をクリックするとチャートデザイナがロードします。

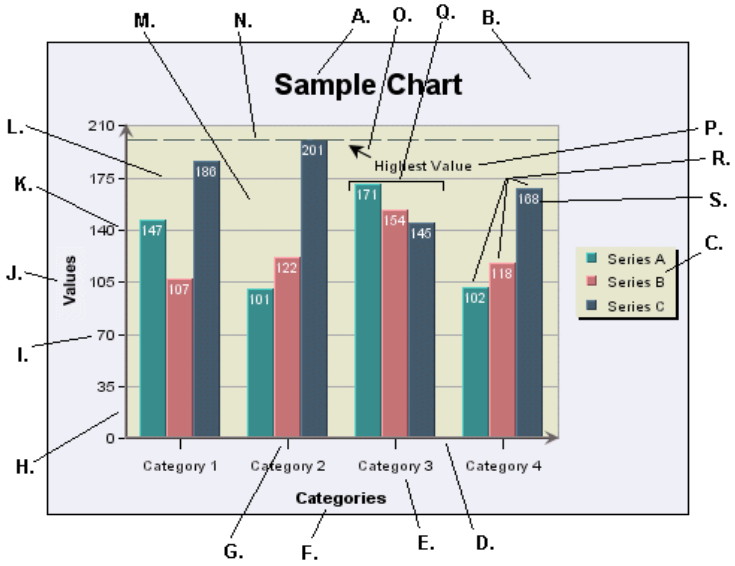
*注 意- EspressChart は JDK 1.2 またはそれ以上の JVM を必要とするめ、ブラウザを通してチャートデザイナを実行するには Java プラグインをダウンロードする必要があります。.

3 チャートの基礎

この章ではチャートのパーツや基本的なデータマッピングから保存/エクスポートのオプションまで EspressChart の基礎となることを説明します。この章で説明することは全てあとの章でより詳しい説明がされています。

3.1 チャートの要素

次の図はチャートを形作るさまざまなコンポーネントを表しています。また、この図で示されている表現が、このガイドでは使われています。



A. **Main Title(主題/タイトル):** チャートの主題/タイトルです。

B. **Chart Canvas（チャートキャンバス）：** チャットの背景です。キャンバスはチャートのサイズや境界線を決定します。通常はエクスポートすると同じサイズになります。キャンバスの色を変更したり、背景のイメージファイルを設定したり、追加したりすることができます。

C. **Legend（凡例）：** チャートの凡例です。凡例は色分けされたカテゴリーやシリーズ名を表示します。追加されるトレンド/コントロールラインやコントロールエリアなどの第二の値も凡例に表示されます。

D. **Category (X) Axis（カテゴリー/X 軸）：** チャートの X 軸あるいはカテゴリー軸です。一般的にカテゴリー軸はデータセットからの個別エントリーに区分し、チャートにそれぞれの値を描くようにします。（値は通常 Y 軸で表しま

す。） バーチャートやガントチャートのようなタイプのチャートはＹ軸にカテゴリーで、X 軸に値を描くような逆の場合もあります。分散チャートやバブルプロットチャートのようなタイプのチャートは２D や３D で点を使って値を表すので各軸とも値を示すことになります。

E. X Axis Labels（X 軸ラベル）：X 軸のエレメントまたはカテゴリーのラベルです。

F. X Axis Title（X 軸タイトル）：X 軸のタイトルです。

G. X Ticker（X ティッカー）：X 軸の目盛りです。デフォルトではそのチャートの各データポイントに合わせられます。

H. Value (Y) Axis（値/Y 軸）：チャートの Y 軸または値軸です。一般的に Y 軸はカテゴリーの各値を示します。デフォルトでは Y 軸の目盛りはデータセットに最も合うように自動的に作られますが、マニアルで調整することもできます。コンビネーションチャートでは、プロットの右手側に第２値軸が引かれます。

I. Y Axis Labels（Y 軸ラベル）：Y 軸の値のラベルです。

J. Y Axis Title（Y 軸タイトル）：Y 軸のタイトルです。

K. Y Axis Ticker（Y 軸ティッカー）：Y 軸の目盛りです。

L. Y Grid（Y グリッド）：Y 軸の各目盛りごとに引かれたグリッド線です。グリッド線は X 軸にも（３D チャートなら Z 軸にも）引くことができます。

M. Plot Area（プロットエリア）：T 全てのデータポイントが表示されている軸によって境界線が引かれている領域です。この領域に色をつけたり、その他のオプションを使って領域の周りに境界線を引いたりできます。プロットエリアはチャートキャンバス上で移動させたり、サイズを変更したりすることができます。

N. Control Line（コントロールライン）:チャートに追加できる特殊な線です。この例では、この線はシリーズの中の最も高い値に引かれています。コントロールラインは平均値、複数の標準偏差の値に引くことができます。ユーザはいろいろなトレンドラインもチャートに追加することができます。

O. Floating Line（フローティング）：フローティングラインはチャートに追加する任意のラインです。ここでの例では矢印でポインターとして使われています。フローティングラインはチャートプロットに相対的に位置を移動します。中を塗りつぶした形のものを作成できます。

P. Annotation Text（注釈テキスト）：（ラベルやタイトルではなく）チャートに追加される任意の注釈です。EspressChart はチャートキャンバスのどこにでもテキストを配置することができます。フローティングラインのように、

注釈テキストはチャートのプロットに相対的な位置に置かれるので、チャートプロットに対応して移動します。

Q. Category Elements（カテゴリエレメント）: カテゴリエレメントを示すプロット（線）です。このチャートにはデータシリーズがあり、各カテゴリは 3 つのポイントから成っていることを示しています。

R. Data Series Elements（データエレメント）:データカテゴリはそれぞれの（値の）ポイントから成っています。各チャートではシリーズのデータグループを線で示す（プロットする）ことができます。カテゴリーとシリーズに関する詳細は 3 章のセクション 3.2 を参照してください。

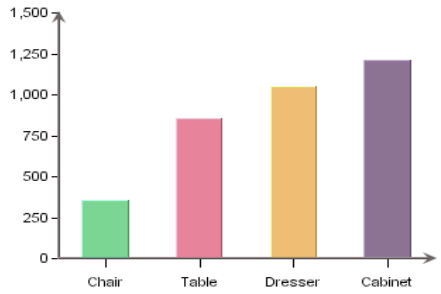
S. Data Top Labels(データトップラベル):それぞれのデータには各値を表示するラベルがあり、それは各データがポイントするところに示されます。

3.2 データマッピングの基礎

データマッピングとは未加工の素のデータがチャートに描写されるときの手法です。データは多数のソースから取り込むことができますが、EspressChart はテーブル形式のデータ構造を扱います。ですから、アーギュメントで渡されたデータや XML ファイルからのデータはマッピングする前にテーブル構造に変換されていなければならないのです。（5 章でさまざまなソースからどうやってデータを取り出すかを説明しています。）データの基本セットは、例えば以下のようなものがあります。

Product（製品）	Sales（販売個数）
Chair（椅子）	362
Table（テーブル）	862
Dresser（鏡台）	1052
Cabinet（飾り棚）	1211

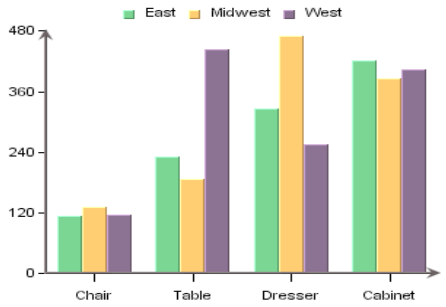
このデータをチャートに描くには、**Product** （製品）カラムの各エントリーごとに **Sales** （販売個数）の値を描きたいと考えます。プロダクト（製品）はカテゴリー、販売個数は値ということになります。チャートには **Product** カラムを X(カテゴリー)軸に、そして **Sales** カラムを Y（値）軸にマッピングします。この結果チャートは以下のようになります。



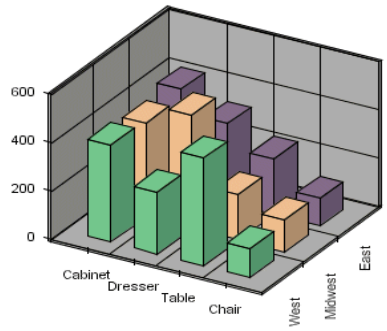
次にカテゴリーカラムの個々のエレメントに対して値を入れたカラムがあります。カテゴリー値に加えて、データシリーズの形式で表示される追加情報が入っています。例えば、プロダクト別の販売個数を表示するだけでなく、販売地域という追加エレメントをデータに入れます。この場合テーブルは次のようになります。

Product（製品）	Region（地域）	Sales（販売個数）
Chair（椅子）	East（東部）	114
Chair（椅子）	Midwest（中西部）	131
Chair（椅子）	West（西部）	117
Table（テーブル）	East(東部)	231
Table（テーブル）	Midwest（中西部）	187
Table（テーブル）	West（西部）	444
Dresser（鏡台）	East(東部)	327
Dresser（鏡台）	Midwest（中西部）	469
Dresser（鏡台）	West（西部）	256
Cabinet（飾り棚）	East(東部)	422
Cabinet（飾り棚）	Midwest（中西部）	386
Cabinet（飾り棚）	West（西部）	403

製品ごとに各地域の値を表示するために、**Region （地域）**カラム をデータシリーズとしてデータマッピングに追加しています。これをチャートにすると次のようになります。



各カテゴリーに 3 つのデータポイントがあり、3 つの異なる地域を表します。2 次元チャートなのでシリーズは必ず一列に横並びになります。3 次元チャートではシリーズを一列に並べて描くこともできますが、規定値では Z 軸に表されて横並びにはなりません。次の図は同じチャートを３D（3 次元）にしたものです。



このチャートではデータシリーズが Z 軸に描かれています。ここでは、データが見やすいように、カテゴリーの順序は変更されています。これはデータマッピングにおいて基本のコンセプトです。ほとんどのチャートタイプでこのマッピング、またはこれに類似するマッピングオプションを使います。

3.3 チャートの保存とエクスポート

チャート定義の保存とイメージファイルとしてのチャートのエクスポートにはいくつかのオプションがあります。チャートの保存とエクスポートの方法（チャートデザイナ及びチャート API での方法も含む）は 9 章及び 11 章に詳細が書かれています。

3.3.1 チャート定義の保存

EspressChart で作成されたチャート定義を保存するには、チャートとして保存するかテンプレートファイルとして保存するかの 2 つ方法があります。

チャートファイル: チャートファイルは *filename*.cht （ファイル名.cht）と呼ばれるバイナリファイルでチャートを保存します。チャートファイルはチャート定義（タイプ/型、次元など）とチャートの作成に使われたデータの両方を保存します。このためチャートファイルは携帯するのに便利です。いつでもチャートファイルを開くとチャート作成に使ったオリジナルのデータも一緒に開きます。チャートを開いた後で、ソースからのデータをリフレッシュできます。または、チャートのデータソースを全部変更することもできます。

テンプレートファイル: テンプレートファイルは *filename*.tpl（ファイル名. tpl）と呼ばれるバイナリファイルでチャートを保存します。テンプレートファイルはチャート定義を保存するだけでデータは保存されません。従ってテンプレートファイルは、開くたびにオリジナルデータソースへ接続してデータを取り込もうとします。このためテンプレートファイルはチャートファイルに比べると携帯するには不向きということになります。テンプレートファイルはチャートからチャートへ属性を引き渡すためにも使われます。これをするにはチャートにテンプレートを適用します。適用するとテンプレートのさまざまな属性をカレントチャートに引き継ぐことができます。テンプレートの適用に関する詳細は 9 章セクション 9.1.1 を参照してください。

更に、チャートファイルとテンプレートファイルはバイナリ形式でチャート定義を保存できるだけはなく、XML 形式でも保存することが可能です。XML チャート定義ファイルはチャートデザイナーやチャート API の外側で編集するこができます。XML チャート属性のリストは Appendix（付録）B を参照してください。

3.3.2 イメージファイルの生成

一般的にチャートが Web 経由でデプロイされるには、チャートがアプレットとしてデプロイされるか、イメージファイルを生成することによってかのどちらか一方の方法で行われます。アプレットはテンプレートファイルまたはチャートファイルのチャート定義を直接ロードします。アプレットの使用に関する詳細は 10 章チャートビュワーを参照してください。EspressChart は次の形式でチャートの画像ファイルを描写します。

GIF: EspressChart RLE かまたは LZW のどちらかの圧縮法を使って GIF 画像を生成します。LZW の方がより早く、より小さいファイルを生成しますが、特許によって保護されています。LZW 圧縮をアンロックするためには Unisys からライセンスを取得する必要があります。規定値では GIF フィルあるは RLE 圧縮を使って生成されます。

JPEG: JPEG はよく使われているもう 1 つの画像形式です。 GIF より高解像度の画像形式で、特許による保護もありません。JPEG ファイル生成時にはファイルの画質/圧縮を指定することができます。画質が高ければ高いほど、ファイルは大きくなります。

PNG: PNG はあまり使われない画像形式ですが、ほとんどのブラウザで表示が可能です。JPEG より小さいサイズのファイルで高画質の画像です。この形式には 3 つの圧縮オプションがあります。

SVG: SVG （Scalable Vector Graphics：スケーラブルベクトルグラフィックス）は XML ベースのテキストフォーマットのベクトルとしてイメージを保存する 比較的新しい画像フォーマットです。一般的にこれらの画像を表示するのにブラウザプラグインを必要とします。

SWF: SWF は Macromedia Flash Fiel（マクロメディア　フラッシュ ファイル）です。フラッシュ形式はベクトルがベースでエクスポートしたあとにチャートのサイズ変更（リサイズ）が可能です。フラッシュも高解像度の出力でサイズの小さいファイルを生成します。

BMP: これは Windows ビットマップ形式です。

WMF: WMF は Windows Meta File（メタファイル）形式です。 これは MS オフィースのドキュメントへのインポート/エクスポートに使われます。

EspressChart では静止画像に加え、チャートデータをテキストファイルや XML ファイルとしてエクスポートすることも可能です。.

4 Quick Start

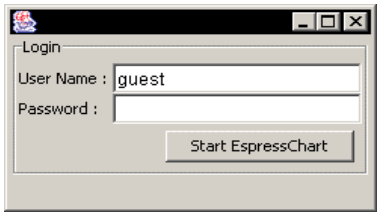
This guide will run you through the basics of setting up a data source and creating charts using Chart Designer. This guide assumes you are running locally on a Windows NT/2000/XP machine.

If you're not running on a Windows machine you can still use this guide, however, you won't be able to use the sample Access database. There is a text file that you can use instead to create the same chart.

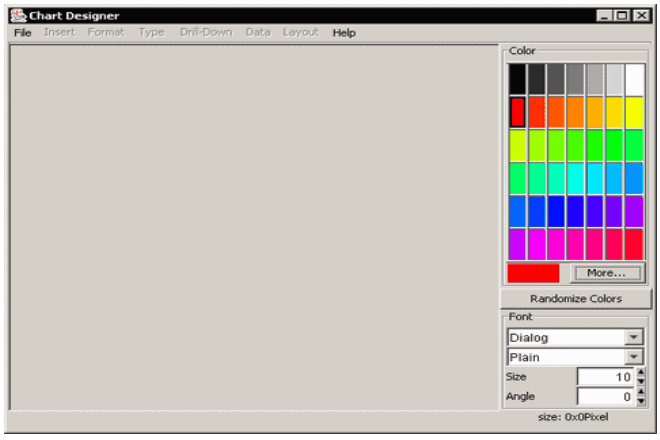
4.1 Step 1: Start Chart Designer

To start Chart Designer you will first need to start EspressManager. To start EspressManager, execute the EspressManager batch file that is in the root directory of your installation. For Windows installations, you can also start EspressManager from the Start menu if you selected to create shortcuts when installing the software.

Once EspressManager is running, you can launch Chart Designer by executing the Designer batch file in the root directory of your installation. For Windows installations, you can also start Chart Designer from the Start menu if you selected to create shortcuts when installing the software. When Chart Designer is started, a login window will appear prompting you for a user name and a password. Enter "guest" as the user name, and leave the password field blank.

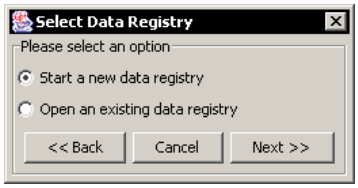


Click 'Start EspressChart' and Chart Designer will open as a new window.

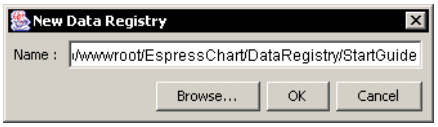


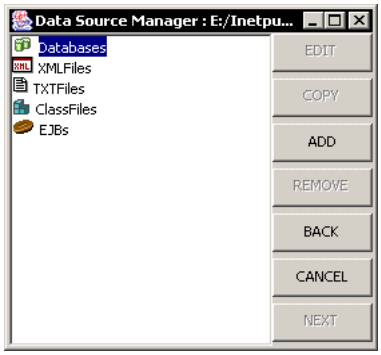
4.2 Step 2: Start new chart & data source registry setup

To start a new chart, select 'New' from the File menu. A dialog will appear asking if you would like to create a new data source registry, or use an existing one. A data registry is an XML file that stores database connections, queries, and file locations for text and XML sources. Select 'Start a new data registry' and click 'Next'.



You will then be prompted to enter a name for your data registry. You can name it anything you want, and an XML file of the same name will be created in the DataRegistry directory. After you have specified a name, the Data Source Manager will open. Since this is a new registry there are no nodes under the data source types in the window.





4.3 Step 3: Data sources setup

The first data source to set up is a database. EspressChart comes with two sample databases. One is HSQL which is a pure Java application database. The other is a MS Access Database. Both contain the same data.

To set up the HSQL data source, you will need to modify EspressManager so that it picks up the classes for the JDBC driver when it starts. (This procedure will be the same for any JDBC data source). To modify the EspressManager's classpath you will need to modify the EspressManager.bat or .sh file that was created by the installer.

Open the EspressManager .bat or .sh file in a text editor or at the command line, and modify the "-classpath" argument to point to the hsqldb.jar file which is under the help/examples/DataSources/database directory of your EspressChart installation. Be sure to put the path to the .jar file in the argument as well. In the .bat file separate files in the classpath with a semi-colon, and in the .sh file use a colon.

*Note - If you're running EspressManager and/or Chart Designer, you'll want to shut them down before modifying the .bat or .sh files, as you'll need to re-start them in order for the changes to take effect.

The modified EspressManager.bat should look like this:

```
set JAVA_EXECUTABLE=D:\j2sdk1.4.1_02\bin\java.exe
set PATH=%PATH%;.\lib\
"%JAVA_EXECUTABLE%" -Xmx32M -classpath ".\lib\EspressManager.jar;
.\lib\FlashExport.jar;.\lib\ExportLib.jar;.\lib\jaxp.jar;
.\lib\parser.jar;.\lib\SVGExport.jar;
.\help\examples\DataSources\database\hsqldb.jar;.;"
quadbase.common.server.Server -monitor:ON -log -DBBuffer:1 -
DBCleanAll:600
```

The modified EspressManager.sh should look like this:

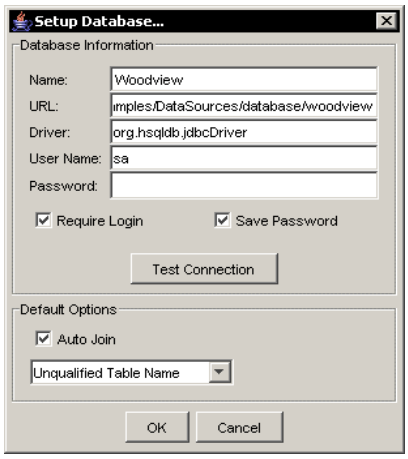
```
JAVA_EXECUTABLE=/System/Library/Frameworks/JavaVM.framework/
Versions/1.3.1/Home/bin/java
export JAVA_EXECUTABLE PATH=$PATH:./lib
export PATH
exec $JAVA_EXECUTABLE -Xmx32M -classpath "./lib/EspressManager.jar:
./lib/FlashExport.jar:./lib/ExportLib.jar./lib/jaxp.jar:
./lib/parser.jar:./lib/SVGExport.jar:
./help/examples/DataSources/database/hsqldb.jar:.:"
quadbase.common.server.Server -monitor:ON -log -DBBuffer:0 -
DBCleanAll:600
```

Once you have finished the modifications, save the changes you've made to the .bat or .sh file, and re-start EspressManager.

*Note - If you're running on Mac OS X and you elected to create aliases when installing, you will need to modify the espressmanager.app package to add the hsqldb.jar to the classpath. To do this right-click (CTRL+Click) on the espressmanager.app and select 'Show Package Contents' from the pop-up menu. Then navigate to Contents/Resources where you will see a file called 'MRJApp.Properties'. Open this file in a text editor and add the .jar file to the classpath argument.

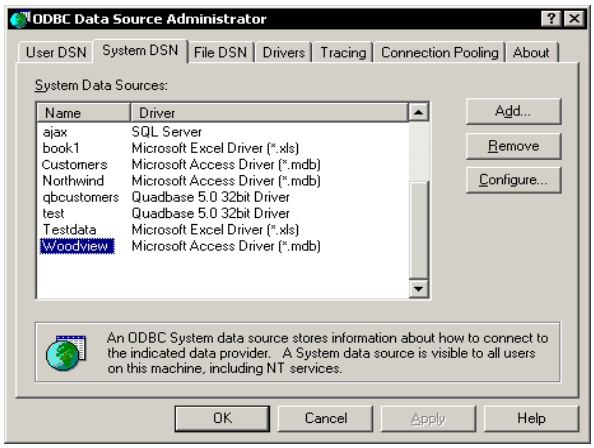
With the modified EspressManager running, re-start Chart Designer, and open your data registry again. From the Data Source Manager, click on the "Databases" note in the left-hand frame, and click the 'Add' button. A dialog will then appear prompting you to enter the connection information for the new database. Enter "Woodview" as the name of the Database, enter "jdbc:hsqldb:help/examples/DataSources/database/ woodview" for the URL, and enter "org.hsqldb.jdbcDriver" as the driver. Click on both

the 'Require Login' and 'Save Password' boxes. Then enter "sa" for the user name and leave the password blank.

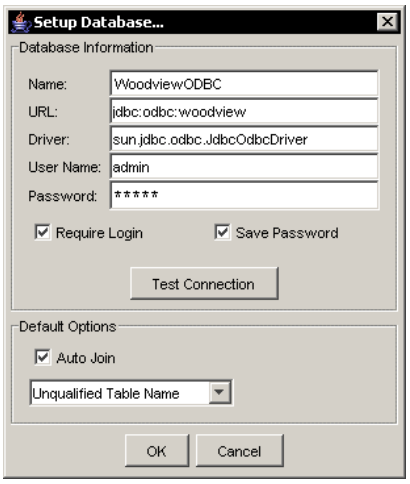


Leave the auto-join and table name properties alone, and click the 'Test Connection' button to make sure you've entered the information correctly. Then click 'OK' to bring back up the data source manager window, where there will be a new node under "Databases" for Woodview.

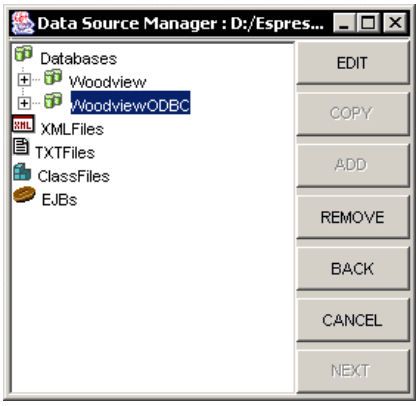
If you're running on Windows, then you can use JDBC-ODBC bridge to connect to the sample MS Access database that is included in the installation. Before setting up this database as a data source in EspressChart, you must first set up the database as an ODBC data source in your Windows environment. To do this, launch the ODBC Data Source Administrator from the Windows control panel. Then add a users or system DSN. Select Microsoft Access as the database driver, and specify the data source name as "Woodview". Then click the 'Select' button under 'Database'. Browse to the Woodview.mdb file found in the EspressChart installation under help\examples\ DataSources\database. Select this file. You should now see a new entry under the "System DSN" or "User DSN" tab called Woodview.



Once you have properly set up the DSN for Woodview, go back to the Data Source Manager. Click on the "Databases" node in the left-hand frame, and click the 'Add' button. A dialog will then appear prompting you to enter the connection information for the new database. If you are using Sun's JVM (i.e. JRE 1.3 or similar) then the default driver supplied in the dialog will work. Enter "WoodviewODBC" as the name of the database, and enter "jdbc:odbc:Woodview" for the URL. Click on both the "Require Login" and "Save Password" boxes. Then enter "admin" for both the user name and password.



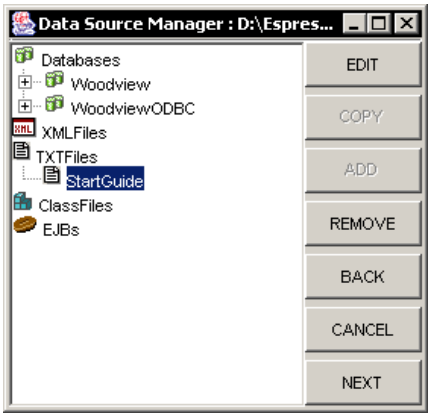
Leave the auto-join and table name properties alone, and click the 'Test Connection' button to make sure you've entered the information correctly. Then click 'OK' to bring back up the data source manager window, where there will be a new node under "Databases" for WoodviewODBC.



To add a text data source, click on the "TXTFiles" node in the left-hand frame of the Data Source Manager and click 'Add'. A dialog will appear prompting you to specify a display name for the data source, and the location of the text file. Enter any name you would like and then press the 'Browse' button. Browse to help/examples/DataSources/text/ and select StartGuide.txt.

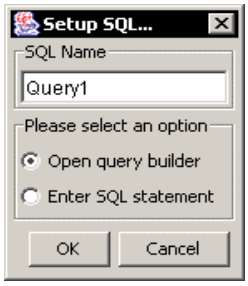
Clicking 'OK' will bring back up the Data Source Manager where you will see a new node under "TXTFiles" for the text data source that you have just entered.

There is an additional sample text file under help/examples/DataSources/text/ called Sample.dat. This expanded sample file contains data for each data type. You can use this file to explore the different chart types and options.

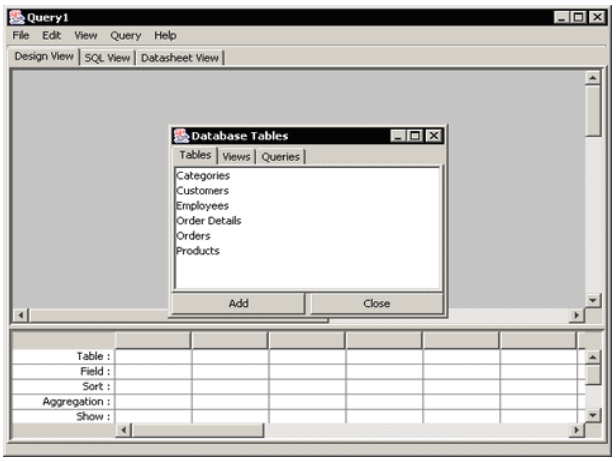


4.4 Step 4: Build a query

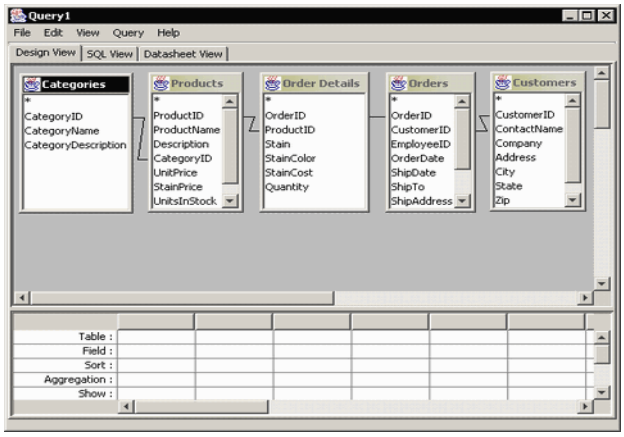
To create a new query, click to expand the "Woodview" or "WoodviewODBC" node in the left-hand frame of the Data Source Manager. Two sub-nodes will appear, one called "Queries" and one called "Data Views". Select the "Queries" node and click 'Add'. A dialog will appear prompting you to specify a name for the query, and to select whether to launch the Query Builder, or enter an SQL statement.



Enter any name you would like, select 'Open query builder', and click on 'Ok'. The Query Builder will then launch. You will see a separate window containing all of the tables for Woodview sitting over top of the main Query Builder window.

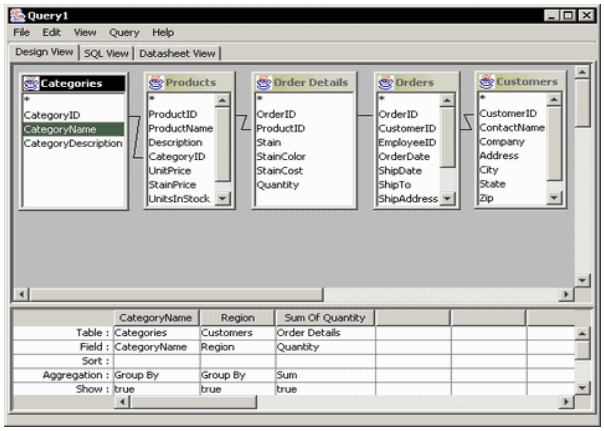


From the "Tables" tab add the Categories, Products, Customers, Order Details, and Orders tables to the query by selecting each, and clicking the 'Add' button. As the tables are added they will appear in the top half of the query builder window. The tables will be auto-joined as indicated by the black lines connecting them. Once you have added the tables, click the 'Close' button on the tables window.

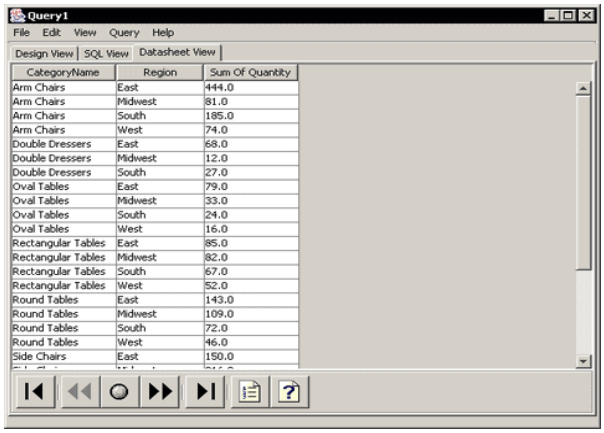


To add fields to the query you can either double-click the field from within the table window, or you can double-click on the "Table" and "Field" fields in the lower-half or QBE (query by example) portions to make selections from a drop-down menu. Using either method, add the following fields to the query: CategoryName from the Categories table, Region from the Customers table, and Quantity from the Orders table.

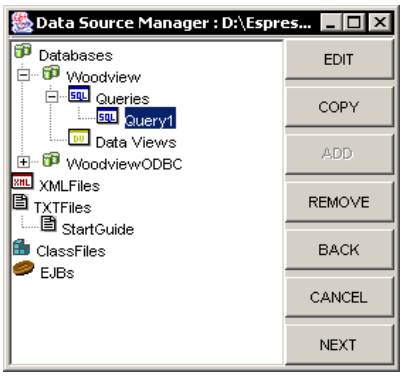
Once you have added the fields, double-click in the "Aggregation" field under the CategoryName column. Select "Group by" as the aggregation type. When you click elsewhere in the QBE frame all the other columns will default to "Group by" as the aggregation. Double-click the "Aggregation" field under the Quantity column, and change the aggregation to "Sum".



Once you have entered the fields, you can preview the results of the query by clicking on the "Datasheet View" tab. The query will run, and you will see the total sales volume broken down by product category and sales region.

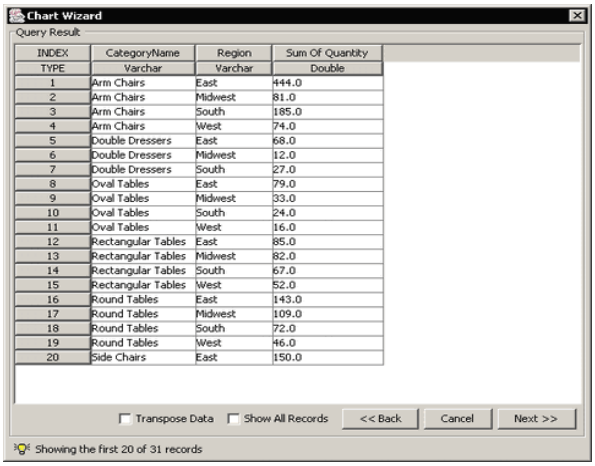


Once you have finished building the query, select 'Done' from the File menu. The Query Builder will close, and the Data Source Manager will come back up. There is now a node under "Queries" for the newly created query.



4.5 Step 5: Build a chart

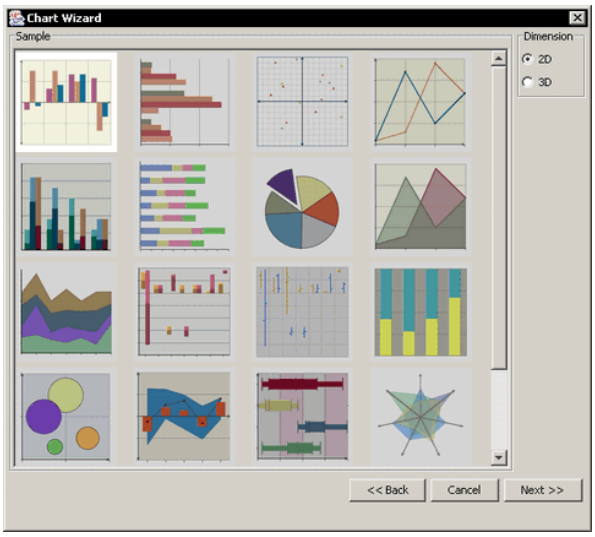
To build a chart, you must first select the data source that you would like to use. Select either the query that you created or the StartGuide text file, and click the 'Next button. A new window will open showing the first twenty records of data.



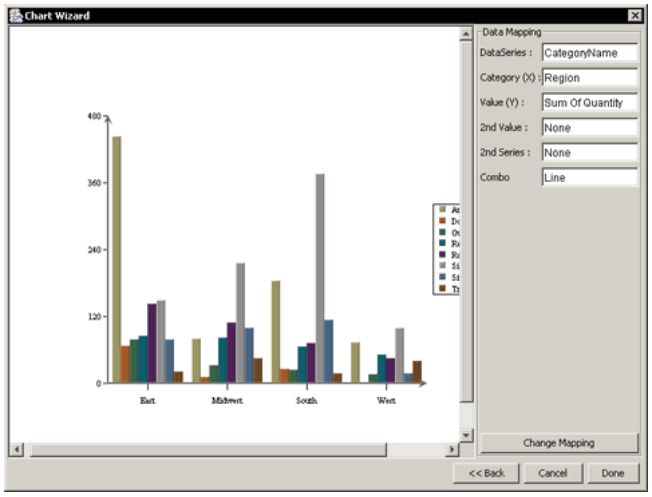
INDEX	CategoryName	Region	Sum Of Quantity
TYPE	Varchar	Varchar	Double
1	Arm Chairs	East	444.0
2	Arm Chairs	Midwest	81.0
3	Arm Chairs	South	185.0
4	Arm Chairs	West	74.0
5	Double Dressers	East	68.0
6	Double Dressers	Midwest	12.0
7	Double Dressers	South	27.0
8	Oval Tables	East	79.0
9	Oval Tables	Midwest	33.0
10	Oval Tables	South	24.0
11	Oval Tables	West	16.0
12	Rectangular Tables	East	85.0
13	Rectangular Tables	Midwest	82.0
14	Rectangular Tables	South	67.0
15	Rectangular Tables	West	52.0
16	Round Tables	East	143.0
17	Round Tables	Midwest	109.0
18	Round Tables	South	72.0
19	Round Tables	West	46.0
20	Side Chairs	East	150.0

From this dialog click 'Next' again. A dialog will appear asking you to process data, or get another data source. Select 'Process Data' and click 'Next'.

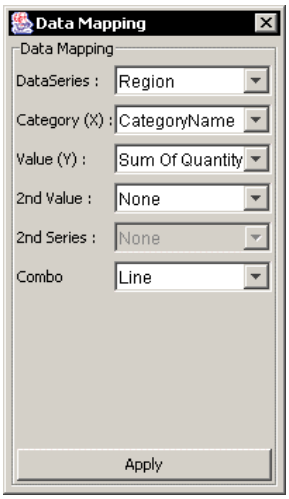
The next screen in the chart wizard allows you to select the chart type that you would like to use. You can toggle between two-dimensional and three-dimensional chart types using the radio buttons. Select a two-dimensional column chart as the type that you would like to use and click 'Next'.



The next screen allows you to set data mapping for the chart. Data mapping is the process by which the columns of data from your data source are mapped to the elements of the chart. By default the first column from left to right (CategoryName) will be mapped as the data series, the second (Region) column as the categories, and the third (Quantity) column as Value. To modify the mapping click on the 'Change Mapping' button.



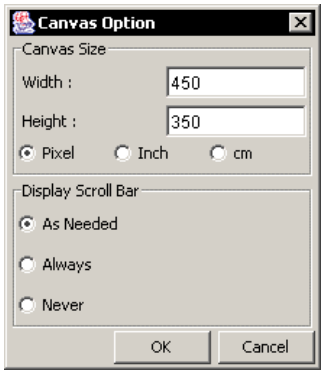
This will bring up a new window allowing you to modify the mapping options. From the drop-down lists, select Region as the data series, and CategoryName as the categories. Leave the Value option the same.



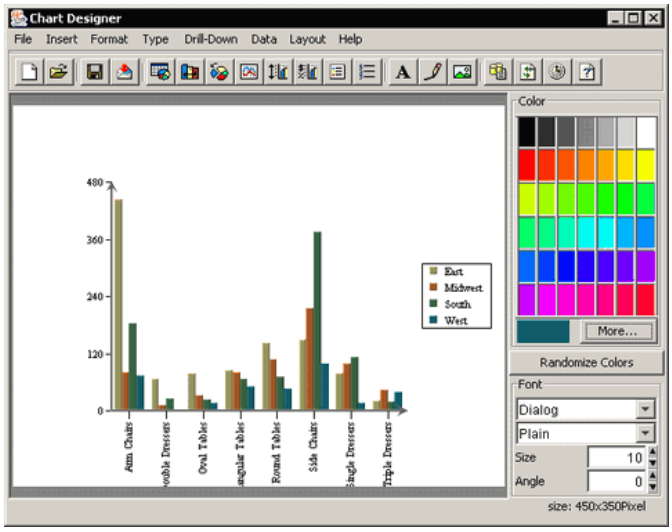
Once you have made the required changes, click on 'Apply' and the chart will be re-drawn in the mapping window to reflect the changes. Once you have finished setting the mapping options, click 'Done' and you will go to the Chart Designer window where you can customize the chart.

4.6 Step 6: Customize chart properties

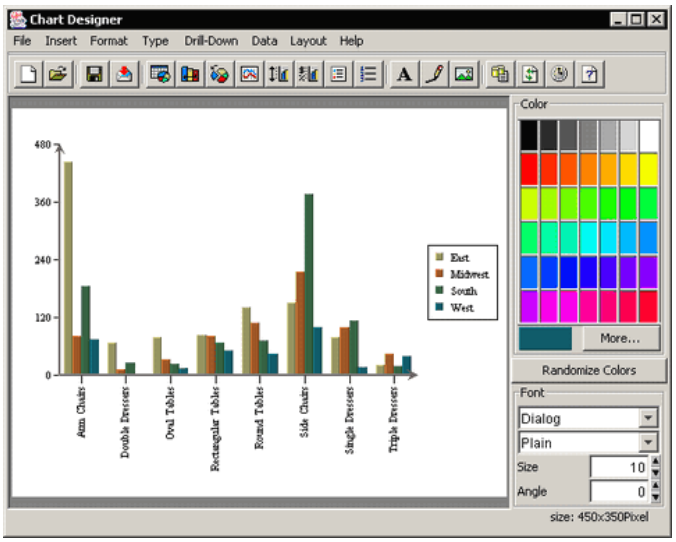
The default canvas size for charts is 600 x 500 pixels. To adjust the size of the chart canvas, select 'Canvas' from the Format menu. This will bring up a dialog allowing you to resize the chart canvas in either pixels, inches, or centimeters.



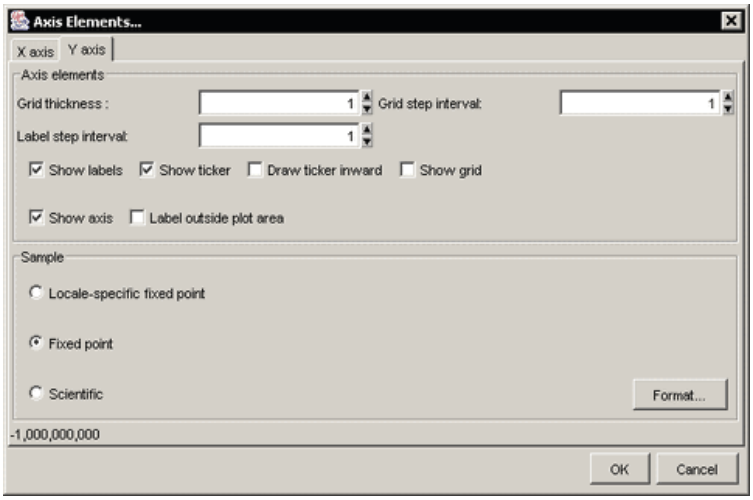
Change the canvas dimensions to 450 x 350 pixels, and click 'Ok'. The canvas will now resize in designer presenting a smaller chart image.



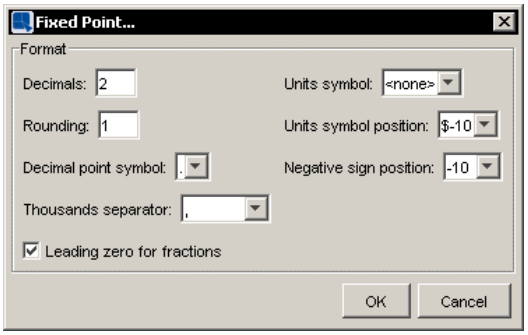
Now that the chart canvas has been shrunk the chart plot will appear small, and portions of the X-Axis labels may be truncated. This can be adjusted by re-sizing the chart. You can click and drag on the chart plot to move it, and right-click and drag to re-size it. You can also click and drag to move the legend. Use these options to position the chart plot and legend on the canvas.



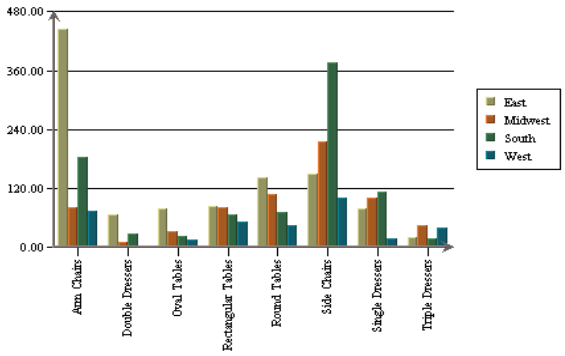
Next you can modify the format of the axis labels. To do this click the Axis Elements button on the toolbar.  This will bring up a tabbed dialog allowing you to set different options for each chart axis. Click on the "Y Axis" tab to bring up options for the value axis.



Check the box marked "Show grid" to add grid lines to the Y-Axis. Then select "Fixed point" for the data format, and click the 'Format' button. This will bring up an additional dialog allowing you to set format options for the numeric data. Select the number of decimals as 2 and click 'OK'.

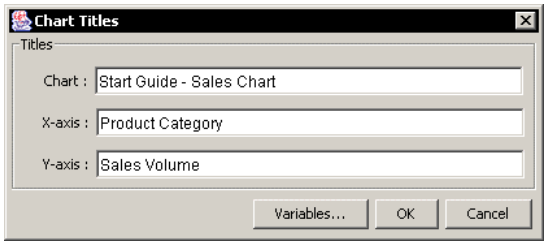


Click 'OK' again to dismiss the axis elements dialog and you will see the specified changes reflected in the chart.

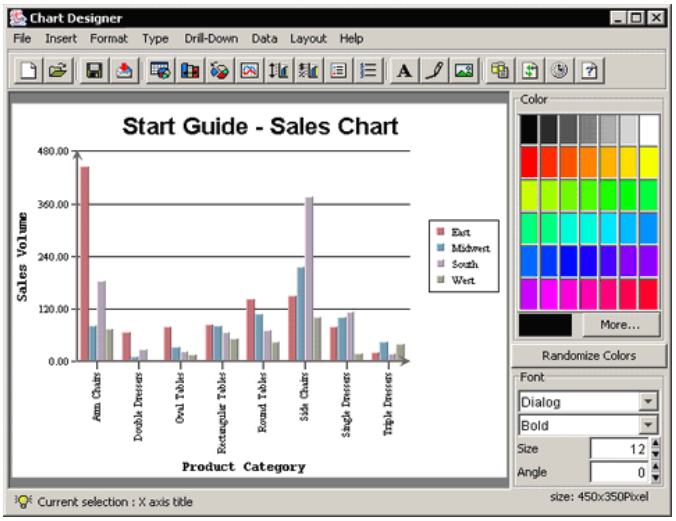


To modify the colors of the chart elements, you can click on the 'Randomize Color' button. This will cycle through various color combinations for the data elements. You can also change the color for any element by clicking to select it (the hint at the lower left-hand corner of the design window will indicate the currently selected element) and picking a new color from the color panel.

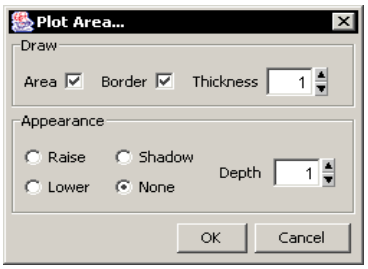
Next, you can add titles to the chart. To do this, select 'Titles' from the Insert menu. This will bring up a dialog allowing you to enter titles for the chart, as well as each of the axes.



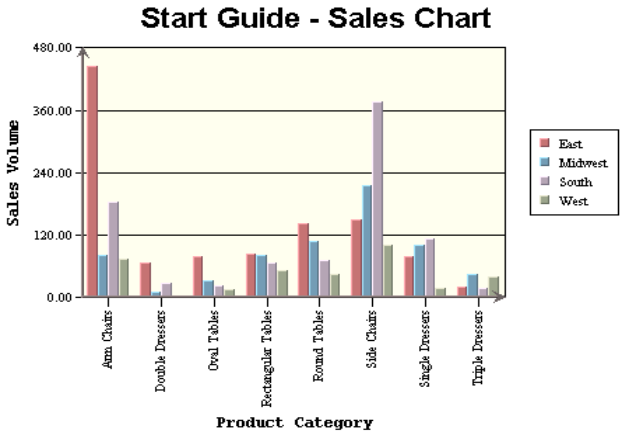
Enter any titles that you would like for the various elements and click 'OK'. The titles will then be added to the chart. Titles are placed automatically, but you can manually adjust their positions, by clicking and dragging the text on the chart canvas.



Finally, you can customize the appearance of the plot area, by adding a background and border to the plot. To do this, select 'Plot Area' from the Format menu. This will bring up a dialog allowing you to set display options for the chart plot.

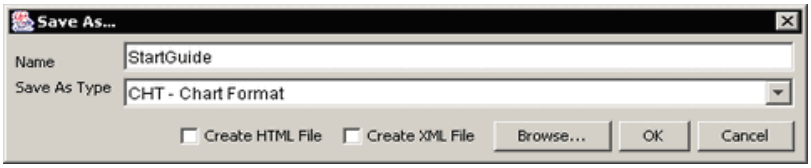


Select to draw both the plot area and the border with a thickness of 1. Specify "None" for the appearance. Once you have specified the options, click 'OK' and the plot area for the chart will be modified. You can change the background color of the plot area, by clicking to select it, and then modifying the color in the color panel.



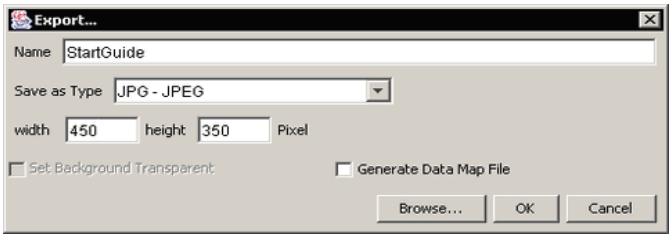
4.7 Step 7: Save and export the chart

Once you have finished customizing the chart, you can save the chart definition by selecting 'Save As' from the file menu. This will bring up a dialog prompting you to specify the file name, as well as several other save options.

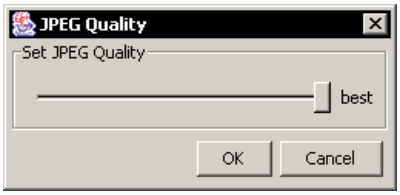


Select to save the chart in CHT format, and specify a file name for the chart. This will save the chart with its data in the root directory. You can specify a different location if you like.

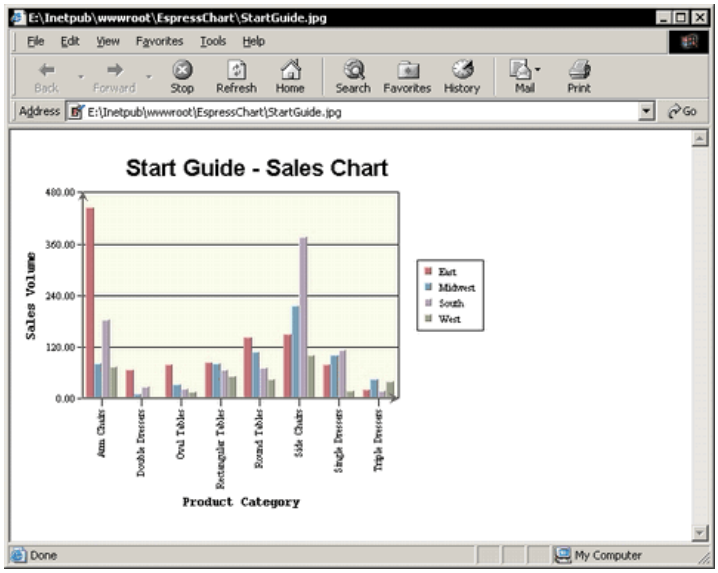
Once you have saved the chart, you can export it by clicking the Export button on the toolbar. This will bring up a dialog prompting you to specify export options for the chart.



Specify a name for the generated image file, and select JPEG as the export format. Click 'OK' and you will be prompted to specify the image quality for the generated image.



Specify the maximum quality and click 'OK'. A JPEG file will be written in the root directory of the installation. You can preview the image by opening the generated file in your Web browser.



Now you've covered some of the basic functionality of the Chart Designer. Please see the User's Guide for more detail on any of the features discussed in this guide.

5 データソースを使用する作業

チャートをデザインするには、まず初めに使用したいデータを取り込む作業を行います。チャートデザイナー内では、JDBC/ODBC に準拠したデータベースやテキストファイル、XML ファイル、EJB からデータを取り込むことができるほか、クラスファイルを通じてオブジェクト／アレイデータを取り込むことも可能です。すべてのデータソース情報はデータソースマネージャ内に保存されます。

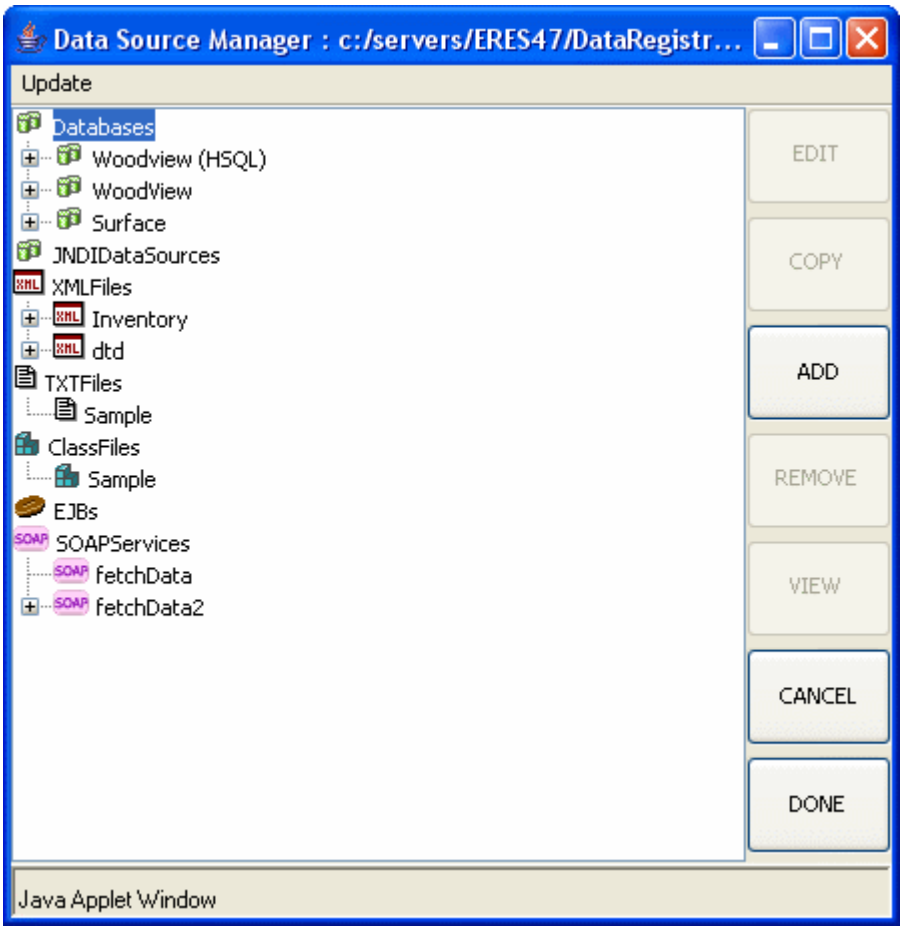
5.1 データソースマネージャ

データソースマネージャは特定のユーザによって利用されたデータソースのロケーションおよび接続情報を保存する統合ユーティリティです。この情報は XML レジストリファイル内に保存されます。設定できるデータレジストリファイルの数に制限はなく、1 つのレジストリ内に保存できるデータソースの数にも制限はありません。レジストリはデザインを行う時の補助として使用されますが、データやデータソース情報はチャートファイルと共に保存されているため、チャートを展開する必要はありません。

＊注意− XML データソースのレポジトリはロケーションおよび接続情報のみを保存し、実際のデータは保存しません。XML ファイルはチャートで使用されるデータは含みません。

5.1.1 データソースマネージャの使用

新しいチャートを開始（File メニューから'New'を選択、またはツールバー上の New Chart ボタンをクリック）したとき、またはチャートデザイナーを最初に起動したときは Chart Wizard が起動されます。Chart Wizard はチャートの基本的な構造をステップごとに作成することを可能にします。Chart Wizard の最初のダイアログで、使用するデータレジストリファイルを指定するか、または新しいレジストリを作成するよう求められます。デフォルトではデータソースレジストリファイルは DataRegistry ディレクトリに保存されています。既存のレジストリを開くかまたは新しいレジストリを開始すると、データソースマネージャのウィンドウが開きます。



データソースマネージャウィンドウ

ウィンドウの左側にはレジストリファイル内のすべてのデータソースがツリー構造でリストアップされます。"Databases"の下には個々のデータベースとそれに関連づけられたクエリおよびデータビューがグループ別に表示されます。"JNDIDataSources"の下には、JDBC の代わりに JNDI（Java Naming and Directory Interface）名を使用して接続するデータベースソースがグループ別に表示されます。"XMLFiles"の下には、すべての XML ファイルとそれに関連づけられたクエリーがグループ別に表示されます。また、"TXTFiles"の下には、指定されたすべてのテキストファイル、"ClassFiles"の下には指定されたすべてのクラスファイル、および"EJBs"の下には指定されたすべての EJB 接続がグループとして表示されます。.

ウィンドウの右側にはデータソースマネージャのすべての機能をコントロールするための一連のボタンが含まれています。各ボタンは以下の機能を実行します。

Edit: データソースの属性を修正するオプションです。データベースでは接続情報の変更やクエリ／データビューの修正を行うことができます。XML ファイルでは表示名やファイルのロケーションを変更することができます。テキストファイルでは表示名やファイルのロケーションを変更することができます。クラスファイルでは表示名やファイルのロケーションを変更することができます。EJB では表示名やパラメータ値を変更することができます。

Copy: このオプションはクエリおよびデータビューにのみ使用することができます。このオプションを使用すると指定されたクエリまたはデータビューのコピーを作成することができます。

Add: データソースを追加するオプションです。左側の部分で選択されているノードに応じた新しいソースが作成されます。例えば"TXTFiles"を選択して'Add'をクリックすると、新しいテキストファイルのデータソースを追加するようプロンプトが表示されます。

Remove: 選択されたデータソースを削除するオプションです。

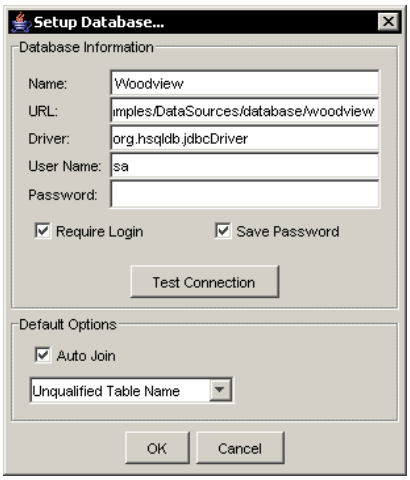
Back: このオプションを使用すると Wizard 内で一つ前のステップに戻ることができ、使用しているレジストリファイルを変更することが可能になります。.

Cancel: Wizard のプロセスをキャンセルします。

Next: 選択したデータソースをチャートに使用し、Wizard の次のステップに進むことができます。

5.2 データベースからのデータ

EspressChart は JDBC/ODBC に準拠したデータベースからデータを取り込むことができます。データベースをデータソースとして利用するには、最初に使用したいデータベースをレジストリで設定し、接続情報を指定します。データベースを追加するには、"Databases"ノードをクリックし、'Add'ボタンをクリックします。データベースの接続情報を指定するよう求めるプロンプトウィンドウが開かれますので、フィールドにデータベース名、URL、ドライバを入力します。データベースがログインを要求するかどうかを選択することもでき、ログインが要求される場合はウィンドウの下部にあるチェックボックスでユーザ名とパスワード情報を保存するよう設定することができます。ログイン情報とパスワードを保存するよう選択した場合、それらの情報を最後の 2 つのスーペースに入力することができます。'Ok'をクリックするとデータソースマネージャウィンドウに新しいデータベースが追加されます。



データベースの追加ダイアログ

EspressChart でデータベースへの接続を行うには、以下の情報を入力する必要があります。

- **URL:** この JDBC URL によって使用するデータベースのロケーションを指定します。標準的な JDBC URL は：（コロン）によって 3 つのパーツに分けられています。

  jdbc:<subprotocol>:<subname>

  JDBC URL の 3 つのパーツは以下のようになっています。

  1. jdbc− プロトコルです。JDBC URL のプロトコルは常に jdbc です。
  2. <subprotocol>− ドライバの名前またはデータベース接続メカニズムの名前で、1 つまたは複数のドライバによってサポートされます。最もよくあるサブプロトコルの名前は"odbc"で、これは ODBC データソース名を指定する URL のために使用されます。例えば、JDBC-ODBC ブリッジを通じてデータベースにアクセスするには、以下のような URL を使用します。

     jdbc:odbc:Northwind

     この例では、サブプロトコルは"odbc"、サブネームの"Northwind"はローカル ODBC データソースで、ODBC 下のシステム DSN として"Northwind"が指定されています。
  3. <subname>− データベースの認識要素です。サブネームはサブプロトコルによって様々に変更が可能で、ドライバライターが選択する内部シンタックスを伴ったサブサブネームを持つことができます。サブネームはデータベースのロケーションを得るために必要な情報を与える機能を持っています。上記の例では、ODBC が他の情報を

与えることによってサブネームは"Northwind"のみで必要な情報として機能することができます。

リモート機器上にあるデータベースに接続するには、さらに追加の情報が必要になります。例えばデータベースに会社のイントラネットを通じてアクセスする場合、JDBC URL にはネットワークのアドレスがサブネームの一部として含まれ、以下のような標準的な URL 名の形式に準ずる必要があります。

//hostname:port/subsubname

例えば会社のイントラネット上にある機器に接続するために"vpn"というプロトコルを使用するのであれば、使用される JDBC URL は次のようになります

jdbc:vpn://dbserver:791/sales
(jdbc:dbvendorname:// machineName/SchemaName に似ています。)

JDBC はデータベースのドライバに接続されるのであって、データベース自体に接続されるのではないことに注意してください。特定のドライバを指定する JDBC URL が何になるかを実際に決定するのはデータベースドライバライターです。ほとんどの場合、適切なドライバはデータベースのベンダーから提供されます。データベースドライバに接続するために必要な適切な JDBC URL について、データベースドライバのベンダーに問い合わせることをお勧めします。

- **Driver:** データベースに接続するために使用される適切な JDBC ドライバです。インストレーションに含まれる JVM（または Sun の J2SE）を使用する場合、以下のドライバ指定を使用して ODBC データソースに接続します。
  sun.jdbc.odbc.JdbcOdbcDriver

  JDBC-ODBC ブリッジを使用していない場合は、使用するデータベースに特有の JDBC ドライバ名を指定することもできます。例えば、Oracle のデータベースはドライバとして oracle.jdbc.driver.OracleDriver を必要とします。

- **User Name:** データベースで使用するログイン名です。.

- **Password:** 上記ユーザのパスワードです。

*注意 – サードパーティ製ドライバ（ODBC-JDBC ブリッジ以外）を経由してデータベースに接続するには、EspressManager がそのドライバのクラスを検出できるように EspressManager の.bat または .sh ファイルを編集する必要があります。適切なクラスまたはアーカイブを.bad または.sh ファイの"-classpath" アーギュメントに追加します。

接続情報を指定したら、'Test Connection' ボタンをクリックしてデータベース接続をテストできます。これは、提供された情報を使用して接続のテストを行い、何らかの問題があればそれを図示します。

Mac OS X 上で実行し、インストール時にエイリアスを作成するように選択した場合、espressmanager.app パッケージを修正して JDBC ドライバをクラスパスに追加する必要があります。このためには、espressmanager.app 上で右クリック（CTRL+Click ）し、ポップアップメニューから 'Show Package Contents'を選択します。次に、Contents フォルダに進みます。このフォルダに'info.plist'という名前のファイルがあります。テキストエディター内でこのファイルを開いて、適切なクラスまたはアーカイブをクラスパスアーギュメントに追加します。

ダイアログの"Default Options"部分では、Query Builder インターフェイスまたはデータビューで生成されたクエリのプロパティの一部を指定することができます。選択したテーブルを自動結合するかどうかを指定できます。自動結合では、データベース内で定義されたプライマリおよび外部キーが結合されます。また、修飾されていない（テーブル名のみ）クエリあるいは 2 または 3 部分修飾されたクエリに使用しなければならないテーブル名のフォーマットも指定できます。ここで指定されるプロパティは、新しいクエリーおよびデータビューのデフォルトになります。さらに、クエリーごとに修正することもできます。

このダイアログの"Multiple Database Options"部分では、クエリ内からデータを取得するための追加データベース（すなわち、追加データベース URL）を指定することができます。このオプションは、データベース（オリジナルおよび追加データベース）が MS SQL Server および 3-Part Qualified Table Name オプションを選択している場合のみ使用することができます。さらに、指定した追加データベースに接続するには、（オリジナルの接続で定義されているものと）同一のログイン詳細と同一のドライバを使用することに注意してください。クエリは、3 パートテーブルの名前を使用して追加データベース内のカラムを参照することによりデータを取得することができます。

EspressChart インストレーションには 2 つのサンプルデータが組み込まれています。1 つは HSQL (ピュア Java アプリケーションデータベース)で、もう 1 つは MS アクセスデータベースです。どちらにも同じデータが入っており、help/examples/DataSources/database ディレクトリにあります。これらのサンプルデータベースに対する接続のセットアップ方法の詳細については、セクション 3.3 を参照してください。

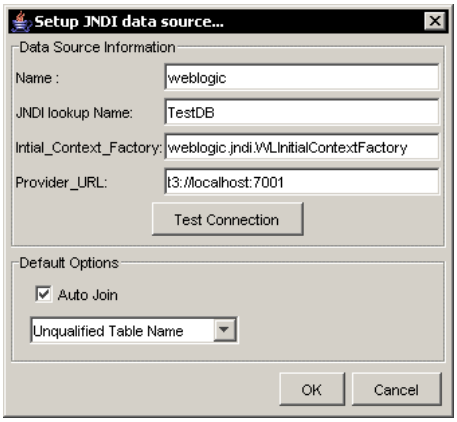
5.2.1 JNDI データソース

EspressChart は JDBC 経由でデータベースに接続されるだけでなく、JNDI（Java Naming and Directory Interface）を使ってデータソースに接続することができます。

EspressChart では、JNDI データソースはデータベースのデータソースとまったく同様にみなされ、同一の機能（クエリー、パラメータ、データビューなど）をサポートします。JNDI データソースを使用すると、環境間でのチャートの移行が簡単になるという利点があります。両方の環境のデータソースを同一のルックアップ名で設定すると、何も変更しないでチャートを移動できます。

チャートデザイナーで JNDI データソースに接続するには、Web アプリケーション環境にデータソースが配備されていなければなりません。また、同一の環境で EspressManager がサーブレットとして実行されている必要があります。EspressManager をサーブレットとして実行するための詳細については、セクション 2.3.1 を参照してください。

JNDI データソースをセットアップするには、データソースマネージャで "JNDIDataSources" ノードを選択し、'Add'ボタンをクリックします。こうすることによって、接続情報を指定できるダイアログが表示されます。



JNDI セトアップダイアログ

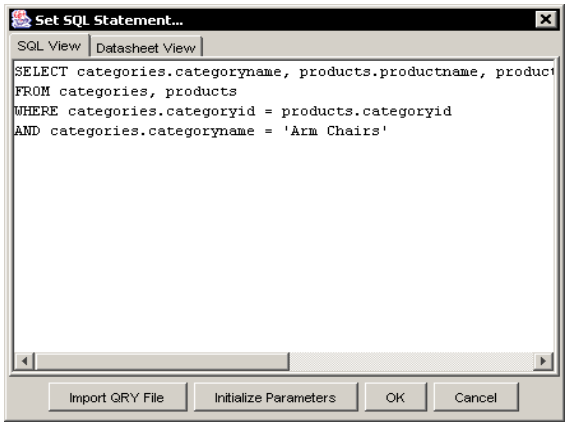
最初のオプションでは、データソースの表示名を指定できます。2 番目のオプションでは、データソースの JNDI ルックアップ名を指定できます。3 番目のオプションでは、データソースのイニシャルコンテキストファクトリを指定でき、最後のオプションでは、プロバイダーURL を指定できます。JNDI データソースのインプリメント方法はベンダーによって異なるため、この情報は、ご使用のアプリケーションサーバーによって異なります。Test Connection' ボタンをクリックすると、接続をテストすることができます。

5.2.2 クエリー

データベースを追加すると、そのデータベースのための新しいノードがデータソースマネージャウィンドウに表示されます。ノードを展開するとさらに"Queries（クエリー）"および"Data Views（データビュー）"という 2 つのノードが現れます。この 2 つによって使用するデータベースからデータを取り込むことができます。新しいクエ

リーを作成するには、"Queries"ノードを選択して'Add'ボタンをクリックします。ダイアログが表示されますので、クエリー名を指定し、SQL ステートメントをテキストとして入力するか、または Query Builder（クエリービルダー）を起動するかを選択します。

SQL ステートメントを入力するよう選択した場合、ダイアログボックスが表示されますのでそこに SQL ステートメントを入力します。このダイアログからは SQL テキストを含んでいるテキストファイルまたは QRY を読み込むか、あるいは既に保存されている手順を実行することも可能です。クエリービルダーを起動するよう選択した場合、新しいウィンドウでクエリービルダーが開いて図や表を使ってわかりやすくクエリーを構築することができるようになります。クエリーの構築または入力が完了するとデータソースマネージャウィンドウに戻り、使用するデータベースの"Queries"ノードの下にクエリが新しいエントリとして表示されます。



SQL ステートメント入力ダイアログ

このインターフェースを使用して Microsoft Excel のスプレッドシートからデータを取り込むこともできます。これを行うにはまず現在の環境で ODBC DSN を設定し、使用したい Excel ファイルを指定します。次に DSN を指定するデータソースマネージャにデータベースを追加します。ここで以下の SQL ステートメントを入力することによって Excel ファイルのクエリを行うことができます。

`Select * from "SheetName$".` (SheetName には Excel シートの名前が入ります。)

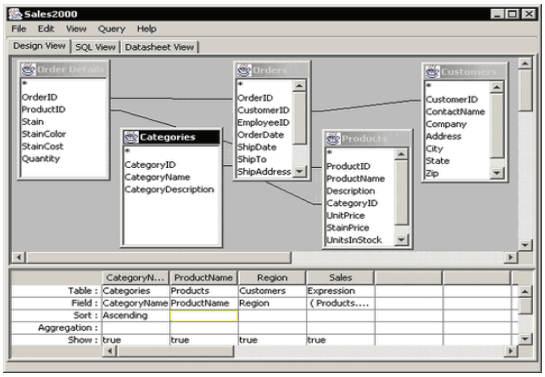
このクエリを実行すると指定したワークシート全体が返されます。

5.2.2.1 クエリービルダーの使用

クエリービルダーはリレーショナルデータベースに対するクエリの構築をビジュアル環境で行うことを可能にする統合ユーティリティです。クエリビルダを起動するには、データソースマネージャの中に新しいクエリーを追加し、'Open Query Builder（クエリービルダーを開く）'オプションを選択します。新しいウィンドウでクエリービル

ダーが開きます。既にあるクエリーを修正するためにクエリービルダーを使用することもでき、この場合はデータソースマネージャ内でクエリー名をダブルクリックします。

クエリービルダーのメインウィンドウは 2 つの部分によって構成されています。ウィンドウの上半分はそのクエリーのために選択されたすべてのデータベーステーブル、および関連するカラムを含んでいます。またこの部分にはカラム間に設定されているジョインも表示されます。ウィンドウの下半分は QBE（query by example）です。このウィンドウにはクエリーのために選択または構築されたカラムと、それに関連するコンディションが含まれています。



クエリービルダーウィンドウ

クエリビルダウィンドウの上部には 3 つのタブがあります。これらのタブを選択することによって異なるビューに切り替えることができます。"Design View（デザインビュー）"タブは先に解説したメインデザイナーウィンドウを表示します。"SQL View（ＳＱＬビュー）"タブは現在のクエリによって生成された SQL ステートメントを表示します。"Datasheet View（データシートビュー）"タブはクエリの結果を表示します。

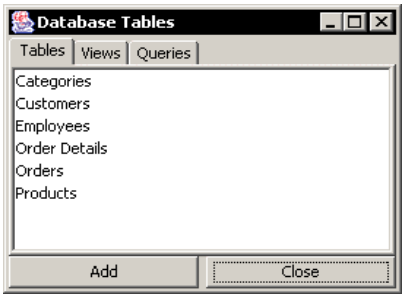
クエリの構築が終了したら、File（ファイル）メニューから'Done（完了）'を選択してデータソースマネージャに戻ります。.

5.2.2.1.1 テーブル

クエリービルダーを最初に起動すると、データベース内のすべてのテーブルのリストを含んだタブ付きのウィンドウが表示されます。2 番目のタブはデータベース内のすべてのビューのリストを表示します。3 番目のタブはデータベースのためにデザインされたその他のクエリーのリストを"Queries"というヘッディングの下に表示します。このウィンドウからクエリーを構築するために使用するテーブル／ビューを選択することができます。既にデザインされたクエリをテーブルとして読み込むこともできます。テーブルを追加するには、テーブルを選択して'Add'ボタンをクリックするか、テーブル名をダブルクリックします。テーブルが追加されるとメインクエリビルダウィンドウに表示され、そのテーブル内にあるすべてのカラムが表示されます。テーブルを削除するには、テーブル内で右クリックしてポップアップメニューから'Delete'

を選択します。テーブルのエイリアスを指定し、このメニューからフィールドをアルファベット順にソートすることもできます。'Close'ボタンをクリックするとテーブルウィンドウを閉じることができます。テーブルウィンドウを再び開くには'Query（クエリー）'メニューから'Show Tables（テーブルの表示）'を選択します。

*注意– デフォルトではテーブルは無効な名前で表示されますが、Query（クエリー）メニューから'Table Name Format...'を選択して 2 パートまたは 3 パートの有効な名前に変更することができます。



クエリービルダーテーブルウィドウ

5.2.2.1.2 結合

クエリーを作成するためのデータベーステーブルを選択すると、クエリービルダーがデータベースの primary key（プライマリキー）と foreign key（フォーリンキー）の関係に基づいてカラムフィールド間の結合を自動的に検知します。自動結合が行われると標準の結合がテーブル間に作成されます。結合はデザインウィンドウの上部の中で 2 つのフィールド間に描画される線で表されます。結合を削除するか、または結合のプロパティを編集するには、線上で右クリックして任意の項目をポップアップメニューから選択します。結合を追加するには、カラムフィールドをクリックして他のテーブルのカラムフィールドにドラッグします。自動ジョインの機能を不使用にしたい場合は Query（クエリー）メニューから'Auto Join（自動結合）'を選択します。

Join Properties（結合プロパティ）：ポップアップメニューから'Join Properties（結合プロパティ）'を選択すると 3 つのオプションが表示され、カラムフィールド間で使用する結合のタイプを選択することができます。クエリービルダーは均等結合のみをサポートします。非均等結合を行う場合は"conditions（条件）"フィールドを使用し、内部結合、左外部結合び右外部結合を指定することができます。以下の例で各結合のタイプを解説します。

例として Customers および Orders という 2 つのテーブルを使用します。

CustomerID	CustomerName
1	Bob
2	Ivan
3	Sarah
4	Randy

5	Jennifer

OrderID	CustomerID	Sales
1	4	$2,224
2	3	$1,224
3	4	$3,115
4	2	$1,221

2 つのテーブル上の **CustomerID** にある内部結合によって、**Customers** テーブルの列と **Orders** テーブルの列が組み合わされ、**Customers** テーブルの各列が **Orders** テーブルの適合する **CustomerID** 値を持つすべての列と「結合」されます。**Orders** テーブルの適合する **CustomerID** フィールドを持たない **Customers** テーブルの列はクエリー結果のセットには含まれません。

ここで **CustomerID** フィールドの内部ジョインを伴う **OrderID**、**CustomerName**、および **Sales** フィールドを選択してクエリーを作成したとします。この場合、クエリービルダーによって生成される選択ステートメントは以下のようになります

```
Select Orders.OrderID, Customers.CustomerName, Orders.Sales
From Customers, Orders
Where Customers.CustomerID = Orders.CustomerID
Order by Orders.OrderID;
```

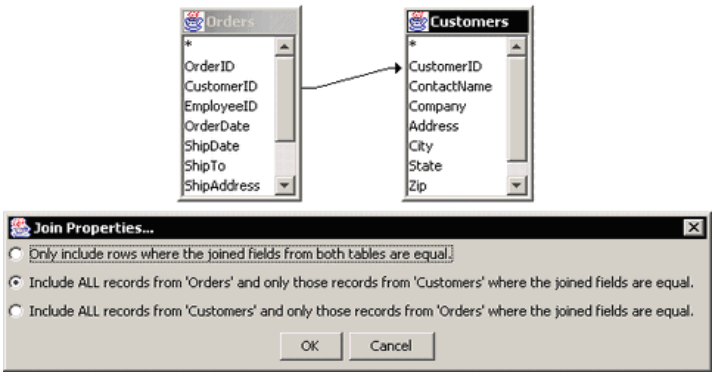
クエリーの結果は以下のようになります。

OrderID	CustomerName	Sales
1	Randy	$2,224
2	Sarah	$1,224
3	Randy	$3,115
4	Ivan	$1,221

このように、**CustomerName** で入力されている"Bob"と"Jennifer"は結果セットに表示されていません。これはこれらのカスタマーがオーダーを行っていないためです。適合する記録が関連するテーブル（この場合は **Orders** テーブル）に存在しない場合でもすべての記録（この場合はカスタマー名）を含めておきたいときは、外部結合を使用することによってこれを行うことができます。

クエリービルダーでは右または左の外部結合をオプションとして選択することができます。「右」または「左」はクエリービルダー内でテーブルが選択される順序によって決定されるに過ぎず、重要な意味を持つものではありません。外部テーブル（適合する結合の有無に関わらず含まれるすべての記録を持ったテーブル）が最初に選択された場合、クエリービルダーは右外部結合を使用します。外部テーブルが他の結合テーブルの後から選択された場合は、左外部結合が使用されます。この例では **Orders** テーブルの前に **Customers** テーブルが選択されています。そのためクエリービルダ

ーは **CustomerID** フィールドで右外部結合を使用し、**CustomerName** フィールドのすべての記録を選択します。



結合プロパティのダイアログ

ここで前出の例を使用して、前回と同じクエリを作成します。ただし今回は **Customers** からの記録をすべて含むよう指定します。クエリービルダーによって生成される選択ステートメントは以下のようになります。

```
Select Orders.OrderID, Customers.CustomerName, Orders.Sales
From Orders right outer join Customers on Orders.CustomerID =
Customers.CustomerID
Order by Orders.OrderID;
```

新しいクエリーの結果は以下のようになります。

OrderID	CustomerName	Sales
	Jennifer	
	Bob	
1	Randy	$2.224
2	Sarah	$1.244
3	Randy	$3.115
4	Ivan	$1,221

以上のように、すべてのカスタマー名が選択され、適合する記録がない結果セットには null 値が挿入されています。外部結合を選択すると、クエリービルダー内で 2 つのテーブルを接続する結合線は結合の方向を示す矢印として表示されます。

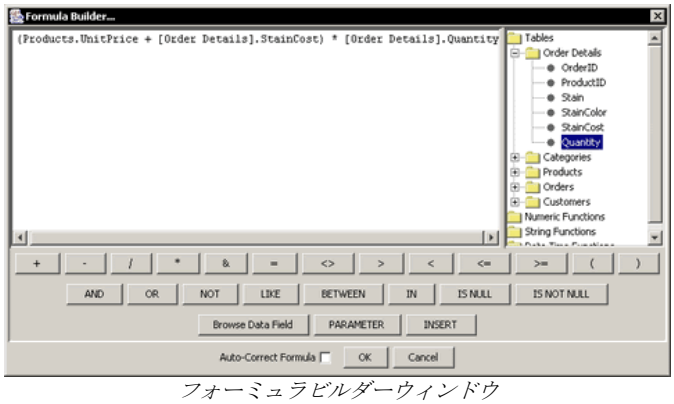
5.2.2.1.3 カラム

QBE ウィンドウは現在のクエリーのために選択されたカラムフィールドの情報と、その選択に関わるコンディション（条件）を表示します。

カラムフィールドの選択: 次の 2 つの方法のいずれかによって、テーブルからクエリにカラムフィールドを追加することができます。テーブル内でクエリに追加したいフィールド名をダブルクリックするか、あるいは"Table"または"Field"フィ

ールドをダブルクリックしてドロップダウンメニューを表示し、そこからフィールドを選択します。クエリからカラムを削除するには下部ウィンドウを右クリックしてポップアップメニューから'Delete Column'を選択するか、Edit メニューから'Delete Column'を選択します。カラムフィールドを選択したら、"Sort"フィールドをクリックすることによって昇順または降順にカラムのソート方法を指定することができます。グループ化またはカラムの集計を指定するには、"Aggregation"フィールドをダブルクリックします。集計方法のオプションとして group by（グループ化）、sum（合計）、average（平均）、min（最小）、max（最大）、count（カウント）、standard deviation（標準偏差）、variance（分散）、first（最初）、last（最後）を選択することができます。あるカラムにグループ化を選択した場合、他のすべてのカラムにもグループ化（または集計）を選択する必要があります。カラムのエイリアスを指定するには、カラムを右クリックしてポップアップメニューから'Alias'を選択します。

カラムの作成: カラムを作成するには、QBE ウィンドウ内の空白のカラムを右クリックします。ポップアップメニューから'Build'を選択するとフォーミュラビルダが起動されます。フォーミュラビルダを使用すると、選択したテーブルと使用中のデータベースのフォーミュラライブラリを使用してビジュアル環境でカラムを作成することができます。



フォーミュラビルダーウィンドウ

コンディション(条件): "Condition（条件）"または"Or"フィールドに条件を入力すると、クエリー選択のコンディション（条件）を指定することができます。"Condition（条件）"フィールドに入力された条件は、生成される SQL 内に AND 項を作成します。"Or"フィールドに入力された条件は、SQL 内に OR 項を作成します。いずれかのフィールド内で右クリックしてポップアップメニューから'Build'を選択すると、フォーミュラビルダーが起動されます。フォーミュラビルダーでは標準的なコンディション（条件）である=、<、>、BETWEEN、LIKE、NOT などが指定できるほか、クエリーをフィルタリングするためのフォーミュラを作成することもできます。ここでクエリーパラメータを指定することも可能です。

*注意- EspressChart はクエリーのコンディションとして入力されたアイテムを自動修正し、フィールド名に適切な文字列を追加すると共にストリングアーギュメントを引用符で囲みます。例えば"= ARC"と入力すると、EspressChart はクエリーコンディション（条件）を"Categories.CategoryName = 'ARC'"と修正します。複雑な機能を使用している場合（例えばデータベース機能が複数のストリングアーギュメントを使用する場合）、EspressChart は自動修正を適切に実行できない可能性があります。この場合はフォーミュラビルダーウィンドウの底部にあるチェックボックスのチェックを外すと、自動修正の機能を不使用にすることができます。

5.2.2.1.4 データベースファンクションの使用

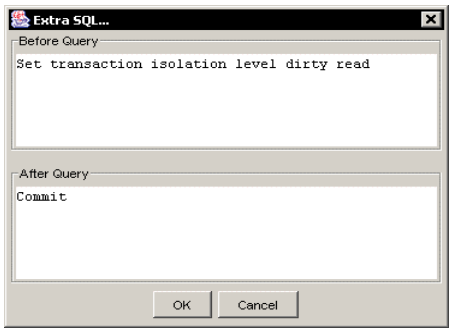
クエリーのためのカラムまたはコンディション（条件）を作成する際、クエリービルダー内のフォーミュラビルダーコンポーネントによってデータベース特有の機能を使用することができます。EspressChart が提供する機能を使用することも、あるいはユーザ独自の機能をインターフェースに追加することもできます。

EspressChart には Oracle、Access、MS SQL、および DB2 のファンクション（機能）ライブラリがあらかじめ搭載されています。これらのライブラリは userdb ディレクトリ内の DatabaseFunctions.xml ファイルに XML フォーマットで保存されています。XML で保存されていない機能を伴うデータベースの場合、EspressChart はデフォルトの機能を使用します。保存されている XML ファイルを編集して異なるデータベース機能を指定するか、または userdb ディレクトリ内の DatabaseFunctions.dtd ファイルをベースに新しいファンクションを作成することもできます。サンプルのデータベースファンクションのファイルは以下のようになります

```
<DatabaseFunctions>
   <Database ProductName="ACCESS">
      <FunctionSet Name="Numeric Functions">
         <Function>Abs(number)</Function>
         <Function>Atn(number)</Function>
      </FunctionSet>
   </Database>
</DatabaseFunctions>
```

5.2.2.1.5 Extra SQL の追加

時には extra SQL ステートメントを追加してクエリーの前或いは後に実行させる必要が出てきます。例えば、クエリー実行前にトランザクションレベルのセットが必要になるかもしれないし、ストアしたプロシージャーを呼び出さなくてはならないことがあるかもしれません。また、クエリー実行後にとトランザクションを commit したりテンポラリーテーブルを drop する必要があるかもしれません。クエリービルダーを使ってこれらの extra SQL ステートメントを指定するすることができますが、それには Query メニューの'Extra SQL' を選択します。そうするとウィンドウが立ち上がり、クエリーの前または後に実行するステートメントを書き込むことができます。



Extra SQL ダイアログ

Before Query または After Query のボックスで、それぞれ実行したい SQL ステートメントを入力することができます。入力し終えたら、'Ok'をクリックするとそのステートメントがクエリーに追加されます。

5.2.2.1.6 クエリのアウトプット

"SQL View"および"Datasheet View"タブを切り替えることによって 2 つの異なるビューでクエリを表示することができます。

SQL View: "SQL View"タブを選択すると、クエリによって生成された SQL ステートメントをデザインビューで表示することができます。これによってクエリビルダが異なるオペレーションを SQL に変換する方法を確認することができます。生成された SQL を編集することは可能ですが、SQL を変更してから'Design View'に戻ると変更は失われてしまいます。SQL を変更した後にクエリを保存すると、編集を行うためにクエリを選択した時にクエリが再度"SQL View"で開きます。

Datasheet View: "Datasheet View"タブを選択すると、クエリの結果がデータテーブルの形で表示されます。Datasheet View はクエリの実行の結果として得られるすべてのデータを表示します。Datasheet View はクエリのテストを行うため、デザインのエラーをチェックすることもできます。クエリの結果をナビゲートするにはウィンドウの底部にあるツールバーを使用します。

 データテーブルの最初のページへ移動

 データテーブルの前のページへ移動

 指定されたデータ行へ移動（0 で始まる列）

 データテーブルの次のページへ移動

 データテーブルの最後のページへ移動

 1 ページに表示する列の数を設定（デフォルトは 30）

Exporting Queries: クエリのエキスポートは 2 つの方法で行うことができます。1 つは SQL ステートメントをテキストで出力する方法、もう 1 つはクエリ結果を CSV ファイルで出力する方法です。ファイルをエキスポートするには'File'メニューから'Export'を選択します。2 番目のメニューで'Generate SQL'または'Generate CSV'オプションが表示されますので、使用するオプションを選択します。ダイアログボックスが表示されますので、ファイル名と場所を指定します。

＊注意- クエリを保存してクエリビルダを終了するには'File'メニューから'Done'を選択します。

5.2.2.2 パラメータ化されたクエリ

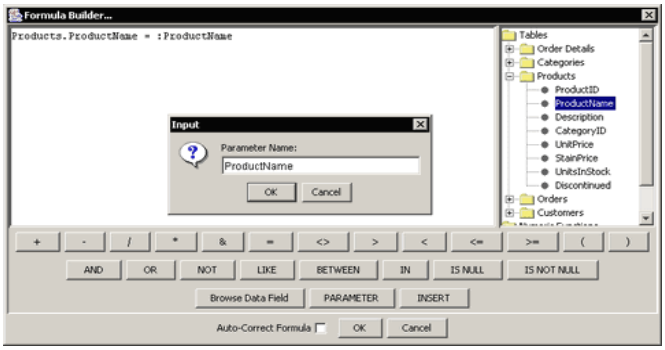
クエリビルダを使用してパラメータ化されたクエリをデザインすることもできます。この機能を使用すると実行時にデータのフィルタリングを行うことができます。パラメータ化されたクエリはチャート内のパラメータドリルダウンに使用することもできます。この機能についての詳細はセクション 8.3 を参照してください。

クエリパラメータは SQL ステートメントを入力する時、またはクエリビルダを使用する時に定義されます。また、データビューを実行する時にも定義が行われます（これについては次のセクションで解説されます）。パラメータは SQL 内で":"キャラクタによって指定されます。パラメータは一般的に SQL Select ステートメントの WHERE クローズに置かれます。例えば、以下の SQL ステートメントは"Name"というパラメータを指定します：

```
Select * From Products Where ProductName = :Name
```

これで実行時に製品名（ProductName）を入力すると、その製品のデータのみを入手することができます。

クエリパラメータの指定はクエリビルダ内で"Condition"フィールドを右クリックし、ポップアップメニューから'Build'を選択することによって行うことも可能です。フォーミュラビルダが開き、カラムにコンディションを配置することができます。



フォーミュラビルダでのパラメータの指定

パラメータを挿入するには'PARAMETER'ボタンをクリックします。パラメータの名前を指定するためのダイアログが表示されますので、パラメータ名を入力して'OK'をクリックし、もう一度'OK'をクリックしてフォーミュラビルダを閉じます。ひとつのクエリに対して指定できるパラメータの数に制限はありません。

5.2.2.2.1 マルチバリューパラメータ

EspressChartt は シングルバリューとういよりインプットとしてバリューのアレイ（配列）を取る特殊なパラメータをサポートしています。マルチバリューパラメータは、ユーザがいくつあるかわからないバリューをベースにリザルトセットをフィルターしたいときに便利です。例えば、チャートが特定の州や省のカスタマリストを返すために実行されているときなどです。ユーザは多数の異なる州や省を選び、妥当な/関連のある情報を返します。

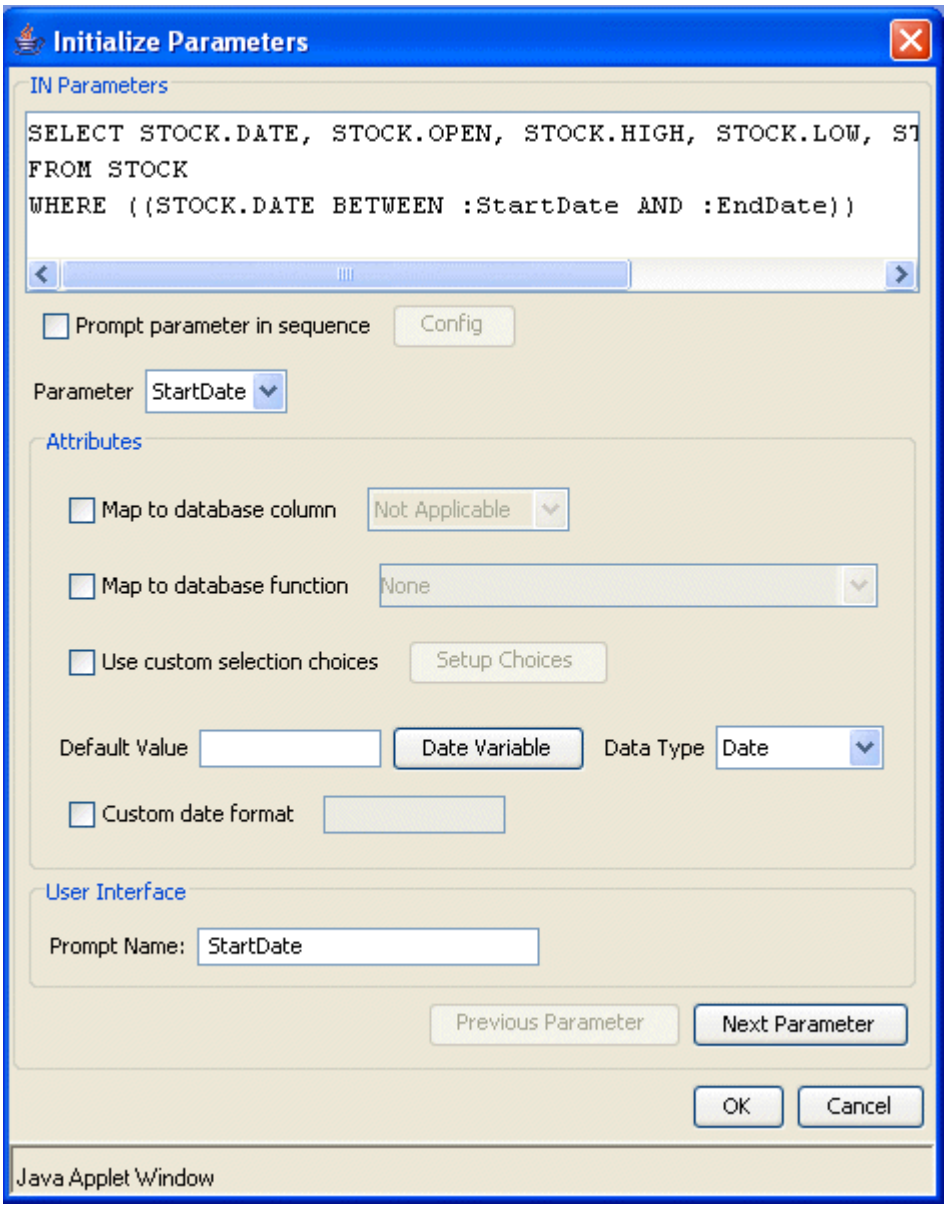
マルチバリューパラメータを作成するには、パラメータを SQL ステートメントの IN クローズ（）のカッコの中に入れます。例：

```
Select Customers.Company, Customers.Address, Customers.City,
Customers.State, Customers.Zip
From Customers
Where Customers.State IN (:State);
```

上記のクエリーは"State"という名前のマルチバリューパラメータを作成します。IN クローズ（）の中にパラメータがひとつだけのときマルチバリューパラメータは作成されます。IN クローズ（）にひとつ以上のパラメータを入れると、例えば次のような場合は、3 つのシングルバリューパラメータが作成されます。`Customers.State IN (:State1, :State2, :State3).`

5.2.2.2.2 クエリパラメータの初期化

パラメータ化されたクエリの保存（File メニューから'Done'を選択）またはプレビュー（"Datasheet View"タブをクリック）を行う場合、最初にパラメータを初期化するよう求められます。パラメータを初期化するには、Query メニューから'Initialize Parameters...'を選択するか、または Enter SQL Dialog で'Initialize Parameters'ボタンをクリックします。



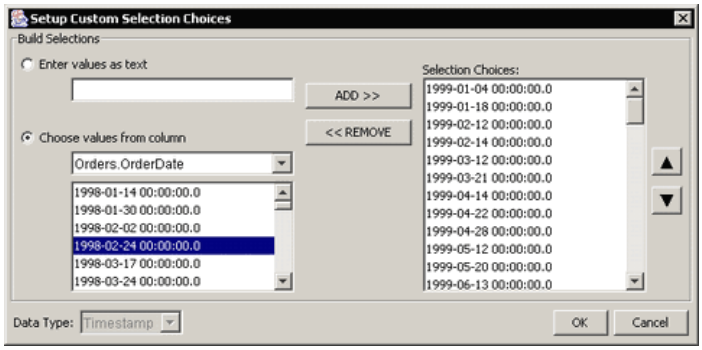
パラメータの初期化ダイアログ

このダイアログから次のオプションが指定できます。

Map to database column（データベースカラムへのマップ）：パラメータ入力に使用される値が入っているデータベースのカラムを指定できます。このオプションを選択すると、チャートビューワでチャートをプレビューまたは実行しているときにエンドユーザが取得するパラメータプロンプトを変更されます。パラメータをデータベースのカラムにマッピングすると、個別の値のドロップダウンリストが表示され、ユーザに対してそのリストからパラメータ値を選択するように要求してきます。マッピングしない場合、ユーザはパラメータ値を個々に入力しなければなりません。

*注意 –通常、このドロップダウンリストは、選択された個別の値をカラムに対して実行すると同時にクエリーの結合および条件を適用することによって埋められます。制約なしにカラムからすべてのデータを取得したい場合（こうすることによって、パラメータプロンプトのパフォーマンスが向上する場合があります）、EspressManager の スタート時に、"-singleTableForDistinctParamValue" アーギュメントを設定できます。EspressManager コンフィギュレーションオプションの詳細については、セクション 2.3 を参照してください。

Use custom selection choices（カスタムセレクションチョイスを使用）：個々のカラム値をすべてドロップダウンメニューで表示するのではなく、パラメータ値のカスタムリストを作成して、そこからエンドユーザに選択させることができます。リストを設定するには、このオプションを選択したあと、'Setup Choices' ボタンをクリックします。こうすることによって、新しいダイアログが起動され、選択肢のリストを作成することができます。

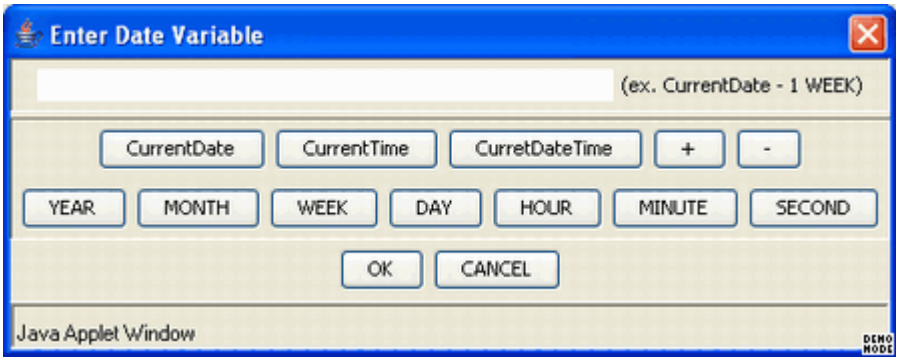


カスタムパラメータリストダイアログ

このダイアログでは、カスタム値をカンマで区切られたテキストとして入力するか、またはデータベースのカラムの個々の値から値を選択します。リストの値を指定したら、'OK'をクリックして選択肢を保存します。

Default Value（デフォルト値）:パラメータのデフォルト値を指定します。デフォルト値を指定する必要はありませんが、指定することを推奨します。デフォルト値を指定しない場合、データソースが存在しないときにチャートテンプレートを開いたり、操作したりすることができません。

Date Variable（日付変数）：このオプションは、パラメータをデータベースのカラムまたは関数にマップしない場合、または SQL 結果セットにマップして、カスタムセレクションチョイスを設定しない場合にのみ使用できます。このオプションは、日付/時刻の変数タイプを持つパラメータだけに使用できます。このオプションをクリックした場合、下図のパネルがポップアップし、サポートされるすべてのキーワードがリストされます。



Enter Date Variable（日付変数入力）ダイアログ

このダイアログでは、*CurrentDate*、*CurrentTime* および *CurrentDateTime* の 3 つのキーワードのいずれか 1 つを選択することができます。何らかの単位の時間を現在の日付/時刻に加算するか、またはそこから減算すると、日付の範囲を動的に設定することができます。たとえば、チャートに以下のデフォルト値を設定することができます。

```
StartDate: CurrentDate - 1 WEEK
EndDate: CurrentDate
```

これは、チャートが実行されるたびに、デフォルトのプロンプトが、現在の日付から 1 週間前でなければならないことを示します。これ以外にサポートされる単位としては、*YEAR*（月）、*MONTH*（月）、*DAY*（日）、*MINUTE*（分）および *SECOND*（秒）があります。この機能は、1 つの加算または減算だけをサポートします。この機能は、複数値パラメータをサポートしません。

Data Type（データ型）：パラメータ値のデータ型を指定できます。パラメータをカラムにマッピングした場合、データ型は自動的に設定されます。

Custom Date Format（カスタム日付形式）：入力すべき日付パラメータの形式を設定します。このオプションは、パラメータをカラムにマッピングしてないとき、またはカスタムセレクションチョイスを入力した場合（すなわちエンドユーザが日付値を入力する場合）のみ有効です。このオプションをチェックすると、

時間エレメントを表示する文字を組み合わせた形式で日付を入力することができます。次の表は使用できる文字の組み合わせを示しています。

文字	意味	出力 (テキスト/数字)	例
G	紀元	テキスト	AD
Y	年	数字	1996, 96
M	月	（長さに応じて）テキストまたは数字	July, Jul, 07
D	日	数字	10
H	時（午前または午後） (1-12)	数字	1
H	24 時間制の時(0-23)	数字	18
m	分	数字	30
s	秒	数字	55
S	ミリ秒	数字	978
E	曜日	テキスト	Tuesday, Tue
D	日数	数字	189
F	当該月の何週目の曜日か	数字	2 (as in 2nd Wed. in July)
W	当該年の何週目か	数字	27
W	当該月の何週目か	数字	2
A	午前/午後	テキスト	AM, PM
K	24 時間制の時 (1-24)	数字	24
K	時（午前または午後） (0-11)	数字	0
Z	時間帯	テキスト	Pacific Standard Time, PST

これらの文字を自由に組み合わせて、好みに応じた形式で日付を表すことができます。エレメントが使用する書式は、グループ内の文字数によって決定されます。テキストエレメントの場合、グループ内の文字数が 4 文字以上であれば、エレメントの完全な書式が使用されます。文字数が 4 文字未満の場合、省略形があればそれを使用します。例えば EEEE は”Monday”を返し、”EE”は”Mon”を返します。Month（月）の場合は、M はテキストあるいは数字で表示できるので、グループ内の文字数が 4 文字以上であれば完全形で、3 文字であれば省略形で、2 文字未満であれば数字で表示されます。

数値エレメントの場合、文字数が、エレメントが使用する最小桁数です。数値の方が短い場合は前にゼロを付けます。例えば、日付の日にちが 2 の場合、dd は "02" を返し、d は"2".を返します。

a-z や A-Z 以外の文字、例えば、";"（セミコロン）、 ":"(コロン), "@"（アットマーク）などは、文字列式内のどこにでも挿入でき、入力したとおりに表示されます。また、単語および式をシングルクォテーションで囲んで挿入することもできます(アポストロフィーをテキストとして 挿入するには、2 つのシングルクォテーションを入力します。)

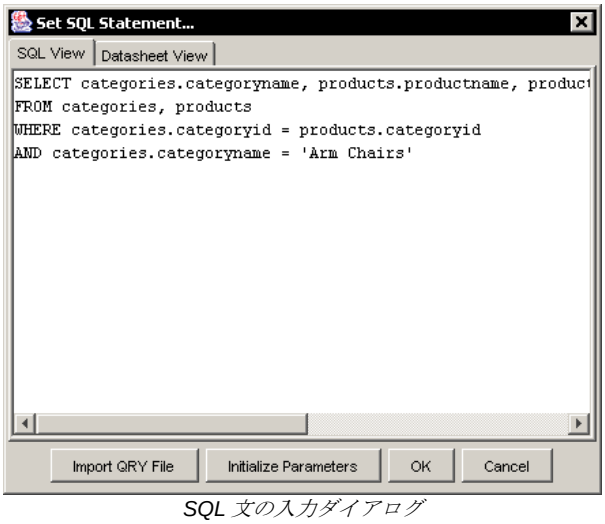
Prompt Name（プロンプト名）:パラメータダイアログでユーザに表示するプロンプトを指定できます。

'Previous Parameter'および'Next Parameter'ボタンをクリックすると、クエリーで定義されている各パラメータを初期化することができます。

パラメータ化されたクエリーを使用してチャートを設計することを選択するか、またはパラメータ化されたクエリーを使用するチャートを開く場合、チャートはデフォルト値でロード/スタートされます。チャートをプレビューすると、パラメータ値の入力を求めるプロンプトが表示されます。

5.2.2.3 SQL 文の入力

一般に、クエリーの作成にはクエリービルダーを使用することを推奨します。ただし、SQL 文を直接入力しなければならない場合もあります。例えば、クエリーが QRY ファイル内にすでに作成されている場合、クエリーをストアードプロシージャ/関数に組み込む場合、クエリーがクエリービルダーでサポートされていないコマンドを必要とする場合などがあります。このような場合、"Enter SQL statement"を選択して Set SQL Statement（SQL 文の設定）ウィンドウを開きます。次の図で示すように、このウィンドウのテキスト領域に SQL 文を直接入力するか、または既存の QRYF アイルをロードすることができます。



SQL 文の入力ダイアログ

結果セットをプレビューするには、Datasheet View（データシートビュー）タブをクリックします。

さらに、このインターフェイスを使用して Microsoft Excel のスプレッドシートからデータを引き出すこともできます。このためにはまず、ご使用の環境で、使用したい Excel ファイルを指示する ODBC DSN を設定します。次に、DSN を指示するデータベースをデータソースマネージャに追加します。この後、以下の SQL 文を入力することによって Excel ファイルに問い合わせることができます。

```
Select * from "SheetName$" (Replace SheetName with the name of your
Excel sheet.)
```

このクエリーを実行すると、指定したワークシート全体が返されます。

5.2.2.3.1 Oracle ストアードプロシージャの呼び出し

Oracle は、ストアードプロシージャおよび関数を処理する際、ほかのデータベースとは異なる方法を使用します。たとえば、MS SQL サーバでは、結果セットを返すために EXEC コマンドを使用します。一方、Oracle で結果セットを返すためには、REF CURSOR 型の OUT パラメータを使用しなければなりません。さらに、Oracle は、単一のクエリーからの複数の文を受け入れません。したがって、クエリーをストアード関数内に格納して、既存の Oracle ストアードプロシージャにアクセスするための特殊な構文を使用する必要があります。

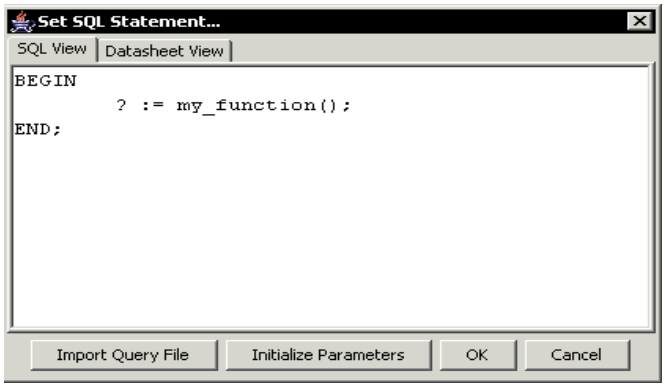
Oracle ストアードプロシージャにアクセスするにはまず、以下の PL/SQL 文を使用して弱く片付けされた REF CURSOR を定義します。

```
CREATE OR REPLACE PACKAGE types
AS
TYPE ref_cursor IS REF CURSOR;
END;
```

この ref_cursor 型は、クエリーの結果セットを格納し、OUT パラメータとして返すために使用されます。次のステップは、ストアードプロシージャを呼び出してクエリーを実行する関数の作成です。以下のスケルトンコードは、ref_cursor 型を使用して単純なクエリーを返します。

```
CREATE OR REPLACE FUNCTION my_function()
   RETURN types.ref_cursor
AS
   result_cursor types.ref_cursor;
BEGIN
   do_stored_procedure();
   OPEN result_cursor FOR
      SELECT * FROM Categories
   RETURN result_cursor;
END;
```

以上で Oracle ストアード関数が設定されたので、特殊な PL/SQL に似た構文を使用してチャートデザイナーから関数を簡単に呼び出すことができます。SQL 文の設定ウィンドウで以下の構文を入力して、Oracle ストアード関数を呼び出します。



単純な Oracle ストアード関数の呼び出し

'BEGIN ... END;'構文は、ユーザが Oracle ストアード関数にアクセスしようとしていることをシステムに警告します。さらに、'?'は、変数が OUT パラメータ用に予約されていることをチャートデザイナーに通知します。Oracle ストアードプロシージャを呼び出すための JDBC 構文は、以下のとおりです。

```
( call ? := my_function() )
```

ただし、EspressChart はこのフォーマットをサポートしていません。データベースビュータブをクリックして、結果をプレビューします。

EspressChart でストアードプロシージャを使用して有用なソリューションを開発する方法を示すために、より実用的な例を以下に示します。employee_table という名前のテーブルがあるとしましょう。これには、次に示すような組織の活動拠点を階層構造にしたデータが格納されています。

ID	名称	親	従業員数
1	All	NULL	0
2	America	1	0
3	Europe	1	0
4	New York	2	20
5	Santa Clara	2	30
6	Dallas	2	12
7	London	3	14
8	Paris	3	11

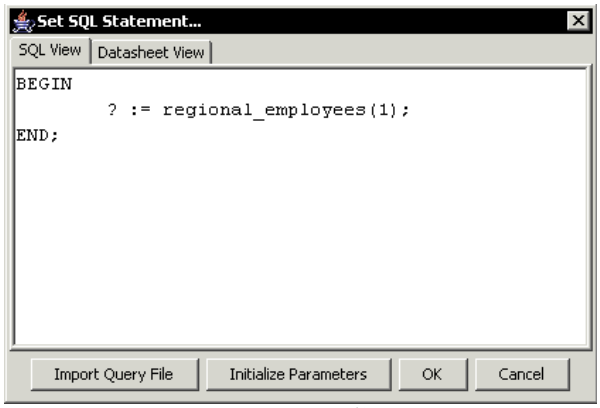
このテーブルは、企業の各種の活動拠点をツリー構造でリストします。リーフノード（例えば、New York、London など）には従業員数が格納され、各ノードには、すぐ上の親ノードに関する情報が入っています。ここで、特定の地域とその地域内の個々の支店に関する情報を表示するチャートを作成するとしましょう。例えば、ユーザが ID=2 を入力した場合、America の総従業員と支店を表示するチャートが必要です。Oracle の CONNECT BY および START WITH 句を使用すると、この 2 つの単純な Oracle ストアード関数を使用して問題を解決することができます。

```
CREATE OR REPLACE FUNCTION sum_employees(locID IN NUMBER)
       RETURN NUMBER
AS
       sum_emp NUMBER;
BEGIN
       SELECT sum(employee) INTO sum_emp
       FROM employee_table
       CONNECT BY PRIOR id = parent
       START WITH id = locID;

       RETURN sum_emp;
END;
CREATE OR REPLACE FUNCTION regional_employees (locID IN NUMBER)
       RETURN types.ref_cursor
AS
       result_cursor types.ref_cursor;
BEGIN
       OPEN result_cursor FOR
              SELECT id, name, sumEmployees(id) AS Employees
              FROM employee_table
              CONNECT BY prior id = parent
              START WITH id = locID;
```

```
        RETURN result_cursor;
END;
```

関数 sum_employees は、開始ノードをアーギュメントとして使用し、そのノードから派生するすべてのリーフノードの合計を検出します。例えば、Europe の従業員数は 25 名（London14 名、Paris11 名）であるため、sum_employees(3)は 25 を返します。2 番目の関数 regional_employees は、LocID から順にツリー構造を走査して、ID、名称および sum_employees 関数から得られた結果を元に結果セットを構築します。この後、REF CURSOR によって結果セットが返されます。

アーギュメントを必要とするストアード関数を呼び出すには、SQL 文の設定ウィンドウで以下の文を入力します。



regional_employees 関数の呼び出し

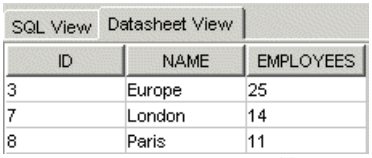
データシートビュータブをクリックして結果をプレビューします。

SQL View | Datasheet View

ID	NAME	EMPLOYEES
1	All	87
2	America	62
4	New York	20
5	Santa Clara	30
6	Dallas	12
3	Europe	25
7	London	14
8	Paris	11

regional_employees からの結果セット

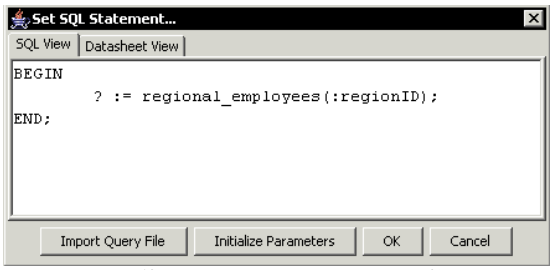
結果からわかるように、CONNECT BY 句は、ツリー構造を再帰的に走査して、America のノードをリストした後、Europe のノードをリストします。ユーザが Europe だけに関心があれば、パラメータとして 3 を入力します。その結果、以下のような結果セットが返されます。


SQL View | Datasheet View

ID	NAME	EMPLOYEES
3	Europe	25
7	London	14
8	Paris	11

Europe 内の regional_employees から得られた結果セット

パラメータ化されたチャートを作成するには、':param_name'構文を使用します。EspressChart 内の SQL パーサーは、パラメータとして使用するセミコロンと代入演算子(':=')として使用するセミコロンを区別することができます。以下に、パラメータの使用例を示します。



パラメータを使用した Oracle ストアード関数の呼び出し

IN パラメータを使用する場合、クエリーを実行する前にパラメータを初期化する必要があります。パラメータは既存のカラムにマッピングできないため、ストアードプロシージャを実行するには、正しいデフォルトのデータ型を設定することが特に重要です。パラメータの初期化の詳細については、セクション 5.2.2.2 を参照してください。この例を試行するには、<EspressChartInstall>\help\examples\data\ locationHierarchy の Example.sql に employee_table を作成する SQL コマンドと 2 つのストアード関数を格納します。

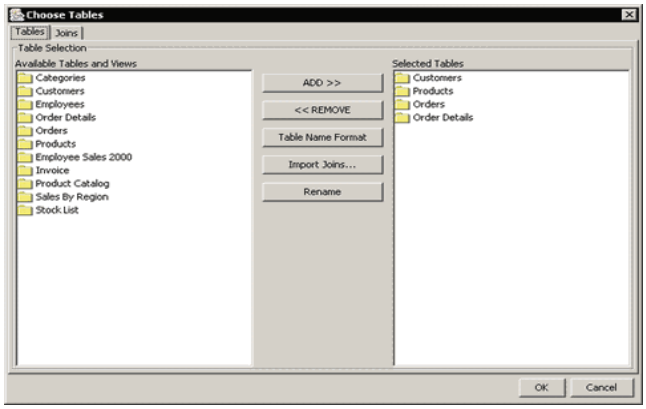
5.2.3 データビュー

EspressChart では、クエリインターフェース以外にデータベースのデータを取り込む方法としてデータビューを使用することができます。データビューはデータベースを簡素化した状態で表示し、フィールドを選択するだけの簡単な操作によってクエリをデザインすることを可能にします。クエリビルダを使用したり、データベースの構造に関する詳細な知識を持っている必要はありません。データビューを使用すると管理者はテーブル、ジョインおよびフィールドを事前に定義し、ローカルスキーマを作成してユーザに選択させることができます。

例えば、管理者が販売部門のためのデータビューを設定したとすると、適切なデータベーステーブルとフィールドが事前に選択されており、ビジネスユーザのロジックに沿った方法でグループ化することができます。例えば「売上」というグループは、適切な製品と売上の合計フィールドを含んでいます。エンドユーザはこのデータビュー

を選択し、関連するフィールドを選択し、データ範囲を指定した上でチャートのデザインを開始します。

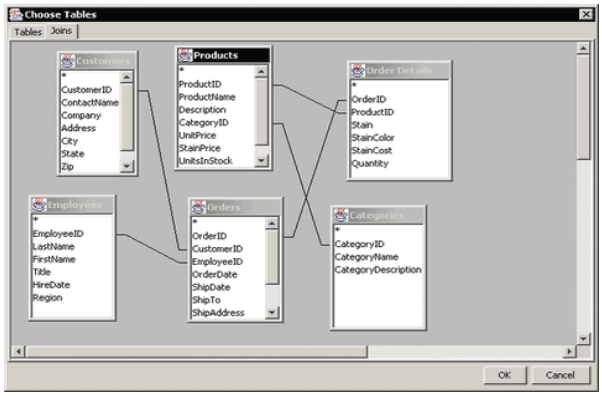
データビューを作成するにはデータソースマネージャウィンドウ内で"Data Views"ノードを選択し、'Add'をクリックします。新しいウィンドウが開きますので、データビューを使用したいデータベーステーブルを選択します。



データビューのテーブル選択ダイアログ

左側のウィンドウは使用可能なすべてのデータベーステーブルとビューを表示しています。テーブルを追加するにはこのウィンドウでテーブルを選択し、'ADD>>'ボタンをクリックします。デフォルトではデータビューは無効なテーブル名を使用します。この名前は 2 パートまたは 3 パートの有効な名前に変更することができます。名前を変更するには'Table Name Format'ボタンをクリックするか、'Rename'ボタンをクリックしてテーブルのエイリアスを指定します。他のデータビューから選択されたテーブルやジョインをインポートすることもでき、その場合は'Import Joins…'ボタンをクリックします。

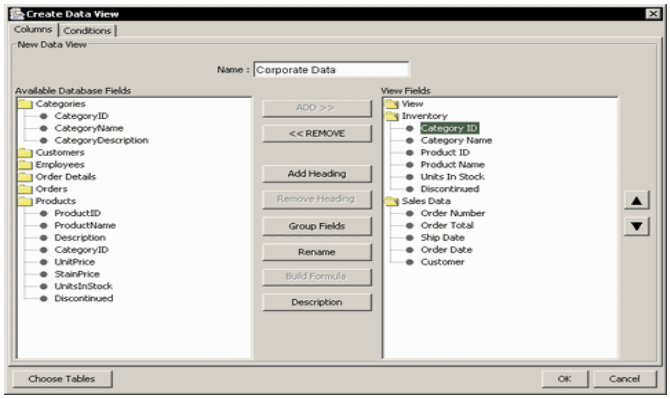
このウィンドウの'Joins'タブを選択すると、選択されたテーブル間のジョインを指定することができます。



データビューのジョインダイアログ

'Joins'タブに切り替えると、すべての選択されたテーブルと関連するフィールドが表示されます。デフォルトでは、テーブルはデータベース内での primary key と foreign key の関係に基づいてジョインされます。これらの自動的なジョインによってテーブル間に標準のジョインが作成されます。2 つのテーブルフィールド間に引かれた線がジョインを表します。ジョインを削除したりジョインプロパティを編集したい場合、線上で右クリックしてポップアップメニューから選択します。ジョインを追加するには、カラムフィールドをクリックして別のテーブル内のカラムにドラッグするとジョインが作成されます。データビューはクエリビルダと同じジョインプロパティを使用します。

テーブルの選択とジョインが終了したら'OK'をクリックします。新しいウィンドウが開き、データビューの構築が可能になります。



データビューの作成ダイアログ

左側のウィンドウには選択したテーブルと、それに関連するフィールドのリストが表示されます。各フォルダは 1 つのテーブルを表し、ダブルクリックによって開閉することができます。右側のウィンドウにはデータビューのために選択されたフィールドが表示されます。フィールドをデータビューに追加するには、左側のウィンドウでフ

ィールドを選択して'ADD >>'ボタンをクリックします。フィールドをデータビューから削除するには、フィールドを右側のウィンドウで選択して'<< REMOVE'ボタンをクリックします。'Build Formula'ボタンをクリックするとフォーミュラビルダが開き、演算されたカラムを作成することができます。エイリアスを定義するには右側のウィンドウで任意のビューフィールドを選択して'Rename'ボタンをクリックします。

ヘッディングを追加することによってデータビュー内のフィールドをグループ化することもできます。これによって独自に組織化された「バーチャルテーブル」構造を作成することができ、異なるデータベーステーブルからのデータをひとつのヘッディングの下にグループ化することができます。ヘッディングを作成するには'Add Heading'ボタンをクリックし、表示されたプロンプトでヘッディングの名前を指定します。新しいヘッディングが右側のウィンドウにフォルダとして表示されます。ヘッディング下にフィールドを追加するには、右側のウィンドウで追加したいフィールドを選択して'Group Fields'ボタンをクリックします。ドロップダウンメニューが表示されますので、フィールドの追加先となるヘッディングを選択します。

解説を加えるにはフィールドを選択して'Description'ボタンをクリックするか、右側のウィンドウ内でフィールドをダブルクリックします。そうすると新しいダイアログが立ち上がり、フィールドへの説明を指定することができます。エンドユーザはデータビューが実行されているとき、これらの説明をにアクセスすることができます。

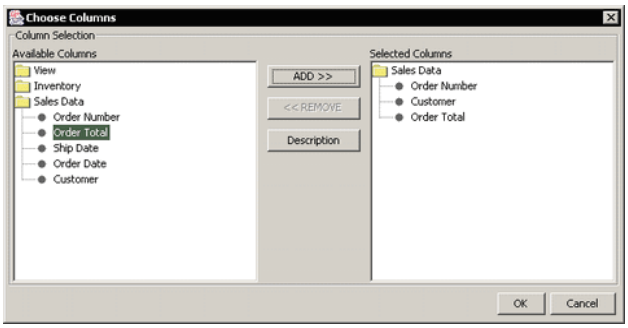
"Conditions" タブのページはフォーミュラービルダーウィンドウになっていて、エンドユーザのためにフィルタリングの基準を指定することができます。このウィンドウで追加したことはどれも SQL の Whrere クローズ（）に追加されます。フォーミュラビルダの使用方法の詳細についてはセクション 5.2.2.1.3 を参照してください。

.

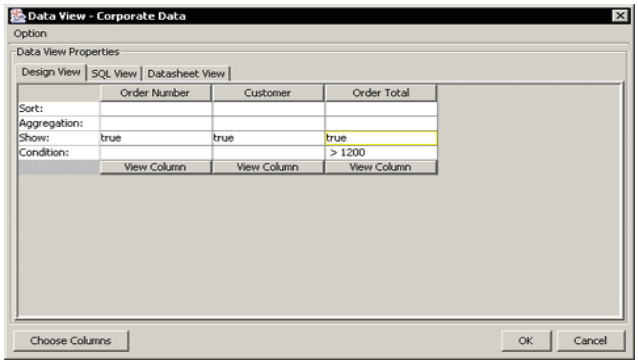
データビューの作成を終了して'OK'ボタンをクリックすると、データビューがデータソースマネージャに追加されます。これでユーザはこのビューを使用して任意の/アドホックなクエリを構築することができます。

データビューをデータソースとして使用してリポートまたはチャートをデザインするときは（データビューを選択して'Next'ボタンをクリック）、ビュー内のどのフィールドを使用するかを選択するためのウィンドウが開きます。このダイアログで、ビューカラムをベースにした計算フィールドを作成することができます。.

フィールドを選択したら'OK'をクリックします。新しいウィンドウが開いたらデータビューのためのソート、集計およびフィルタリングのコンディションを指定します。



データビューのフィールド選択ダイアログ



データビューのコンディションウィンドウ

データビュー内の各フィールドをダブルクリックすることにより、各フィールドごとのソートおよび集計のコンディション（条件）を指定することができます。ソートと集約はドロップダウンメニューから選択することができます。'Conditions'フィールドをダブルクリックすると新しいウィンドウが開き、'>'、'<'、'='、および'between'といった単純な選択基準を指定することができます。より詳細なフィルタリング基準を構築するには、'Conditions'フィールドを右クリックしてポップアップメニューから'Build'を選択するとコンディションを構築するためのフォーミュラビルダウィンドウが開きます。'View Column'ボタンをダブルクリックすることによって、カラム内の固有の値をすべて表示することもできます。

コンディションウィンドウの左上部にあるオプションメニューを使用すると、コンディションウィンドウの垂直／水平ビューの選択、データビュー内のパラメータの初期化、または選択セットとコンディション（条件）をクエリーとして保存することができます。

指定した選択セットとコンディション（条件）は名前フィールド指定した名前でデータビュークエリーとして保存されます。データビュークエリーはデータビューのノードの下に保存されます。データビューから作成されたチャートは更新や編集するのにデータビュークエリーを参照します。

5.2.3.1 データビューパラメータ

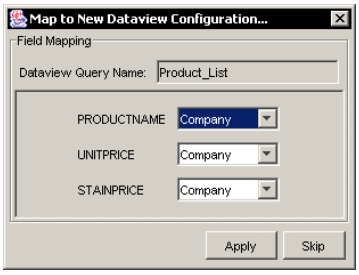
クエリビルダの場合と同様に、Data Views においてもユーザがクエリパラメータを指定することができます。データビューにパラメータを追加するにはデータソースマネージャでデータビューを選択して'Next'をクリックし、データビューを実行します。データビューのためのフィールドを選択し、コンディションウィンドウでカラムの"Condition"フィールドを右クリックし、ポップアップメニューから'Build'を選択します。フォーミュラビルダが起動しますので、クエリビルダと同様のやり方でパラメータを指定します。詳細については、第 5 章のセクション 4.2.2.2 を参照してください。

一度パラメータを入力したら、"Datasheet View"タブへ移動したとき、'Ok'をクリックして Chart Wizard を続けるとき、または選択をクエリとして保存するときにパラメータを初期化するよう求められます。Option メニューから'Initialize Parameters'を選択することによってパラメータを初期化することもできます。

5.2.3.2 データビュークエリーの更新

時々、データモデルとしてのデータビュー構造/構成に変更する必要があったり、変更要求などがあったりします。変更はフィールドの追加削除/リネームを含みます。データソースマネージャーで選択することによってデータビューからその関連クエリーまで変更を伝えることができます。まず Update（更新）メニューから'Data View Queries'（データビュークエリー）を選択します。

ビューに関連する全てのクエリーはスキャンされ、変更を行うユーザにフィールドやフィールド名の矛盾が表示されます。



更新クエリーフィールドダイアログ

各クエリーでは、既にデータビュー構成に合わなくなったフィールドを変更するようにプロンプトが出されます。 クエリーをそのままにしておきたい場合は'Skip'ボタンをクリックします。クエリーは実行を続けますが、古いデータビュー構成を参照することになります。'Apply'をクリックしてデータビュークエリーへの変更を保存します。

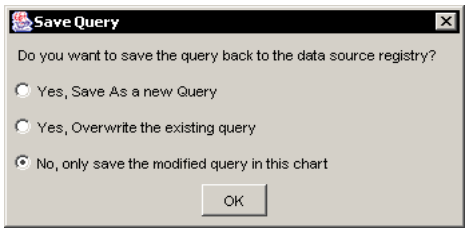
5.2.4 クエリーの編集

データベースのデータを使うチャートを作成する場合、クエリービルダーでクエリーをデザインすることによって、または SQL ステートメントを書くことによって、またはデータビューを実行することによって、のいずれかで作成するならば、データソ

ースマネージャに戻ることなく、チャートデザイナーから直接クエリーを修正することができます。

チャートクエリーを編集するにはチャートデザインのデータメニューから'Modify Query'を選びます。クエリービルダーでクエリーをデザインした場合、クエリー編集できるクエリービルダーインターフェースが再オープンします。SQL ステートメントを入れるなら、テキストボックスが開いて、SQL を編集することができます。データービューを使ったなら、データビューコンディション（条件）ウィンドウは再オープンして、フィルターの変更またはフィールドの追加ができます。

変更を指定したら、データレジストリの既存クエリーを修正する、新規クエリーをデータレジストリに保存する、またはテンプレート内のクエリーだけを修正するためのオプションが表示されます。

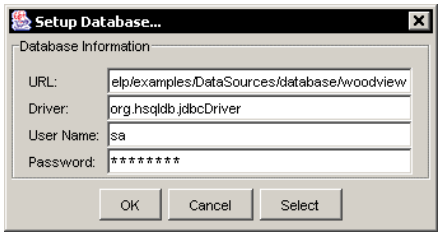


クエリーの保存オプション

保存オプションを指定すると、修正したクエリーがチャートに適用されます。クエリーに重要な変更を行った場合、チャートのデータマッピングを再度実行しなければならない場合があるので注意してください。データマッピングの詳細については第 6 章を参照してください。

5.2.5 データベース接続の編集

データベースデータを使うチャートの作成を選択した場合、チャートデザイナー中でテンプレートのデータベース接続情報を直接編集することができます。レポートの接続情報を編集するには、データメニューの'Modify Database' を選択します。これにより、データベースに接続するときに使われるレポートの URL,ドライバー、ユーザー名そしてパスワードを編集することができます。



データベース接続変更ダイアログ

さらに、手動でデータベース情報を入れるだけでなく、データレジストリからデータベース接続情報を取ってくることができます。それにはデータベース接続ダイアログの'Select' ボタンをクリックして、引き出したいデータベース接続の情報を持つ XML レジストリファイルをブラウズします。レジストリファイルを選択すると、レジストリに定義されているデータベースのリストが表示されます。

レジストリからのデータベースの選択

使用したいデータベースを選択し、'Ok'をクリックします。データベースの接続情報が自動的に接続ダイアログに適用されます。テンプレートの接続情報を設定したあと、'Ok'をクリックし、変更を適用させます。

クエリー機能をの編集と違い、テンプレートのデータベース接続への変更はテンプレートでのみ保存されます。データレジストリへは保存しません。

5.3 XML ファイルからのデータ

リレーショナルデータベースに加えて、EspressChart ではデータおよびクエリを XML ファイルから取り込むことができます。ほとんどすべてのフォーマットの XML データを使用できますが、XML データと共に DTD ファイルを指定する必要があります。XML データソースを設定するには、Data Source Manage の"XMLFiles"ノードを選択して'Add'をクリックします。ダイアログが表示されますので、新しい XML ソースのオプションを指定します。

XML データソースの設定ダイアログ

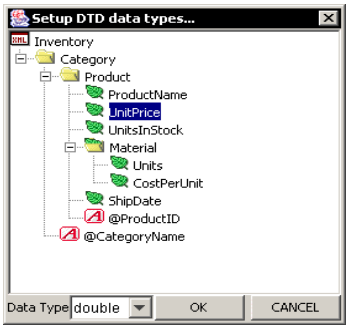
最初のオプションは XML データソースのディスプレイ名を指定します。2 番目のオプションは取り出したいデータを持つ XML ファイルのロケーションを指定します。ここではファイルロケーションとして特定の URL を追加することによって HTTP サーバから XML データを取ってくるデータソースを設定することもできます。3 番目のオプションはバリッド DTD または XML ファイルの XML スキーマファイルのロケーションを指定します。

'Quadbase Format' チェックボックスでは、XML ファイルが EspressChart からエクスポートされた XML 形式であるかどうかを指示することができます。例えば、チャートデータを XML 形式でエクスポートすることを選択すると、この形式を使ってそのデータを読み取ることができます。XML へのエクスポートに関しては第 9 章のセクション 9.2 を参照してください。この形式のファイルを使用する場合、DTD または XML スキーマを指定する必要はありません。

'Verify XML against DTD/XML Schema' チェックボックスは、与えられた XML ファイル/ソースが DTD または XML スキーマファイルで指定されたレイアウトに従っているかどうか検証させたいときはチェックを入れます。クエリーは DTD/XML スキーマファイルの構成をベースにデザインされているので、レイアウトに従っていない XML ソースは予想外の結果を作り出す可能性があります。XML が DTD もしくは XML スキーマに対して従っていない場合は注意が出されます。しかし、そのままソースの設定を続行することができます。

'Refresh Schema' チェックボックスは、レジストリの XML データソースの構造に対する変更を組み入れるために、スキーマまたは DTD 定義をリロードします。このオプションは、データソースが初めに作成されてから DTD またはスキーマ定義が変更されている場合のみ必要です。

XML ファイルと DTD/XML スキーマの設定を完了したら、XML データソース内のすべての選択可能なエレメントのデータ型を指定できる新しいダイアログが表示されます。このダイアログは、DTD ファイルまたは XML スキーマのどちらを使用しているかによって異なります。



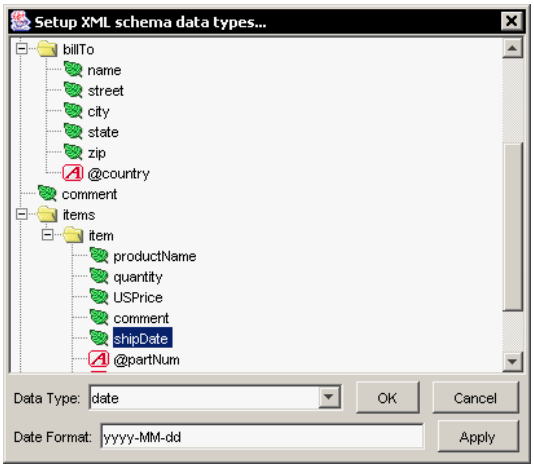
DTD のデータ型選択ダイアログ

DTD ファイルはエレメントの正確なデータタイプを指定しませんので、すべてのエレメントのデフォルトは文字列（string）になっています。エレメントのデータ型を変更するにはエレメントを選択し、ダイアログの下部にあるドロップダウンウィンドウからデータ型を選択します。XML ファイルのクエリーを行う時に正しい結果を得るためには、選択可能なすべてのエレメントについてデータ型を設定する必要があります。リーフノード、データを含むペアレントノード、および属性エレメントがこれに当たります。サポートされるデータ型は以下の通りです：
:

- **String** （文字列型）
- **Integer**（整数型）
- **Double** （倍精度浮動小数点型）
- **Boolean**（ブール型）
- **Date**（日付型） （データ型として Date を指定した場合は日付けのフォーマットも指定する必要があります。）

データ型の指定が終了して'OK'をクリックすると、データソースマネージャに XML ソースが追加されます。

If you're using an XML schema a different data types dialog will open.

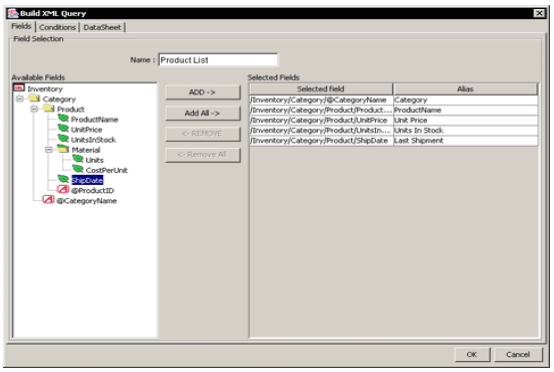


XML Schema Data Type Selection Dialog

Generally, the data types should already be defined in the XML schema file, but you can make any changes as necessary in this dialog. Once you have finished specifying the data types, click 'OK' and the XML source will be added to the Data Source Manager.

5.3.1 XML クエリ

XML データソースを設定したら、クエリを作成してノードを選択すること、フィルタリング条件を指定すること、およびツリー構造を EspressChart で使用する表形式に変換することができます。クエリを追加するには、XML ソースからノードを選択して'Add'ボタンをクリックします。XML クエリビルダインターフェースが開き、クエリの構築が可能になります。

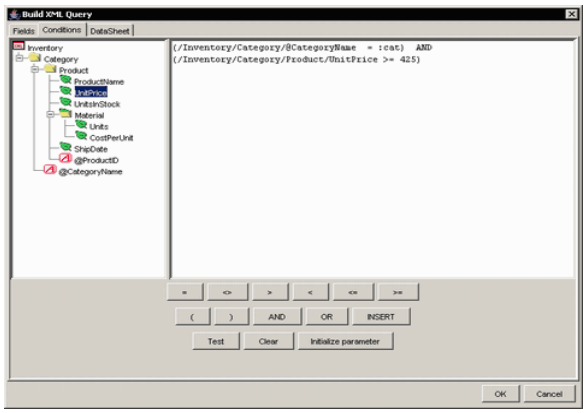


XML クエリフィールド選択タブ

XML クエリビルダの 1 番目のタブでは、レポートで使用したい XML ファイルからフィールド／ノードを選択することができます。ウィンドウの左側には DTD ファイルによるツリー構造が表示されます。エレメントを選択して'Add'ボタンをクリックするとエレメントをクエリに追加することができます。選択されたフィールドはウィンドウの右側に表示されます。フィールドのエイリアスを指定するには、カラムで "Alias"フィールドをダブルクリックして新しいカラムエイリアスを入力します。

＊注意− 選択された各エレメントはレポート内で 1 つのカラムになります。一対一の関係を決定することができないエレメントを伴う結果の場合、表構造はデータ内のすべての順列によって構築されます（SQL におけるクロスジョインと同様）。最適な結果を得るためには、明確な上下関係を持つフィールドをクエリに選択することをお勧めします。

XML クエリビルダの"Conditions"タブでは、選択したエレメントの基本的なフィルタリング基準を指定することができます。



XML クエリ Conditions タブ

XML ファイル内の選択可能なエレメントに指定できる条件は「等」、「不等」、「より大」、「より小」、「等またはより小」、「等またはより大」です。AND または OR 演算子を使用して複合条件を構築することも可能です。フィールドは XML ツリー下のダイレクトパスによって指定されます。現在のところサポートされているのはダイレクトパスのみで、より複雑な XPath 式を使用することはできません。フィールドを追加するにはウィンドウの左側でフィールドをダブルクリックするか、またはフィールドを選択して'Insert'ボタンをクリックします。

条件の入力を行ったら、'Test'ボタンをクリックしてシンタックスが正しいことを確認します。

"DataSheet"タブを使用するとクエリ結果をプレビューし、XML データがどのように表形式に変換されるかを確認することができます。クエリビルダ（セクション 5.2.2.16）で解説したのと同様の方法でリザルトセットをナビゲートすることができます。

フィールドを選択して適切な条件を指定したら'OK'ボタンをクリックします。クエリが Data Source Manager 内の XML ソース下に新しいノードとして追加され、レポートの作成に使用することが可能になります。

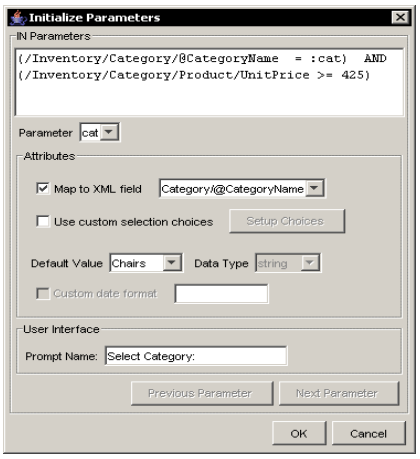
EspressChart インストレーションにサンプル XML ファイルとサンプル DTD ファイルが含まれています。ファイルはインストレーションの help/examples/DataSources/XML ディレクトリにあり、Inventory.xml および Inventory.dtd という名前になっています。Report Designer に XML データを流すためのサンプルのサーブレットも含まれています。サーブレットコードとインストラクションは help/examples/DataSources/XML/servlet ディレクトリに含まれています。

5.3.1.1 XML パラメータ

データベースクエリーと同様、XML クエリもパラメータを指定することができます。XML 条件でパラメータを表す場合も、クエリー条件と同一の":"構文を使用します。したがって、次の XML 条件は、"CategoryName"属性としてクエリーの動的フィルタを格納します。

```
/Inventory/Category/@CategoryName = :category
```

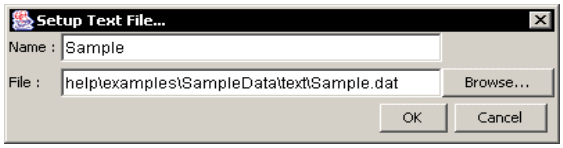
XML パラメータはクエリーパラメータと同一の方法で初期化されます。クエリーをプレビューするか、または閉じるときに、初期化ダイアログが表示されます。また、'Initialize Parameters' ボタンをクリックして初期化ダイアログを表示することもできます。異なるのは、XML ファイルでは、パラメータプロンプトをデータベースフィールドにマッピングする代わりにノードにマッピングできるという点だけです。



XML パラメータ初期化ダイアログ

5.4 テキストファイルからのデータ

EspressChart では標準的なテキストファイルからデータを取り込むこともできます。テキストファイルをデータソースとして取り込むには、Data Source Manager で"TXTFiles"ノードを選択して'Add'をクリックします。ダイアログが表示されますので、表示名と使用するテキストファイルの場所を指定します。



テキストデータソースの追加ダイアログ

情報を指定して'OK'をクリックすると、Data Source Manager ウィンドウの"TXTFiles"ノードの下にテキストソースが表示されます。

5.4.1 テキストファイルのフォーマット条件

EspressChart がテキストファイル内のデータを読み込めるようにするには、いくつかのフォーマット条件があります。一般的にはデータは以下のような形態になっている必要があります：

```
String,date,decimal
Name,day,Volume
"John","1997-10-3",32.3
"John","1997-4-3",20.2
"Mary","1997-9-3",10.2
"Mary","1997-10-04",18.6
```

上記のデータファイルは標準的なテキストファイルです。第 1 列はデータ型を指定し、第 2 列はフィールド名を指定します。第 3 列以降はレコードとなります。すべてのテ

キストファイルはこれら 3 つのパートで構成されている必要があります。例のデータファイルの場合は 4 つのレコードがあり、それぞれ 3 つのフィールドを持っています。フィールド間の区切り符号は","、";"、または" "（カンマ、セミコロン、またはスペース）が使用できます。各フィールドはクォーテーションまたはダブルクォーテーションで囲みます。

5.4.2 テキストファイルのデータ型とフォーマット

テキストデータファイルでは、データ型はキーワードを使用して指定されます。以下に認識可能なキーワードとそれらに対応する JDBC タイプと Java タイプのリストを示します。

データファイルキーワード (大文字小文字の区別はない)	JDBC タイプ	EspressChart の Java タイプ
boolean, logical, bit	BIT	boolean
Tinyint	TINYINT	byte
smallint, short	SMALLINT	short
int, integer	INTEGER	int
long, bigint	BIGINT	long
Float	FLOAT	double
Real	REAL	float
Double	DOUBLE	double
Numeric	NUMERIC	java.math.BigDecimal
Decimal	DECIMAL	java.math.BigDecimal
Date	DATE	java.sql.Date
Time	TIME	java.sql.Time
Timestamp	TIMESTAMP	java.sql.Timestamp
String	CHAR	String
Varchar	VARCHAR	String
Longvarchar	LONGVARCHAR	String

ある種のデータ型においては、テキストファイル内のデータは特定のフォーマットに沿っている必要があります。以下に特定のフォーマットを必要とするデータ型のリストを示します。

データ型	フォーマット	例
Date	yyyy-mm-dd or yyyy-mm	2001-06-12 or 200-06
Time	hh:mm:ss	12:17:34
Timestamp	yyyy-mm-dd hh:mm:ss	2001-06-12 12:17:34
boolean	true/false, t/f, 1/0 (case insensitive)	true

EspressChart インストレーション内にサンプルのテキストファイルが含まれています。このファイルは Sample.dat というファイル名で、ご使用のインストレーションの help/examples/DataSources/text ディレクトリにあります。

5.5 クラスファイルからのデータ

EspressChart ではより柔軟性の高いチャートデザインを可能にするため、Report Designer にデータを渡すインターフェースを供給することによってオブジェクトまたはアレイデータを使用したデザインを行うことができます。API を使用すると、JDBC リザルトセットで使用する java.sql.ResultSet に似た IResultSet オブジェクトを返す IDataSource を実装することができます。ユーザは独自の IResultSet の実装を供給するか、または EspressChart が供給する IResultSet を使用することができます。

この機能の詳細についてはセクション 11.5.3.4 を参照してください。

クラスファイルをデータソースとして追加するには、Data Source Manager で"ClassFiles"ノードを選択して'Add'をクリックします。ダイアログが開きますので、表示名と使用するクラスファイルの場所を指定します。

クラスファイルの追加ダイアログ

インストレーションの help/examples/DataSources/classes ディレクトリにサンプルのコードがあります。コンパイルされたコードは Report Designer へデータアレイを渡すクラスファイルを生成します。クラスファイルをデータソースとして使用するためには、ファイルをクラスパス内（Report Designer を使用している時は EspressManager バッチファイルの-classpath 変数）に持っている必要があります。

5.5.1 パラメータ化されたクラスファイル

EspressChart は追加の API インターフェースとして IParameterizedDataSource を提供しており、これによってクラスファイルデータソースのコンテクスト内にあるレポートパラメータを定義することができます。
クラスファイルのコンテクスト内で定義されたパラメータはクエリパラメータと同様の方法で作動します。パラメータ化されたクラスファイルデータソースの詳細についてはセクション 11.A.5 を参照してください。

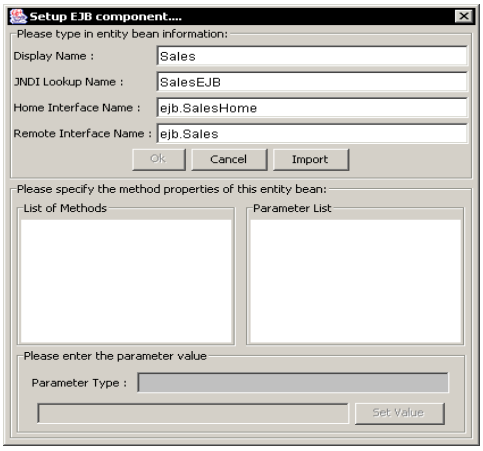
5.6 EJB からのデータ

Enterprise JavaBeans™ テクノロジーを使用すると、開発者は大規模な企業アプリケーションの開発をより簡単に行うことができます。EJB テクノロジーによって開発者はビジネスロジックの構築を行うことができ、アプリケーションサーバー（EJB コンテナ）にすべてのシステムレベルのサービスを管理させることができます。

J2EE™ 環境で作業している場合、持続的なデータインターフェースがエンティティビーンズより供給されます。内部のストレージメカニズムはリレーショナルデータベースである場合もありますが、アプリケーションデータモデルは EJB であり、冗長

なデータベース接続を行うチャート作成ツールを持つことは好ましくない可能性があります。このような場合、EspressChart ではデータのクエリをエンティティビーンより直接行うことができます。

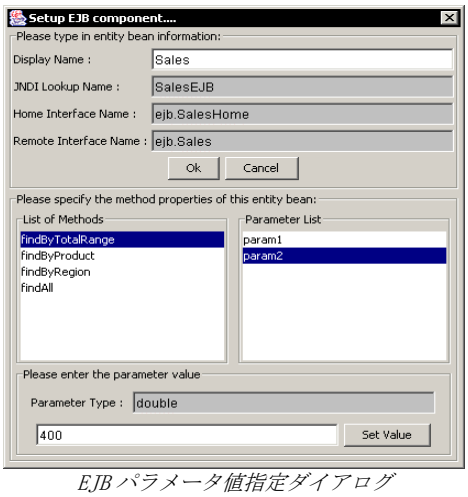
EJB をデータソースとして追加するには、最初に EJB がアプリケーションサーバに展開され、適切なスタブクラスを含むクライアント JAR ファイルがクラスパス（チャートデザイナーを使用している時は EspressManager バッチファイルの-classpath アーギュメント）に追加される必要があります。データソースマネージャで"EJBs"ノードを選択して'Add'をクリックするとダイアログが表示されますので、表示名とビーンの接続情報を指定します。



EJB の追加ダイアログ

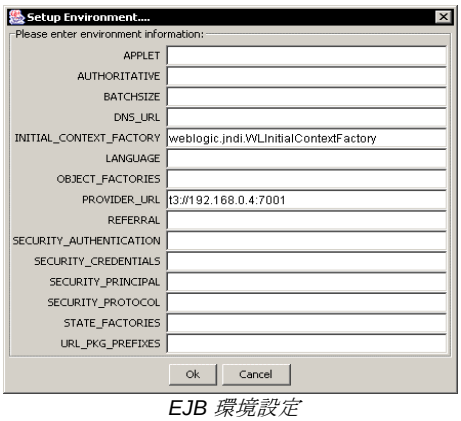
EJB データソースに接続するには、ビーンのための JNDI ルックアップ名を供給する必要があります（これはビーンを展開する時に指定されます）。また、ホームインターフェースの名前とリモートインターフェースを指定する必要もあります。すべての情報を指定して'Import'ボタンをクリックすると、ホームインターフェースの分析とすべてのファインダメソッドの取得が行われます。すべてのメソッドはダイアログの"List of Methods"セクションに表示されます。

同じダイアログを使用して、ファインダメソッド内に存在するパラメータに基づいて取得されたデータのフィルタリングを行うことができます。左側のダイアログでメソッドを選択すると、存在するすべてのパラメータが"Parameter List"セクションに表示されます。パラメータをクリックするとその値を設定することができます。



EJB パラメータ値指定ダイアログ

パラメータを選択すると、そのパラメータのデータ型がウィンドウの下部に表示されます。パラメータの値はその下に指定することができます。正しいデータ型の値が入力されていることを確認したら、"Set Value"ボタンをクリックしてパラメータ値を固定します。全てのパラメータ値を設定し終わったら'Environment' ボタンをクリックします。アプリケーションサーバの環境のプロパティを指定するダイアログが開きます。この情報は EspressManager が EJB に接続するのに必要です。



EJB 環境設定

ここでのフィールドは JNDIT 環境インターフェイスに利用可能な環境プロパティです。全てのフィールドに値を入れる必要はなく、その環境（アプリケーションサーバ）に必要な情報だけしてすればよいのです。環境変数を指定し終えたら、'OK'をクリックします。そうすると EJB 設定ウィンドウに戻ります。'OK'をクリックして EJB データソースの設定を終らせます。新しいノードが"EJBs" の下に表示されています。

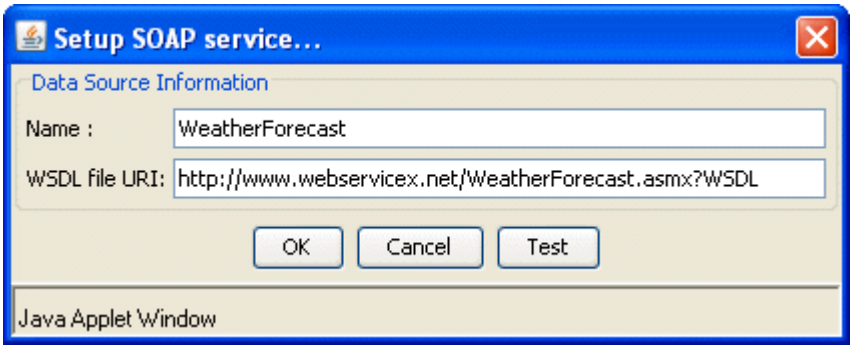
パラメータ値を修正するには EJB ソースを選択し、'Edit'ボタンをクリックします。

インストレーションの help/examples/DataSources/EJB ディレクトリにサンプルの EJB データソースがあります。このディレクトリには sales.ear と salesClient.jar ファイル、および Sales エンティティビーンのソースコードが含まれています。sales.ear ファイルは Java 2 Reference Implementation で展開されるようデザインされており、内部ストレージメカニズムとして Cloudscape データベースを使用します。SalesClient.java プログラムを使用して Cloudscape データベースを実装することができます。salesClient.jar ファイルは展開された Sales EJB に接続するためのスタブクラスを含んでおり、EspressManager クラスパスの中にある必要があります。

5.7 WSDL サポートを使用した SOAP からのデータの検索

ERES では、SOAP（サービス指向アーキテクチャプロトコル）を使用してデータを検索することもできます。WSDL を使用して SOAP データソースに接続するために、サービスの URL、SOAP のアクション、オペレーション名およびパラメータがわかっている必要はありません。わかっていなければならないのは、必要な情報がすべて入っている WSDL ファイルの場所だけです。

SOAP データソースを設定するには、データソースマネージャで"SOAPServices"ノードを選択した後、'Add'をクリックします。ダイアログが表示されて、新しい SOAP データソースのオプションを指定するように要求してきます。

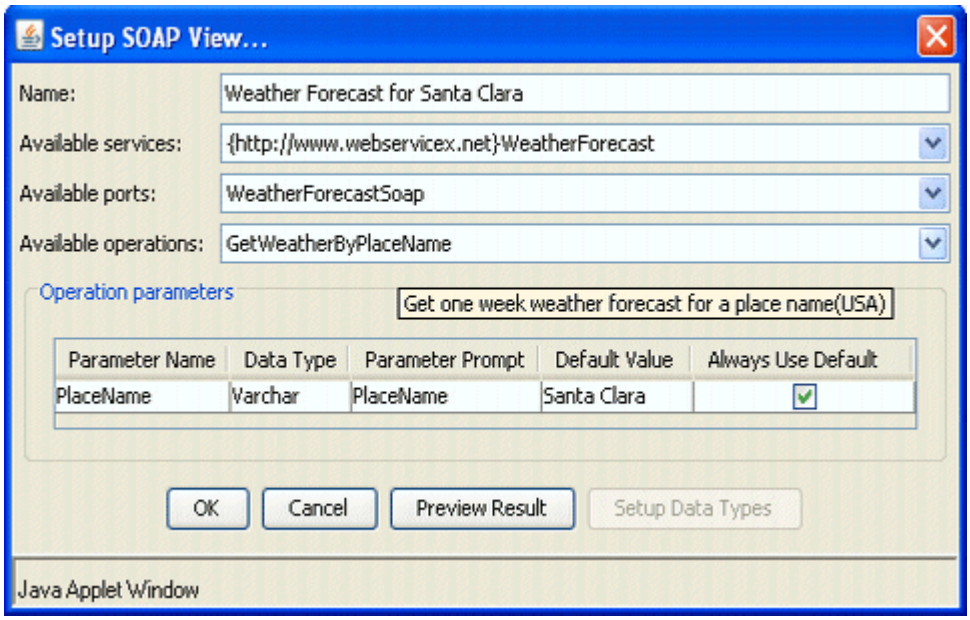


SOAP データソースの設定

最初のオプションでは、データソースの表示名を指定することができます。2 番目のオプションでは、WSDL ファイルの場所を指定することができます。場所としては、サーバ上の絶対パス、または ERES のインストールディレクトリに対する相対パスか、あるいは URL のいずれかを指定することができます。接続情報を指定すると、'Test'ボタンをクリックして WSDL に対する接続をテストすることができます。この

テストでは、指定した URI から WSDL ファイルを検索して、サポートされているすべての SOAP 操作について、そのファイルをチェックした後、何らかの問題があれば、それを報告します。'OK'をクリックすると、新しい SOAP データソースのノードがデータレジストリに追加されます。

新しい SOAP ビューを追加するには、既存の SOAP データソースを選択した後、'Add'ボタンをクリックします。ダイアログが表示されて、SOAP クエリを作成するために必要なすべてのパラメータを要求してきます。

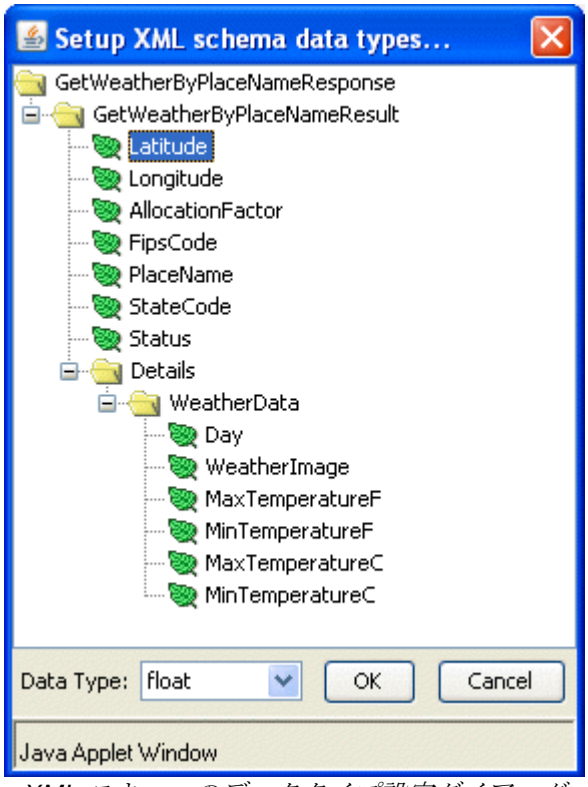


SOAP ビュー設定ダイアログ

最初のオプションでは、データソースマネージャ内の表示名を指定することができます。次に、ダイアログ内には、3 つのドロップダウンメニューがあります。最初のドロップダウンメニューには、WSDL ファイル内に記述されているすべての SOAP サービスが入っています。サービスを指定すると、2 番目のドロップダウンリストに、このサービスのすべてのポートが自動的に入力されます。ポートを選択すると、最後のドロップダウンリストに、このポートのすべてのオペレーションが自動的に入力されます。いずれかのドロップダウンリスト上にマウスを移動すると、ヒントと（WSDL ファイルにマニュアルが提供されている場合）サービス/ポート/オペレーションに関するマニュアルが表示されます。サービス、ポートおよびオペレーションを指定すると、オペレーションのすべてのパラメータが読み込まれます。複数のパラメータがある場合、それらは表として表示されます。この表の最初の 2 つの列は、編集することができません。これらは、WSDL ファイルから読み取られます。次の 2 つの列は編集可能で、パラメータのプロンプトとデフォルト値を指定することができます。

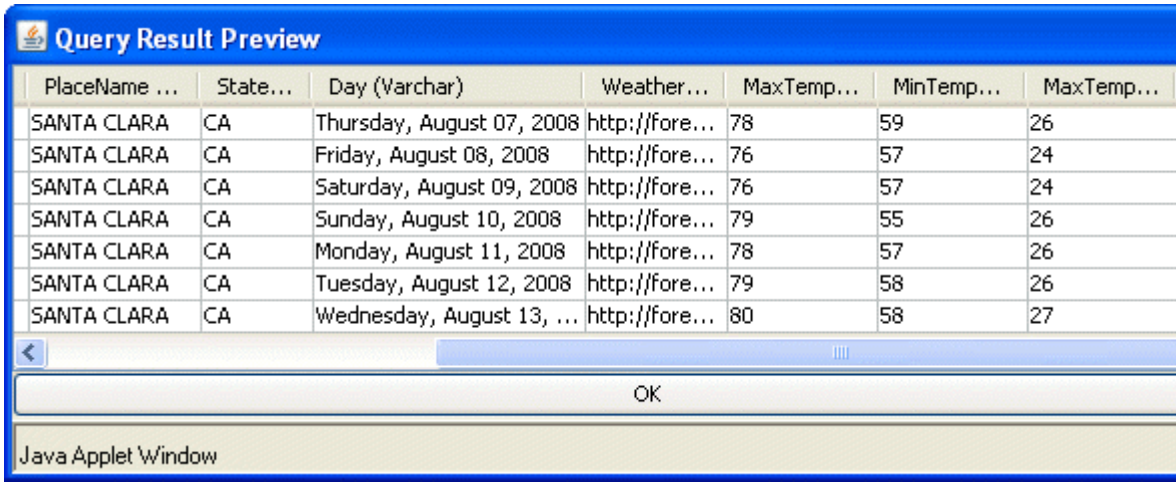
最後の列には、チェックボックスが表示されます。このチェックボックスでは、デフォルトのパラメータ値を常に使用するかどうかを選択することができます。これは、このパラメータの値が固定され、値の入力要求が表示されないことを意味します。このチェックボックスでチェックしないパラメータはすべて、レポート/チャートパラメータとして使用されます。

'Setup Data Type'ボタンは、データソースマネージャから SOAP ビューを編集する際にのみ使用することができます。このボタンを使用すると、必要に応じてデータタイプを調整することができます。SOAP 応答からの結果を検証するには、'Preview Result'ボタンをクリックします。その場合、すべてのデフォルト値がテストされ、正しいデータタイプであるかどうかが確認されます。一致しない場合、データタイプを調整するように要求してきます。その後、データタイプの設定ダイアログ（XML データソースの場合と同一）が表示されます。データソースがパラメータ化されている場合、データタイプを指定する前にパラメータプロンプトダイアログが表示されます。XML スキーマを正しく生成するには、既存のパラメータ値を指定しなければならないことに注意してください。



XML スキーマのデータタイプ設定ダイアログ

このダイアログから、データタイプを設定することができます。動作は、DTD スキーマを指定した XML データソースの場合とまったく同じです（第 5 章のセクション 5.4 を参照）。データタイプの指定が完了したら、'OK'をクリックします。結果のプレビューを表示したダイアログが表示されます。



クエリー結果プレビューダイアログ

このダイアログで'OK'をクリックすると、SOAP ビュー設定ダイアログに戻ります。必要な情報をすべて指定し終わったら、このダイアログで'OK'ボタンをクリックします。データソースマネージャ内の SOAP データソースの下に新しいノードが追加され、レポートまたはチャートの作成に使用できるようになります。

www.webservicex.netからのウェブサービスの例をいくつか示します。試してみてください。

USA Weather Forecast: このウェブサービスは、米国内の有効な郵便番号または地名の天気予報を取得します。使用する WSDL ファイルの場所は、
http://www.webservicex.net/WeatherForecast.asmx?wsdl
です。

US Address Verification: このサービスは、米国の住所を確認するだけです。WSDL ファイルの場所は、
`http://www.webservicex.net/usaddressverification.asmx?wsdl` です。

Currency Convertor: このウェブサービスは、ある通貨から別の通貨への為替レートを入手します。WSDL ファイルの場所は、

`http://www.webservicex.net/CurrencyConvertor.asmx?wsd`
です。

5.8 Salesforce からのデータ

Salesforce データソースは、強力なチャートおよびダッシュボードソリューションを使用して ERES 内の Salesforce データを表示したい既存の Salesforce ユーザ向けに設計されています。Salesforce サーバへの接続は、Salesforce Partner WSDL（バージョン 13.0）を使用し、SOAP を経由して確立されます。ユーザは、SOQL（Salesforce オブジェクトクエリ言語）クエリによって Salesforce サーバと通信します。ユーザがこのデータソースを操作するには、ユーザ名とパスワードを指定した有効な Salesforce アカウントが必要であることに注意してください。さらに、ERES Salesforce データソースを使用するユーザは、信頼できるネットワークから Salesforce アカウントにアクセスできなければなりません。信頼できる IP リストに IP アドレスを追加するには、以下の説明に従ってコンピュータをアクティブにする必要があります。

SOQL クエリ、および信頼できるネットワークから Salesforce ユーザのアカウントをアクティブにする方法の詳細については、以下の Salesforce サイトを参照してください。

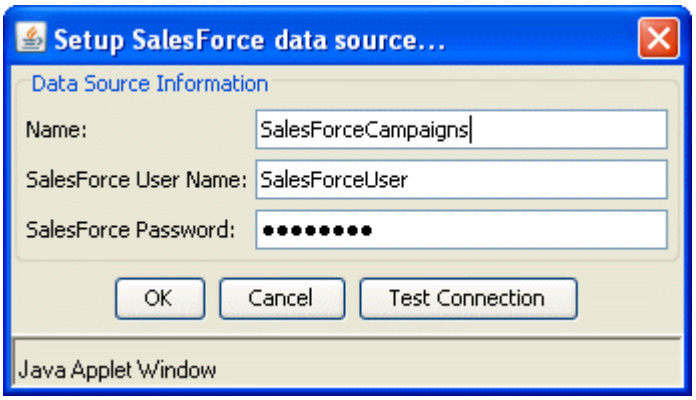
SOQL クエリについて

> http://www.salesforce.com/us/developer/docs/api/Content/sforce_api_calls_soql.htm

Salesforce ユーザのアカウントをアクティブにする方法について

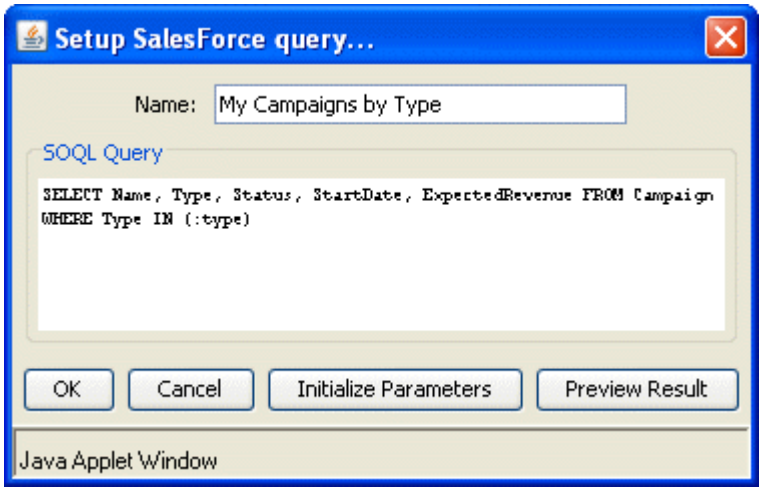
> http://na5.salesforce.com/help/doc/en/security_networkaccess.htm

Salesforce データソースを設定するには、データソースマネージャ内で”SalesForce”ノードを選択して、'Add'ボタンをクリックします。ダイアログが表示されて、データソースの表示名、ユーザ名および Salesforce アカウントのパスワードを入力するように要求してきます。接続情報を指定したら、'Test Connection'ボタンをクリックして Salesforce アカウントに対する接続をテストすることができます。これは、指定した情報を使用して接続をテストした後、問題があれば、それを報告します。



SalesForce データソース設定ダイアログ

Salesforce データソースを追加すると、データソースマネージャウィンドウに新しいノードが表示されます。新しい Salesforce クエリを追加するには、'ADD'ボタンをクリックします。新しいダイアログが表示されて、クエリ名と SOQL クエリを指定するように要求してきます。



SalesForce クエリ設定ダイアログ

現行の ERES バージョンでサポートされているのは、子-親関係のクエリだけであることに注意してください。このため、（ネスト SOQL クエリを使用する）親-子クエリを使用することはできません。Salesforce 関係クエリおよびその構文の詳細については、以下の Salesforce サイトを参照してください。

http://www.salesforce.com/us/developer/docs/api/Content/sforce_api_calls_soql_relationships.htm

さらに、このダイアログでは、クエリに単一値または複数値パラメータを含む場合、クエリパラメータを初期化することができます。パラメータは、SOQL 文内で":"文字によって指定されます。通常、パラメータは、SOQL Select 文の WHERE 句内にあります。たとえば、以下の SOQL 文は、"CampaignName"という名前の単一値パラメータを指定します。

```
Select Name, Type, Status, ActualCost From
Campaign Where Name = :CampaignName
```

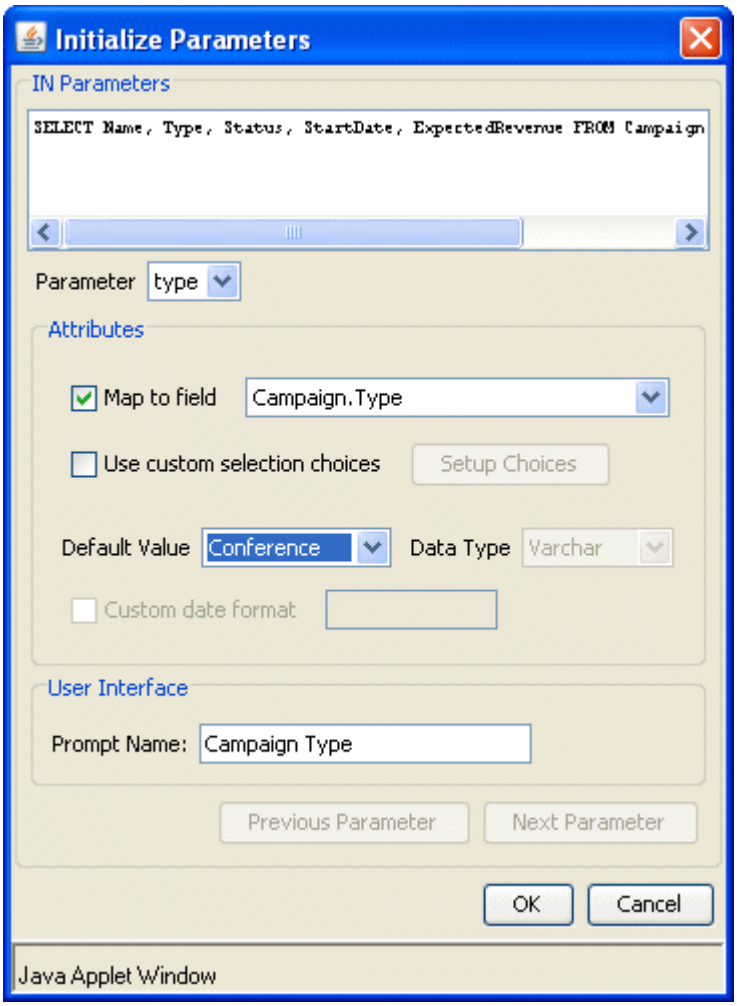
この後、実行時のキャンペーン名を入力して、そのキャンペーンのデータだけを検索することができます。

もう 1 つの SOQL 文の例は、単一値ではなく値のアレイを入力して使用する複数値パラメータの使い方を示しています。

```
Select Name, Description, Type, LeadSource,
Probability From Opportunity Where Type IN
(:OpportunityType) And LeadSource IN
(:OppLeadSource)
```

この文は、"OpportunityType"および"OppLeadSource"という名前の 2 つの複数値パラメータを指定します。この後、実行時の機会タイプとリードソースを指定することができ、指定されたパラメータ値だけに従ってデータが検索されます。

SOQL クエリパラメータを初期化するには、'Initialize Parameters'ボタンをクリックします。パラメータ初期化ダイアログが表示され、パラメータのマッピングを指定することができます。



パラメータ初期化ダイアログ

このダイアログから、以下のオプションを指定することができます。

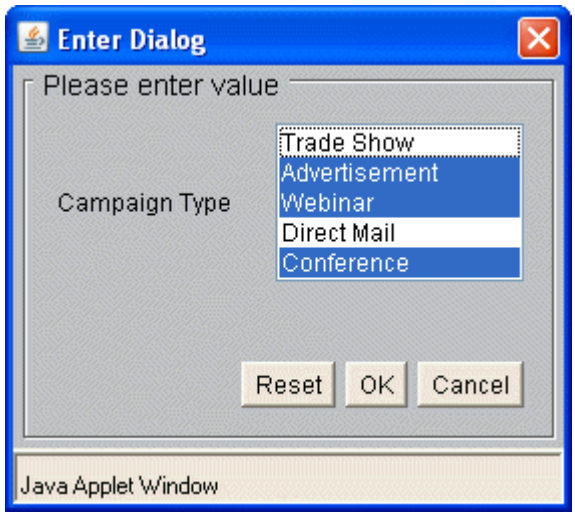
Map to field（フィールドへのマップ）：値をパラメータの入力に使用する Salesforce データソースのフィールドを指定することができます。このオプションを選択すると、レポート/チャートをプレビューするか、または実行するときに表示されるパラメータプロンプトが変更されます。パラメータを Salesforce フィールドにマップすると、別個のパラメータ値のドロップダウンリストが表示され、そこから値を選択します。マップしない場合、個々のパラメータ値を入力しなければなりません。

Use custom selection choices（カスタムセレクションチョイスの使用）：個別のカラム値をすべて表示するドロップダウンメニューを使用しないで、エンドユーザも選択できるパラメータ値のカスタムリストを構築することができます。リストを設定す

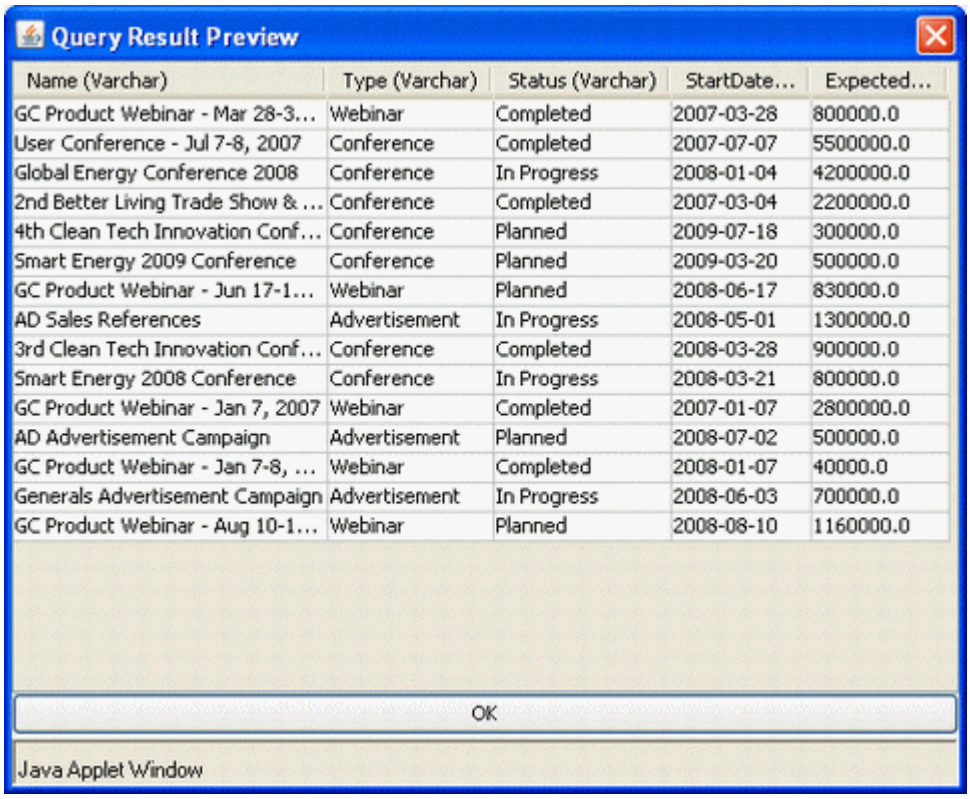
るには、このオプションを選択して、'Setup Choices'ボタンをクリックします。こうすることによって、新しいダイアログが起動され、選択肢リストを作成することができます。

残りのオプションは、基本的に、データベースクエリパラメータと同じです。データベースクエリパラメータの初期化の詳細については、第 5 章のセクション 5.2.2.2.2 を参照してください。すべての利用可能なパラメータに対するマッピングを指定したら、'OK'ボタンをクリックします。Salesforce クエリ設定ダイアログに戻ります。

Salesforce クエリ設定ダイアログでは、クエリからの出力を確認するために、'Preview Result'ボタンを使用してクエリ結果をプレビューすることができます。クエリをパラメータ化している場合、パラメータプロンプトダイアログが表示され、パラメータ値を指定するように要求してきます。パラメータ値を指定したら'OK'ボタンをクリックします。クエリ結果プレビューダイアログが表示されます。



パラメータプロンプト



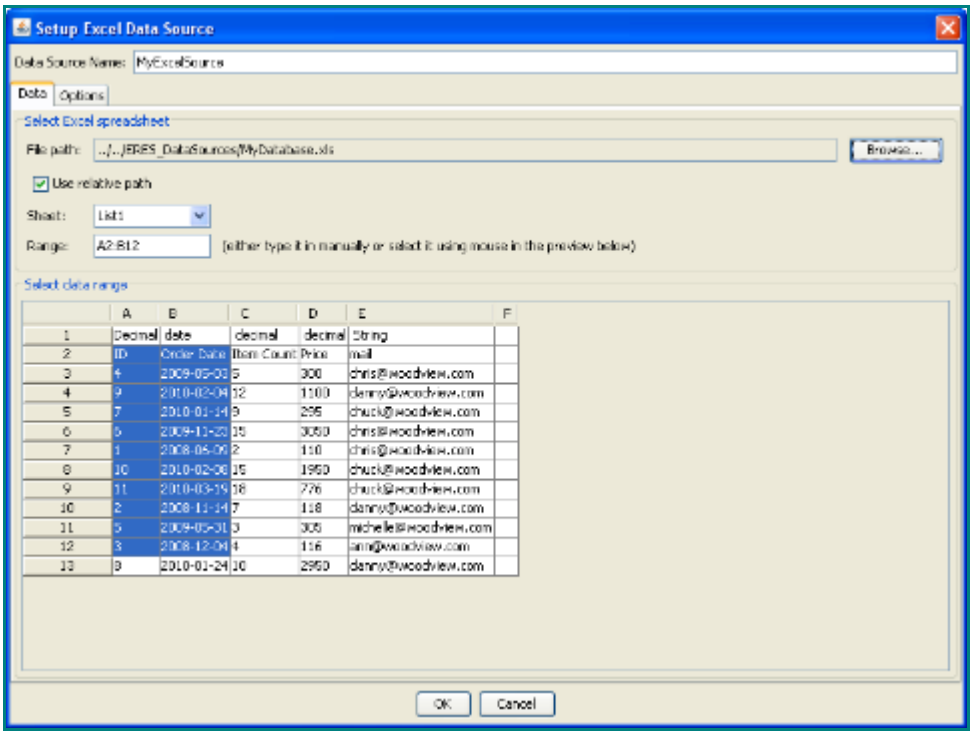
クエリ結果プレビューダイアログ

この第ログでは、クエリ出力を確認することができます。'OK'ボタンをクリックすると、SalesForce クエリダイアログに戻ります。

クエリを指定したら、'OK'ボタンをクリックします。クエリは、データソースマネージャ内の Salesforce データソースの下に新しいノードとして追加され、レポートまたはチャートに使用できるようになります。

5.9 Excel ファイルからのデータ

EspressChart は、Excel ファイルからデータを取り込んで、チャートをデザインすることができます。Excel ファイルをデータソースとして追加するには、データソースマネージャーから ExcelFiles ノードを選択し、「Add」をクリックします。ダイアログがオープンされ、データソース名とデータをインポートするからの Excel ファイルを指定します。



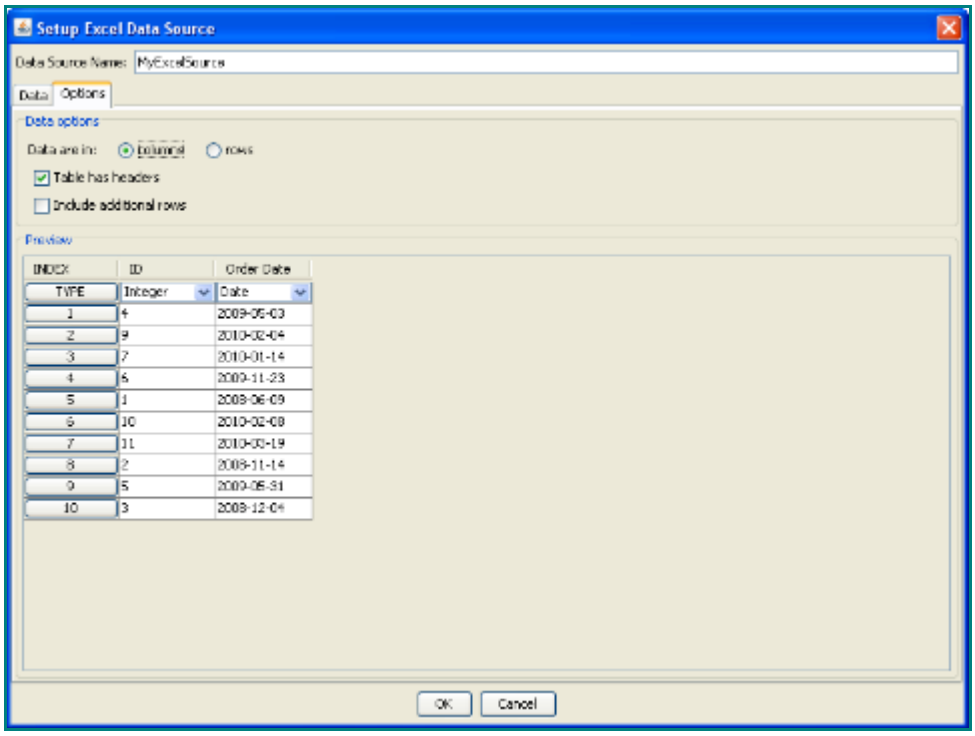
Excel データソースダイアログのセットアップ ー　データ

Excel ファイルを選択した際、取り込んだデータをダイアログにプレビューされます。チェックボックス項目の “Use Relative Path”（相対パス）をチェックしたら、選択した Excel ファイルのファイルパスは EspressChart インストールディレクトリから道筋することになります。そうしなければ、フールパス(絶対パス)を使用します。Excel ファイルは EspressChart インストールディレクトリと異なるディスクドライブに格納されている場合、このオプションは無効になります。マウス（のクリック＆ドラッグ）、それとも、Range ボックスでの範囲の指定で、セットアップしているデータソースに該当するシート（ファイル内に複数の場合）とセルを選択することができます （例えば、Excel シートのコラム A と B の２～１２行のデータをデータソースとして使用する場合は、Range ボックスに `A2:B12` を入力します。この操作は、MS Excel でも提供しています。）その他のデータ選択方法としては、行ヘッダーと列ヘッダーのクリックで行ないます。**Ctrl** か **Shift** を押しながら、複数の行と列を選択します。もしくは、左上隅のクリックで全部のセルを選択します。この操作も、MS Excel のと同じです。

EspressChart は両方の`*.xls` ファイルと `*.xlsx` ファイルが操作できることを留意してください。`*.xlsx` ファイルは、Open XML に基づき、Microsoft Excel 2007/2010 で使用されます。

EspressChart は、Excel の基本フォーミュラも操作できます。もし、入力したフォーミュラが操作できなかった場合は、エラーメッセージが表示されます。

シートの最後にある空白の行（データは列での場合)か列（データは行での場合)）は、選択された場合も、データソースから自動的に削除されます。
データの構造を正しく取得するため、Option タブをクリックし、データは列か行の形で並んでいるかどうか指定します。データ選択にテーブルのヘッダーも含まれているか指定することができます。追加行（データは列での場合)、または、追加列（データは行での場合）のオプションがあります。これに関して、データソースマネージャーでのデータソースを変更をせずに、データソースを作成した後、Excel ファイルに追加されるデータ行と列は自動的に含むことができます。
オプションタブでのダイアログの下部により、現在のデータ構造に従うデータソースを表示します。必要の場合、データソースの各コラムのデータ種類を変更することができます。しかしながら、データ種類は自動的に反映されるため、データ種類の変更は必要ではない場合が多いです。



Excel データソースダイアログのセットアップ ー　オプション

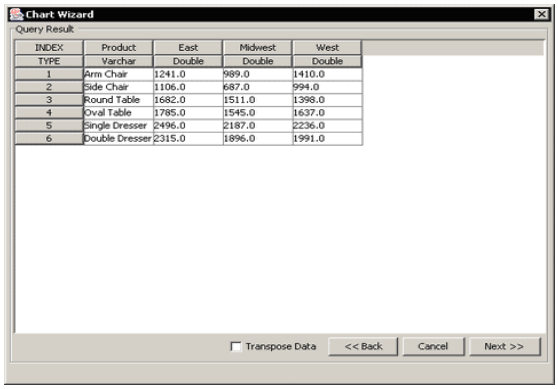
設定完了の際、「OK」をクリックしデータソースを保存します。

5.10 データのトランスポート

使用するデータソースを選択して'Next'をクリックすると、次の画面にデータソースからの最初の 20 のレコードが表示されます（データビューでは最初にフィールドの

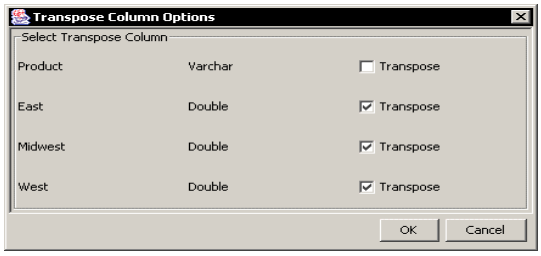
選択とコンディションの設定を行う必要があるため、これにはあてはまりません）。この画面で'Show All Records'ボックスをチェックすると、選択されたソースによって返されたすべてのレコードを表示することができます。

データをスプレッドシート（ピボットテーブル）形式で取り込む場合、データソースからのカラムヘッダをカテゴリまたはデータシリーズとして使用することがしばしばあります。例えばスプレッドシートクエリがデータソースとして選択された場合、テーブルは以下のようになります。



データテーブルダイアログ

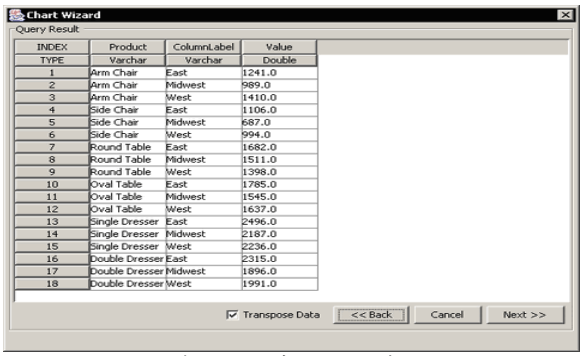
チャート内で地域別のカラムヘッダ（East, Midwest, & West）を作るためには、初めにデータを転置する必要があります。データを転置するにはデータテーブルウィンドウの底部にあるチェックボックスをクリックします。新しいウィンドウが開きますので、転置したいカラムを選択します。



転置の選択ダイアログ

転置したいカラムをチェックして'Ok'をクリックすると、データテーブルに戻ってデータ転置が行われたことを確認できます。選択したカラムはデータテーブルから削除され、転置されたカラムはテーブルの最後に新しいカラムとして配置されます。

＊注意− 選択したカラムは同じデータ型である必要があります。



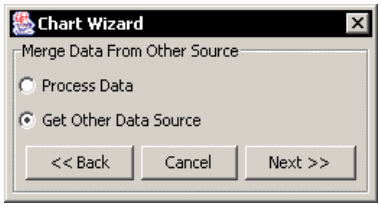
INDEX	Product	ColumnLabel	Value
TYPE	Varchar	Varchar	Double
1	Arm Chair	East	1241.0
2	Arm Chair	Midwest	989.0
3	Arm Chair	West	1410.0
4	Side Chair	East	1106.0
5	Side Chair	Midwest	687.0
6	Side Chair	West	994.0
7	Round Table	East	1682.0
8	Round Table	Midwest	1511.0
9	Round Table	West	1398.0
10	Oval Table	East	1785.0
11	Oval Table	Midwest	1545.0
12	Oval Table	West	1637.0
13	Single Dresser	East	2496.0
14	Single Dresser	Midwest	2187.0
15	Single Dresser	West	2236.0
16	Double Dresser	East	2315.0
17	Double Dresser	Midwest	1896.0
18	Double Dresser	West	1991.0

転置後のデータテーブル

5.11 複数データソースの使用

使用したいデータソースを選択して'Next'をクリックすると、次の画面にデータソースの最初の 20 レコードが表示されます（データビューを除きます。データビューでは、最初にフィールドを選択して条件を設定しなければなりません）。 この画面では、'Show All Records'ボックスをチェックすることによって、選択したソースから返される全てのレコードを表示することができます。

EspressChart では複数のデータソースからチャートを構築することができます。最初のソースを選択してデータテーブルウィンドウから次のソースをクリックすると、現在のデータを処理するか、または他のデータソースを選択するかを尋ねるダイアログが表示されます。

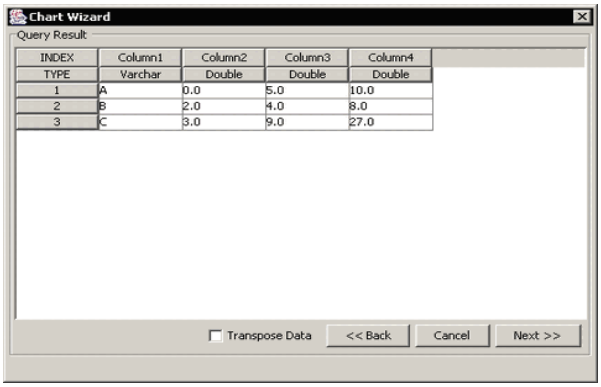


追加のデータソースダイアログ

'Process Data'を選択して'Next'クリックすると、Chart Wizard の次のステップへ進みます。'Get Other Data Source'を選択するとデータソースマネージャに戻り、チャートに使用する他のデータソースを選択することができます。この操作を繰り返すことによっていくつでもソースを選択することができます。

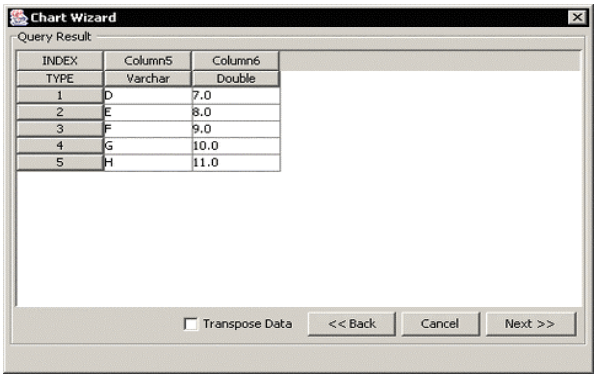
複数のデータソースはデータテーブル内で隣り合って組み合わされた状態で表示されます。このため 2 番目またはそれ以降のデータソースからなるカラムは最初のデータソースからなるカラムの右側に配置されます。あるデータソースからなるカラムが他のカラムよりも多くの列を持つ場合、その列には null 値が配置されます。

例えば 2 つのデータソースを使用してチャートを作成するとします。1 番目のデータソースから生成されるデータテーブルは以下のようになります：



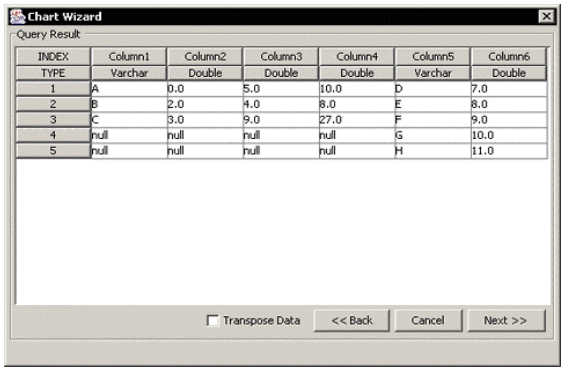
1 番目のデータソースからなるデータ

2 番目のデータソースからなるデータは以下のようになります：



2 番目のデータソースからなるデータ

2 つのデータソースを組み合わせると以下のようなデータテーブルが得られます：



複合ソースからなるデータ

このように 2 つのデータソースは横に並べて表示されます。2 番目のソースの方が 1 番目のソースよりもデータ列が多いため、null データの列が追加されています。

パラメータ化されたデータソースを使用して複数のデータソースを作成できないことに注意してください。

5.12 データソースの変更

チャートデザインではどの時点でもテンプレートのデータソースを変更する選択ができます。チャートテンプレートはデータソース情報を保存しているので、チャートデザイナーでテンプレートのデータソースを変更するオプションを使用する必要があります。データソース更新機能を使わない限り、単にレジストリでデータソースを変更しても作用しません。 (これに関する詳細は 5 章のセクション 5.11 を参照してください。)。 テンプレートのデータソースを変更するにはデータメニューから'Modify Data Source'を選択するか、ツールバーの 'Modify Data Source'ボタンをクリックします。データソースマネージャが立ち上がり、新しいデータソース選択または既存のデータソースの編集ができます。

データソースを変更するとき、データをチャートへ再マッピングする必要があるかもしれません。レイアウトとデータマッピングに関する詳細は 6 章を参照してください。

5.13 データソースの更新

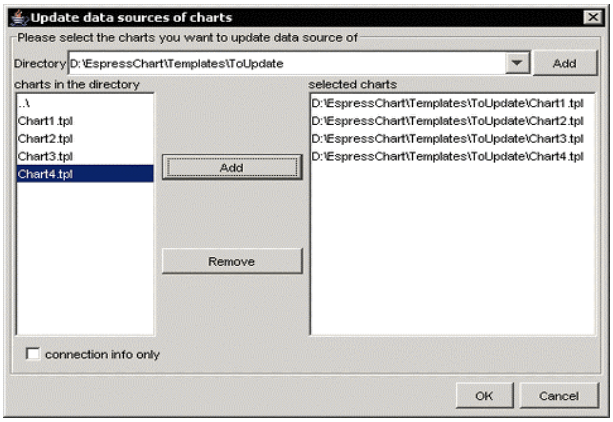
テンプレートのグループをどこかへ移動したいとか、EspressChart のインストレーションをある場所から他の場所へ移動したいなど、いろいろな状況が生まれてきます。例えば、アプリケーションが開発からテストへ、製品へと環境が移るかもしれません。各環境ではデータソースのロケーションと接続情報が変わるかもしれません。このような場合、前のセクションで説明したようなひとつひとつのテンプレートの更新では、かなりの数のテンプレートの接続情報を変更していてはたいへんです。EspressChart は、データレジストリの情報を基にテンプレートのグループを素早く更新することができるのです。

チャートテンプレートはデータソース情報の内部コピーを維持しているが (それぞれ個々にデプロイすることはできるが)、それらが作られたデータレジストリに関する情報（ロケーションとソース）も維持しています。従って、チャートのクエリーを編集するとき（詳細はセクション 5.2.4）、変更をデータレジストリに保存するオプションがあります。この機能を使うためにはデータレジストリを最新に維持しておく必要があります。つまり、クエリーの変更はレストリに保存するようにしてデータビュー構造の変更はデータビュークエリーに反映されるようにします。（詳細はセクション 5.2.3.2）

この機能を使うには、まずチャートテンプレートに反映させたいデータレジストリへの変更を行います。これらの変更はデータベース接続情報、テキスト、XML データファイルのファイルロケーション、テンプレートに引き渡したいデータビューまたはクエリーの変更などです。インストレーション間でチャートを移動する場合はデータ

レジストリファイルが新しいインストレーションの同じ相対ロケーションに移動される必要がありますので注意してください。また、次のレジストリ（.qry/.dvw/.ddt ）に関連するクエリは、新しいインストレーションの/queries/ディレクトリへ移動される必要があります。（クエリーファイル名はレジストリの名前で始まります。）

データソースマネージャのアップデート（更新）メニューから'charts' を選びます。更新したいチャートを選択できるダイアログが開きます。



データソース更新のチャート選択

更新するチャートを選択するには、まずチャートが入っているディレクトリをブラウズします。ディレクトリを選択したあと、チャートテンプレートがダイアログの左手側に表示されます。更新したいテンプレートを選び'Add' ボタンをクリックします。選びたいテンプレートまでディレクトリをナビゲートします。'Connection Info Only' オプションを選ぶと接続情報だけ更新します。（データベース URL、ドライバ、ユーザ名、パスワード、XML f ファイル、テキストファイル、Java クラス） クエリとデータビュー情報はテンプレートでは更新しません。

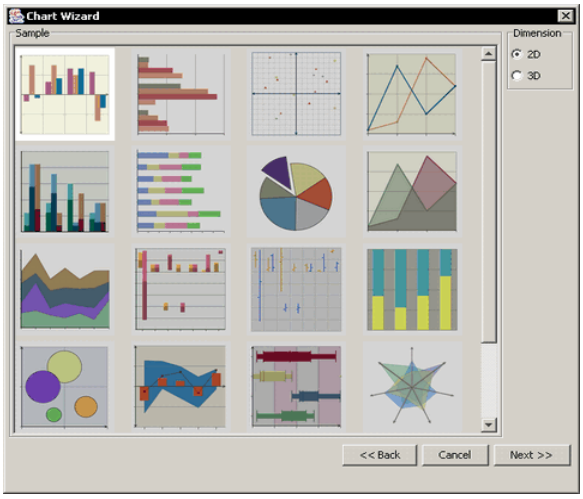
更新したいテンプレートの選択が終ったら、'OK'をクリックします。そうすると更新の処理が始まります。ダイアログがどのテンプレートが更新されているか、どんなエラーが出てくるか表示します。

インストレーションのルートディレクトリには *UpdateDataSources* という名のログファイルがあり、実行スクリーンの内容が書かれています。いろいろな理由で更新できなかったチャートは、チャートデザイナーのオプションを使ってマニアルで更新することができます。

*注意 – カレントデータレジストリのチャートの編集できます。カレントレジストリのソースデータを取らないチャートを選択した場合、それらは無視されます。

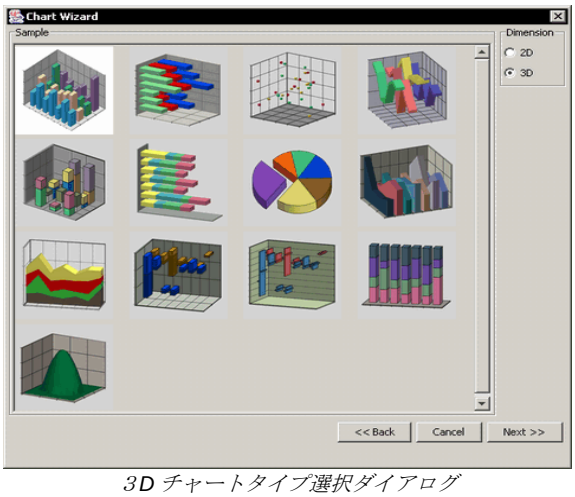
6 チャートタイプとデータマッピング

使いたいデータソースを選択したら、チャートウィザードでの次のステップは使用したいチャートタイプの選択です。チャートタイプを指定するダイアログが表示されます。.



2D チャートタイプ選択ダイアログ

各チャートタイプは全ての種類のデータを適切に表現するためにそれぞれ異なる方法でデータポイントを表示します。異なる種類のチャートは次元によって分類されます。２D で表されるチャートは 19 タイプ、3D で表されるチャートが 13 タイプあります。さらに、ユーザがセカンダリの値（バリュー）/シリーズを追加することにより、混合/結合させたさまざまなタイプのチャートを作成することができます。チャートタイプダイアログの右側にあるトグルボタンをクリックすることによって、２D と３D を切り替えることができます。



3D チャートタイプ選択ダイアログ

このダイアログではイメージを選び、'Next'をクリックするか、チャートイメージの上でダブルクリックすることにより、チャートタイプを選択することができます。次のチャートタイプから選ぶことができます。

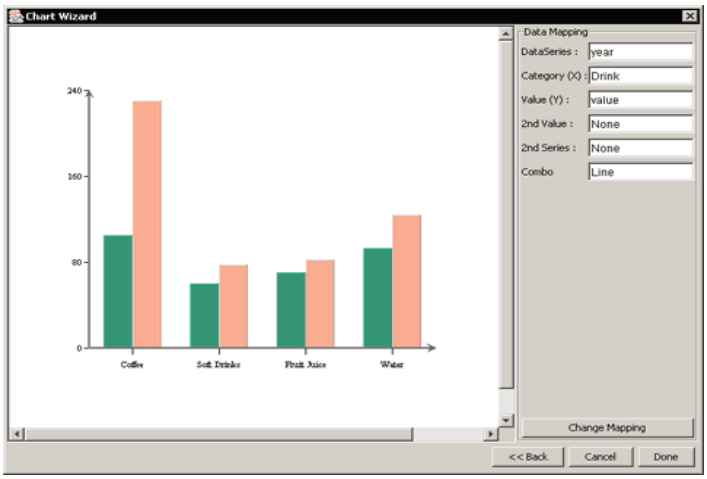
- カラム
- XY(Z) 分散
- ライン
- スタックカラム
- スタックバー
- パイ
- エリア
- スタックエリア
- High-Low
- HLCO
- パーセンテージカラム
- サーフェス（3D のみ）
- バブル（2D のみ）
- オーバーレイ（2D のみ）
- ボックス（2D のみ）
- レーダー（2D のみ）
- ダイアル（2D のみ）
- ガント(（2D のみ）
- ポーラー（2D のみ）
- バー

各チャートタイプの説明はこの章のセクション 6.2 から始まります。

ゲージの詳細については、この章のセクション 6.19.2 を参照してください。

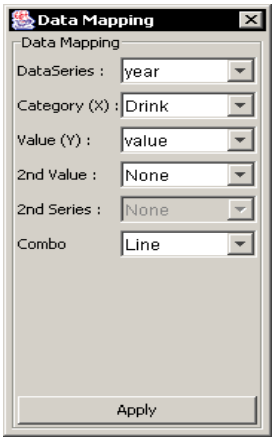
6.1 データマッピング

チャートタイプを選択したら、次はチャートのデータマッピングをしてします。データマッピングは選択したデータソースがチャートに描かれるための手法です。データマッピングの基礎は 3 章のセクション 3.2 に説明してあります。チャートウィザードのデータマッピング画面ではマッピングのオプションを設定とその設定した結果をプレビューすることができます。



データマッピングダイアログ

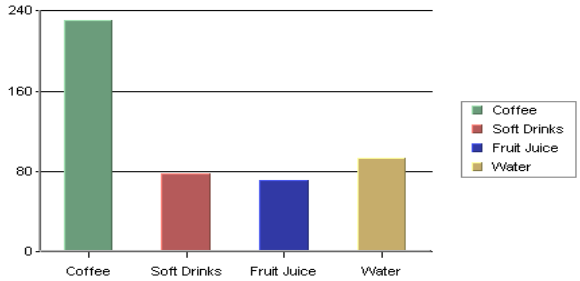
ダイアログの左側にはチャートのプレビューが表示されます。右側にはデータソースのどのカラムがチャートのエレメント（シリーズ、カテゴリ、バリューなど）に選択されているかを表しています。EspressChart の規定値では、（データ型を基本に）最初にプロットするのに利用可能なカラムを選択します。マッピングを変更するには 'Change Mapping' ボタンをクリックします。新しいダイアログが表示されて、チャートエレメントに別のカラムを指定することができます。



マッピングの変更ダイアログ

このダイアログでは、各チャートエレメントがドロップダウンメニューになっているので、そこから新たなカラムを選択することができます。変更いたいカラムを選択したら、'Apply' ボタンをクリックすると、その変更がマッピング画面に反映されます。チャートタイプによってマッピングオプションも異なりますので、この章の 6.2.1 から説明していきます。

6.2 カラムチャート

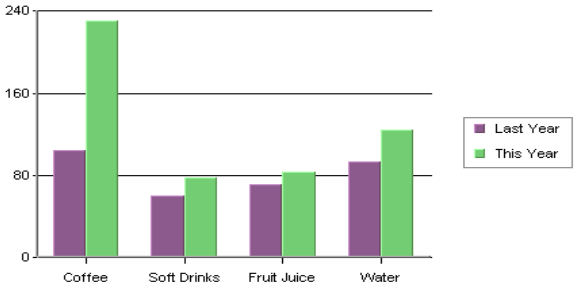


カラムチャート (データシリーズなし)

カラムチャートはデータテーブルの各行を垂直な棒（またはカラム）として表示します。カテゴリは x 軸に沿って表示され、バリュー（値）は y 軸に沿って表示されます。カラムチャートは異なるグループの独立した値を比較するのに適しています。各グループは異なるカラーで表現されます。

2D のチャートでは、データシリーズが選択された場合、ひとつのカテゴリのシリーズ全体がxy 面上に表示されます。 3D のカラムチャートが使用される場合、データシリーズ内のすべての垂直の棒が同じカラーでz 軸上に表示されます。チャートにデータシリーズが存在しない場合、カテゴリは異なるカラーで表現されます。

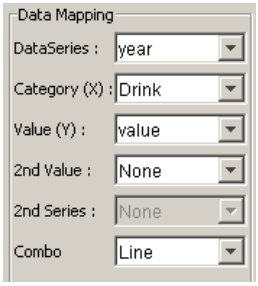
次の例では、飲み物（Drink）がカテゴリとして、値がバリューとして選択しています。（ほとんどの例は EspressChart インストレーションに入っている Sample.dat を使用します。）年はデータシリーズとして選択されています。各名前には 2 つのカラムが表示されます。ひとつは"last year" （昨年）そしてもうひとつは"this year" （今年）で、2 つの値がデータシリーズカラムで表示さています。



データシリーズのあるカラムチャート

6.2.1 データマッピング

カラムチャートのデータマッピングはまったく簡単でセクション 3.2 で最初に説明した例に似ています。カラムチャートには次のオプションがあります。



カラムチャートのマッピングオプション

- **Data Series（データシリーズ）：**チャートのデータシリーズの数を決定する値を持つデータカラムを選択します。データシリーズの各エレメントは、同一の色やその他の属性を伴って表示されます。

- **Category (X)（カテゴリ）：**カテゴリを決定する値を持つデータカラムを選択します。

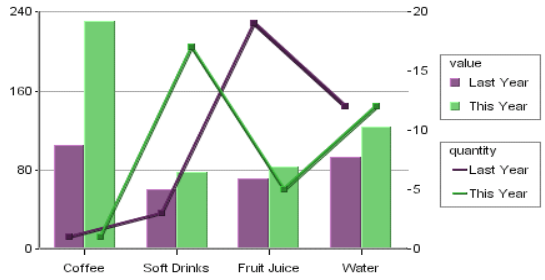
- **Value (Y)（バリュー）：**各カテゴリに値を供給するデータカラムを選択します。

- **2nd value（セカンドバリュー）：**コンビネーションチャートを作成するための二次的な値を追加します。

- **2nd Series（セカンドシリーズ）：**副チャートのシリーズとなる他のカラムを選択します。このオプションは副チャートがオーバーレイチャートである場合にのみ適用されます。

- **Combo（コンボ）：**副チャートのタイプを選択します。カラムチャートのコンボオプションは“ラインチャート”と“オーバレイチャート”です。

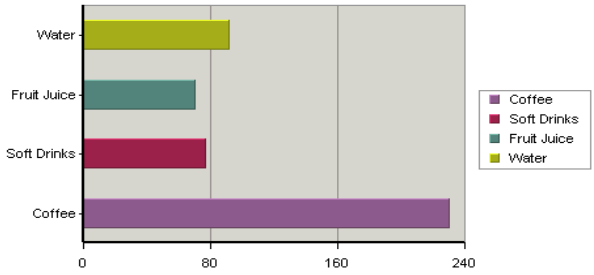
最後の 3 つのデータマッピングオプションはチャートにセカンドバリューを追加するものです。EspressChart ではパイ、レーダー、バブル、ダイアル、サーフェス、および分散の各チャートを除くすべてのチャートタイプに二次値（をサポートしています。下記例は量をセカンドバリューとしてカラムチャートに追加したものです。



セカンドバリューのあるカラムチャート

ご覧の通り、セカンド軸ラベルはプロットエリアの右側に示されています。（この軸の表示は選択できます。）通常、セカンドバリューは同じカテゴリを使い、シリーズも最初のものと同じ値を使います。しかし、2D カラム、スタックカラム、スタックエリア、Hight-Low、HLCO、パーセンテージカラムの各チャートはオーバーレイとの組み合わせをしていすることができ、セカンドシリーズも指定が可能です。 オーバーレイチャートの詳細については、セクション 6.16 を参照してください。

6.3 バーチャート

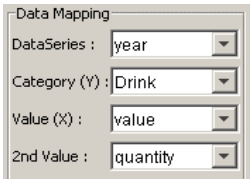


バーチャート

バーチャートは本質的にはカラムチャートと同じものですが、カラムチャートでは垂直のバーが表示されるのに対してバーチャートでは水平のバーが表示される点が異なっています。バーチャートでは、カテゴリは y 軸上に表示され、値は x 軸上に表示されます。

カラムチャートと同様に、データシリーズが選択された場合はひとつのカテゴリのシリーズ全体が xy 面上に表示されます。3D のバーチャートが使用されている場合、データシリーズ中のすべての水平のバーは同じカラーで表示されます。各カテゴリは異なるカラーで表示されます。チャート中にデータシリーズが存在しない場合、カテゴリは異なるカラーで表現されます。

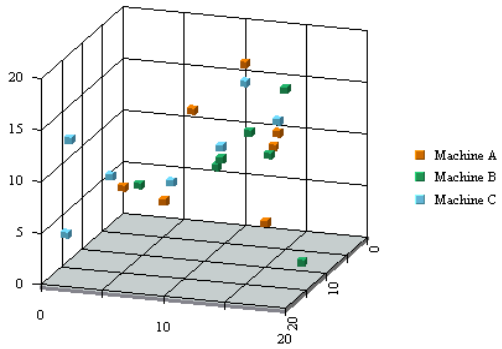
6.3.1 データマッピング



バーチャートのマッピングオプション

このチャートタイプにおけるマッピングはカラムチャートに似ていますが、カラムチャートにおけるカテゴリ (X) とバリュー (Y) はそれぞれカテゴリ (Y) とバリュー (X) になります。これは、カラムチャートでは値が垂直方向に表されるのに対して、バーチャートでは水平方向に表されるためです。バーチャートではセカンドシリーズ およびコンボオプションは使用できません。なぜなら、バーチャートと組み合わせ可能なのは、ラインだけだからです。

6.4 XY(Z) 分散チャート

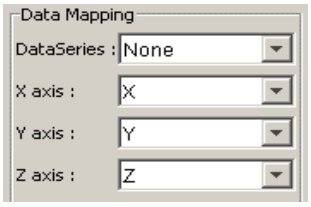


XY(Z) 分散チャート

xy(z) 分散チャートでは、データテーブルで選択された各行が二次元または三次元の空間でポイントとして表されます。従って、各カラムは数値かまたは日付/時間の値を持っている必要がります。マーカーは各ポイントを表します。データテーブルの各行にあるデータカラムが空間上におけるマーカーの位置を決定します。

オプションとして、マーカーをグループに分割するために他のデータテーブルのカラムを選択することができます。各グループのエレメントはこのカラム上で同一の値を持っており、データシリーズカラムとして参照されます。同じグループのマーカーは同じ色と形の表示属性で表示されます。分散チャートの x 軸目盛は、連続した直線になっています。つまり他のチャートタイプと異なり、データポイントは x 軸にそって同等の距離だったり、そうでなかたっりします。

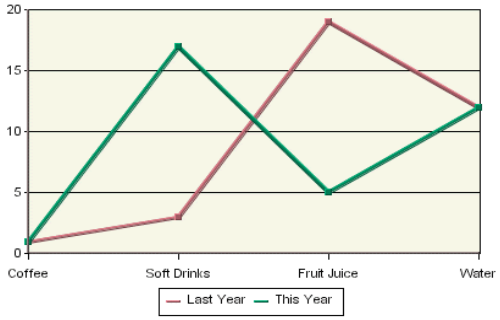
6.4.1 データマッピング



分散チャートのマッピングオプション

分散チャートでは、x 軸、y 軸、および z 軸の値がポイントのそれぞれ x 、y 、および z 座標を決定します。データシリーズボックスによって、チャートのデータシリーズの数を決定するデータカラムを選択することができます。ひとつのデータシリーズ中の各エレメントは、同一のカラーやマーカーなどの描画属性のセットを伴って描画されます。

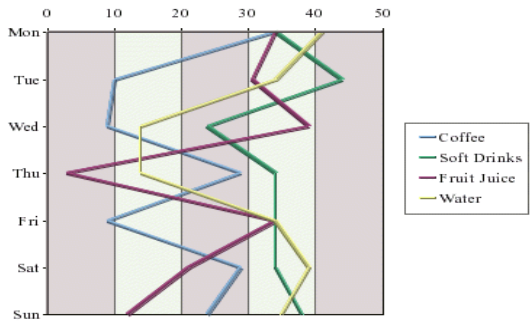
6.5 ラインチャート



ラインチャート

ラインチャートはカラムチャートと同様の方法でデータを表現します。２D ラインチャートは各カテゴリのデータポイントをマーカーが表します。バリューカラムは数値でなければならず、これによってマーカーの高さが決定されます。各マーカーは線で結合されています。チャートにデータシリーズがある場合、シリーズの各エレメントにそれぞれの線が引かれます。

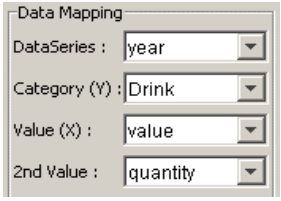
EspressChart では 2D ラインチャートは規定値の水平（横）設定に加え、垂直（縦）にも表示させることができます。チャートデザインでこの機能を使用するにはラインチャートを作成しフォーマットメニューの'Chart Options' を選択し、Vertical（垂直方向）を選択します。チャートは時計回りに 90 度回転します。



縦のラインチャート

3D のラインチャートは、2D のラインチャートを視覚的にのみ拡張したものです。追加の情報はありません。マーカーは表示されず、細いラインに代わってより太いラインが z 軸上に分散して表示されます。

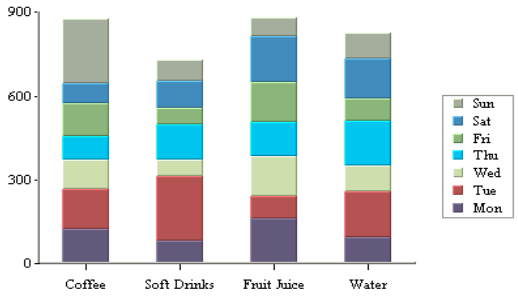
6.5.1 データマッピング



ラインチャートのマッピングオプション

ラインチャートのデータマッピングはカラムチャートとほぼ全く同じです。ただ、ラインチャートにはセカンドシリーズとコンボオプションがありません。これはラインチャートと組み合わせることができるのはラインだけだからです。他のチャートタイプ（バー、カラム、スタックカラムなど）と組み合わせてラインチャートを作るには、他のチャートタイプを最初のチャートとして選び、それからラインチャートをコンボオプションで選びます。

6.6 スタックカラムチャート



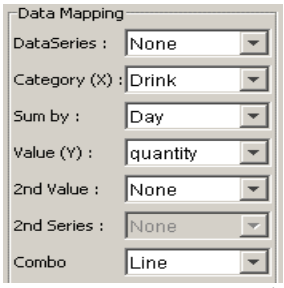
スタックカラムチャート

スタックカラムチャートはカラムチャートの変型で、各々の垂直のカラムがバーの積み重ね（スタック）によって構成されます。データテーブルの中で選択された各行がチャートカラムのコンポーネントとして表現されます。2D スタックカラムチャートでは、x 軸はカテゴリ軸、y 軸はバリュー（値）軸となります。バリュー軸は数値でなければならず、これによって各バーの長さが決定されます。第三のカラムは総計カラムで、独立したカテゴリを表します。任意のカテゴリ値のスタックは、そのカテゴリ値の総計値を積み上げることによって形成されます。チャートカラムのスタックの各コンポーネントは独自の総計値を持ち、独自のカラーで表されます。データテーブルの各行はカテゴリおよび総計カラムの 2 つの固有な値を持っていなければなりません。マイナスの値を持つコンポーネントはプラス値のコンポーネントから分離され、x 軸の下へ「マイナス」の方向に積み上げられます。

上の例では、飲み物のタイプがカテゴリ値として選択され、曜日が総計値を表しています。データシリーズカラムと呼ばれる第四のカラムを基に、（z 軸上に表示される）もうひとつの独立したカテゴリをオプションで指定することができます。この場合、データテーブルの各行はカテゴリ値、総計、およびデータシリーズカラムの 3 つの固有な値を持っている必要があります。2D のチャートでは、ひとつのカテゴリのデータシリーズ全体が xy 面上に隣り合って表示されます。3D のスタックカラムチャートでは、データシリーズは z 軸に沿って表示されます。

6.6.1 データマッピング

スタックカラムチャートでは各カテゴリがコンポーネントに細分化されています。従って、追加 dum-by（総計）オプションがあり、チャートの sum-by（総計）値を決定するのに使用されます。



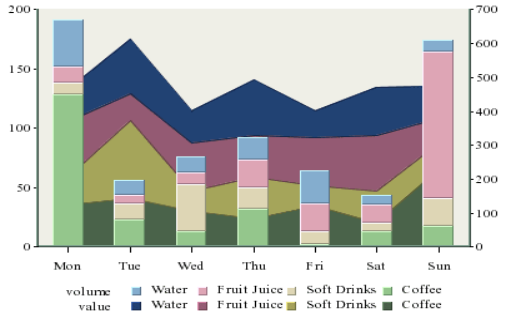
スタックカラムチャートのマッピングオプション

データマッピングオプションは次のものがあります。

- **Data Series（データシリーズ）：**チャートのデータシリーズの数を決定する値を持つデータカラムを選択します。ひとつのデータシリーズ中の各エレメントは同一のカラーで描画されます。.
- **Category (X)（カテゴリ）：**Al カテゴリを決定する値を持つデータカラムを選択します。このデータカラムから得られる値によって x 軸の目盛が決定されます。データシリーズの各カテゴリはチャート内で個別のカラムとして描画されます。

- **Sum-by（総計）：**各カテゴリ内のコンポーネントを決定する値を持つデータカラムを選択します。このカラムから得られる各値によって、カラムのスタックコンポーネントが決定されます。
- **Value (Y)（バリュー）：** 各カテゴリの値を与えるためのデータカラムを選択します。.
- **2nd value（セカンドバリュー）：**A コンビネーションチャートを作成するためのセカンドバリューを追加します。
- **2nd Series（セカンドシリーズ）：**副チャートのシリーズにするもうひとつのカラムを選択します。このオプションは副チャートがオーバーレイチャートの場合のみ適用できます。
- **Combo（コンボ）：**副チャートのチャートタイプを選択します。スタックカラムチャートのコンボオプションは、ラインチャート、スタックエリアチャート、オーバーレイチャートのいずれかです。

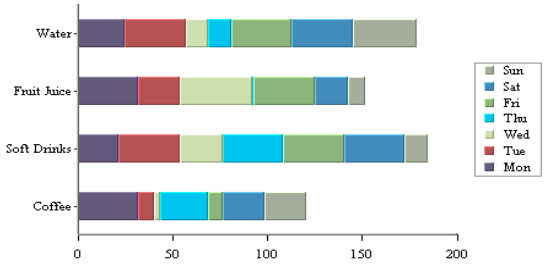
スタックカラムチャートは独自のコンビネーションオプションを持っていて、スタックエリアチャートとしてセカンドバリューを描くことができます。どちらのバリューも同じカテゴリと sum-by（総計）値を共用します。このチャートはデータの異なるバリューでコンポーネントベースのカテゴリを比較できます。



スタックカラムースタックエリアコンビネーションチャート

このコンビネーションチャートではストリップエリアを効果的に見せるために、特別な sum-by（総計）コンポーネントを隠すように指定しすることができます。

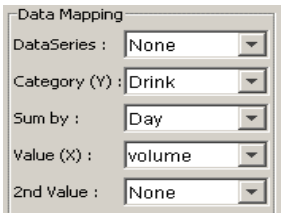
6.7 スタックバーチャート



スタックバーチャート

スタックカラムチャートのようにスタックバーチャートは積み重ね（スタック）られた異なるバリューの総計（データソースの総計カラム）としてチャートカテゴリを表示します。違うのは、スタックバーチャートはカテゴリが y 軸に、バリューが x 軸に表されるということです。

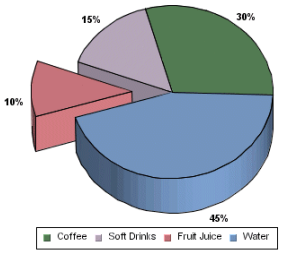
6.7.1 データマッピング



スタックバーチャートデータマッピングオプション

このチャートタイプのマッピングはスタックカラムチャートのマッピングと同様ですがスタックカラムチャートにおけるカテゴリ (X) とバリュー (Y) はそれぞれカテゴリ (Y) とバリュー (X) になります。これは、カラムチャートでは値が垂直方向に表されるのに対して、バーチャートでは水平方向に表されるためです。スタックバーチャートではセカンドシリーズ およびコンボオプションは使用できません。これはタクバーチャートと組み合わせることができるのがラインだけだからです。

6.8 パイチャート



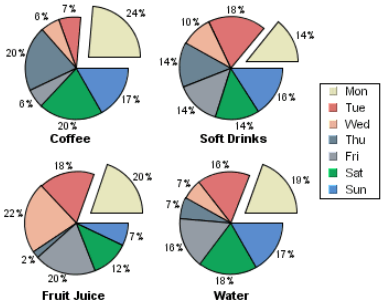
パイチャート

パイチャートでは、データの各行がパイセクタとして表現されます。パイチャートではカテゴリカラムとバリュー（値）カラムが必要になります。バリューカラムは数値である必要があります。カテゴリカラム中にある独立した値の数によって、チャートのパイのセクタ数が決定されます。すべての値が合計され、全体における割合に応じて各スライスが割り当てられる角度が決定されます。このように、各カテゴリのバリューカラムがそのカテゴリを表すスライス部分のサイズを決定します。.

3D のパイチャートは 2D のパイチャートの単なる拡張で、追加の情報はありません。

パイチャートはデータシリーズを持つこともできます。データマッピングでシリーズを指定した場合、各カテゴリエレメントに対して別のパイが描かれ、それぞれがシリ

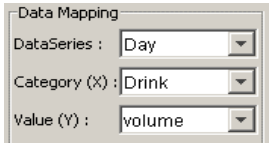
ーズエレメントから成っています。データシリーズのあるパイチャートは下記のようになります。



データシリーズのあるパイチャート

シリーズが表示されている場合、すべてのパイは同一のプロットエリアに描かれ、プロットの比率でサイズが調整されます。異なるパイを積み上げるか、または線でそれらを描くかのオプションがあります。

6.8.1 データマッピング



パイチャートのマッピングオプション

パイチャートのマッピングは次の通りです。

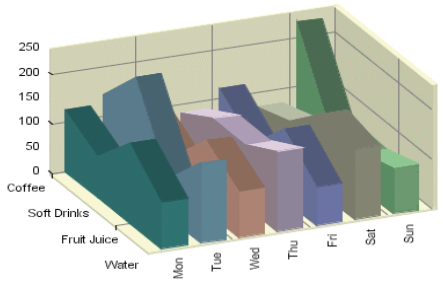
- **Data Series（データシリーズ）：**チャートのデータシリーズの数を決定する個別のバリューを持つデータカラムを選択します。データシリーズの各エレメントは同じ色を使って表現されます。

- **Category (X)（カテゴリ）：**様々なカテゴリを決定する個別の値を持つデータカラムを選択します。.

- **Value (Y)（バリュー）：**各カテゴリに値を供給するデータカラムを選択します。

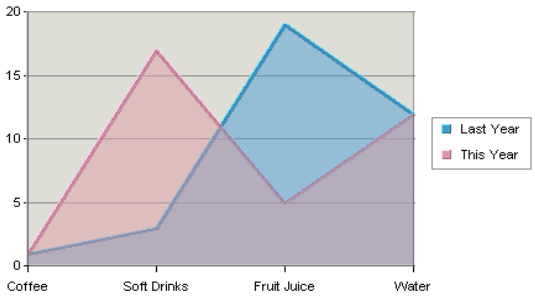
パイチャートと組み合わせられるチャートはありませんのでセカンドバリューオプションはありません。

6.9 エリアチャート



3D エリアチャート

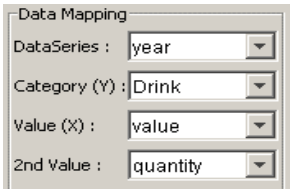
3D エリアチャートは、カラムチャートの変型として見ることができます。次の方法で、3D エリアチャートを 3D カラムチャートから作成することができます。各データシリーズ（z 軸）値のすべてのカラムの頂点が太い線で結ばれます。次にカラムが削除され、各線の下の xy 面上のエリアがそれぞれ個別のカラーで塗りつぶされます。



2D エリアチャート

2D エリアチャートは、実質的には 3D エリアチャートを z 軸に沿って見た形を平らにしたものです。結果として 2D チャートはいずれかのデータシリーズ値がすべて隠れてしまう可能性があるので、注意して使用する必要があります。上の 2 つの図を比較すればこのことがわかります。2D チャートでは各シリーズのいくつかの部分がその前にあるシリーズによって隠されています。この問題を解決するにはデータシリーズの順番を変更するか（セクション 7.8.3 を参照）、上記の例のように半透明になるように設定します（セクション 7.8 を参照）。

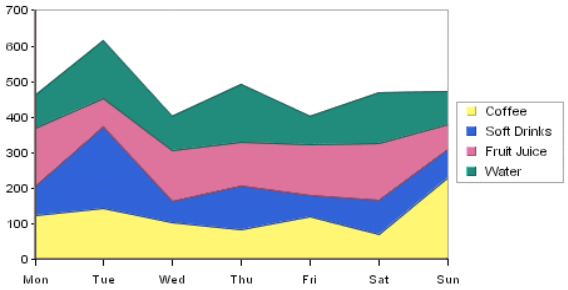
6.9.1 データマッピング



エリアチャートのマッピングオプション

エリアチャートのデータマッピングはほとんど全くカラムチャートのマッピング（セクション 6.2.1）と同じです。ただし、セカンドシリーズとコンポオプションがありません。これはエリアチャートの組み合わせとして可能なのはラインだけだからです。

6.10 スタックエリアチャート



スタックエリアチャート

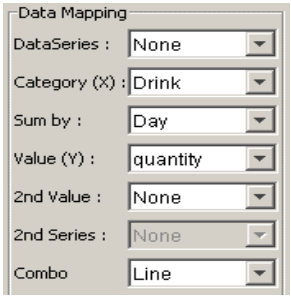
スタックエリアチャート

2D スタックエリアチャートは、2D のスタックカラムチャートからデータシリーズを除いた変型として見ることができます。このチャートは次の方法で、スタックカラムチャートから作成することができます。同じカラーを持つ各スタックコンポーネントの頂点が線で結ばれます。スタックカラムが削除され、線の各セットの間がそれぞれ個別のカラーで塗りつぶされます。

3D のスタックエリアチャートはデータシリーズを持つことができ、上記の方法を利用して 3D のスタックカラムチャートより作成することができます。この場合、独立した各データシリーズ値のスタックカラム（固定された x 軸の値）が個別に結合され、複数のスタックの山が作成されます。

2D および 3D のスタックエリアチャートは、共に各々に必要な同様のデータのセットを使用して作成することができます。ただし先に述べた通り、2D スタックエリアチャートはデータシリーズを持つことができないことに注意してください。

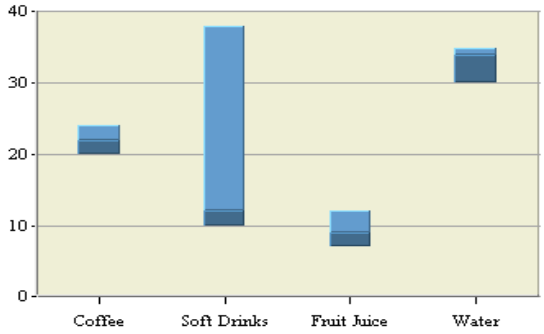
6.10.1 データマッピング



スタックエリアチャートのマッピングオプション

スタックエリアチャートのデータマッピングはスタックカラムのデータマッピング（セクション 6.6.1）ほとんど全く同じです。ただし、２D スタックエリアチャートにはデータシリーズが使えず、コンボオプションはラインチャートとオーバーレイチャートだけです。

6.11High-Low チャート

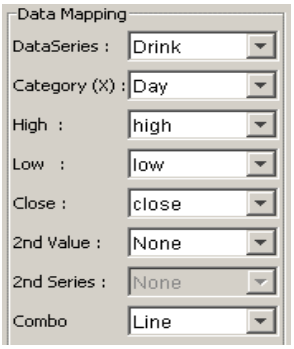


High-Low チャート

High-Low チャートはカラムチャートの変型で、1つのカラムのみでなく2つのカラムを使用することによって値の上限（High ）と下限（Low ）を表します。このチャートのデータは、少なくとも3つのカラム（カテゴリ、上限、下限）を含んでいる必要があります。カテゴリカラムはどのようなタイプの値でも構いませんが、上限および下限カラムは数値である必要があります。

High-Low チャートのもうひとつのオプションとして、“クローズ”カラムと呼ばれるデータカラムがあります。クローズカラムが使用される場合には、上限と下限値との間のいずれかの位置にクローズ値が存在します。バーの下限ポイントとクローズポイントとの間の部分は、クローズポイントと上限ポイントとの間の部分と同系色でより暗いカラーによって表示されます。他の多くのタイプのチャートと同様に、High-Low チャートはデータシリーズカラムを含みます。

6.11.1 データマッピング



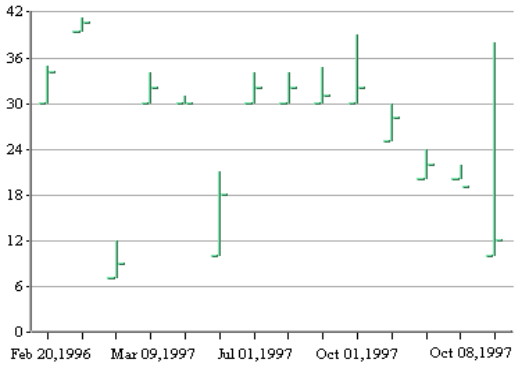
High-Low チャートマッピングオプション

High-Lowチャートのデータマッピングは、カラムチャートと同様です。同じ方法ででシリーズとカテゴリカラムを選びます。High-Lowチャートが異なるところは、上限と下限の少なくとも2つのバリューカラムを指定する必要がありことです。各カテ

ゴリが描かれる2つのポイントです。限、下限、オープン、クローズの各値、および二次的なデータを決定するためのフィールドを含んでいます。

High-Low チャートおよび HLCO チャートのデータパネル上限、下限、オープン、クローズの各値について、任意の数値フィールドを使用することができます。A third value called "Close" can also be specified. Combo options for high-low charts are "Line", "Column", and "Overlay".

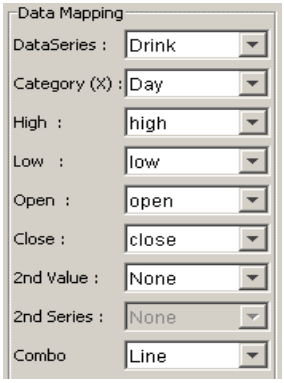
6.12HLCO チャート



HLCO チャート

HLCO (High Low Close Open) チャートは、先述の High-Low チャートに似ていますが、「オープン」カラムを含んでいる点が異なります。このチャートは株価や日中の気温のように、不定期的に変動するデータを表現するのに向いています。HLCO チャートは 2D と 3D の両方で使用できるキャンドルスティックフォーマットで表示されます。この機能は 7 章で説明されています。

6.12.1 データマッピング



HLCO チャートのマッピングオプション

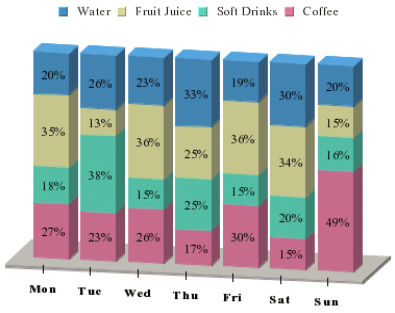
HLCO チャートのデータマッピングは High-Low チャートのデータマッピングと同様です。違いは High-Low チャートでは 2 つの異なるバリューカラムを指定するところを HLCO では 4 つの異なるカラム（"High", "Low", "Open", "Close" カラム）を指定する必要があります。コンボオプションはラインチャート、カラムチャート、オーバーレイチャートです。

よくあるチャートとしては、株価と出来高の両方をひとつのチャートに描いたものがあります。このようなチャートは EspressChart の HLCO とカラムチャートのコンビネーションを使って作成することができます。

HLCO-カラムコンビネーションチャート

このチャートは 3 ヶ月間の株価と出来高をチャートにした例です。

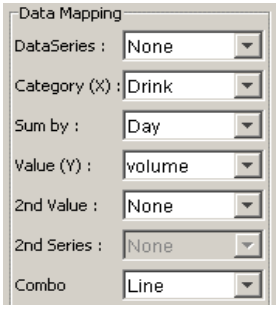
6.13 パーセンテージカラムチャート



パーセンテージカラムチャート

パーセンテージカラムチャートはパイチャートとカラムチャートを併せた変型チャートとして見ることができます。チャートの各カラムはパイチャートのパイのひとつに相当します。パーセンテージカラムチャートは x 軸の値に対応するカテゴリカラムを持っています。この場合、総計カラムはカテゴリを表現し、バリューカラムはパイチャートのバリューカラムと同じです。

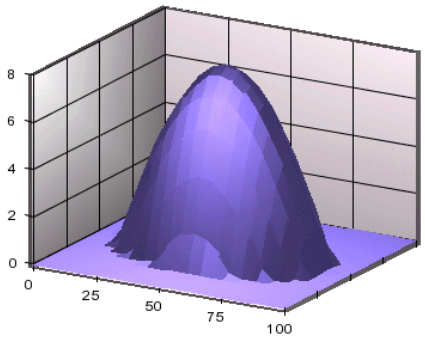
6.13.1 データマッピング



パーセンテージカラムチャートのマッピングオプション

パーセンテージカラムチャートのデータマッピングはスタックカラムチャートのデータマッピング（セクション 6.6.1）と同じです。唯一の違いは個々バリューの代わりに総計コンポーネントがバリューカラムのパーセンテージで描かれることです。パーセンテージカラムチャートのコンボオプションはラインチャートとオーバーレイチャートです。

6.14 サーフェス



Surface Chart

3D サーフェスチャートは3 つの数値のセットを使用して三次元空間に滑らかなサーフェス（表面）を形作るチャートで、科学やビジネスのチャートに利用されます。各データポイントについて、3 つのデータのうちの2 つが水平面における座標を決定し、3 つめのデータがデータポイントの垂直位置を決定します。入力データは下記に示すような長方形のマトリックスにするか、または3 つのカラムを並べて各カラムがそれぞれの軸を表すようにすることもできます。下記のサンプルデータでは、一番目の行（カラムヘッダ）と左カラム（行ヘッダ）が、サーフェスが表示される平面を形成する軸の座標値を表します。チャートをこの平面の上から見ると、隣り合った4 つの各ポイントによってグリッドが形成されていることがわかります。このグリッドが描かれなければ、任意のデータセットがサーフェスチャートを描画することはできません。

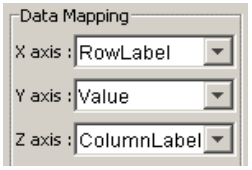
以下はサンプルデータです。

	0	20	40	60	70	80	100
20	0	0	0	0	0	0	0
30	0	10	10	10	10	10	10
40	0	10	25	25	25	25	10
80	0	10	25	30	30	30	25
90	0	10	25	30	30	30	25
100	0	10	10	25	25	25	10

サーフェスチャートは水平方向の距離を適切な大きさに調節するため、カラムヘッダと行ヘッダの値が等しく分割されている必要はありません。このデータセットは、反転した歪んだ円錐を作成します。サーフェスチャートはアニメーションで使用すると大変効果的です。

サーフェスチャートは x 、y 、z 座標のセットを使用して 3D チャートを描画します。垂直の軸は y 軸と呼ばれ各 y 値は x と z の関数 f(x,z)によって表されます。X-Z 面（水平面）上のデータは単純な正方形のマトリックスのように見えます。サーフェスチャートはデータシリーズをサポートせず、2D チャートに変換することもできません。

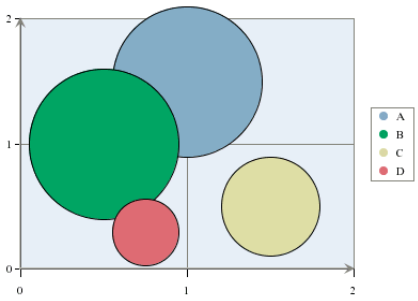
6.14.1 データマッピング



サーフェスチャートのマッピングオプション

サーフェスチャートのデータマッピングは３D 分散チャートと同様です。x 軸、y 軸、および z 軸の値がポイントのそれぞれ x 、y 、および z 座標を決定します。しかし、分散チャートと異なり、サーフェスチャートはデータシリーズをサポートせず、2D チャートに変換することもできません。

6.15バブルチャート



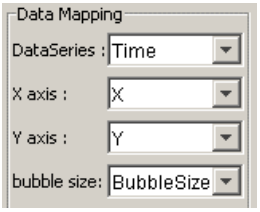
バブルチャート

バブルチャートはバブルの位置だけでなく、その大きさも情報を持つことができるデータ表現を可能にします。データポイントは x-y 座標および第三のポイント、つまり円またはバブルの半径、によって表されます。

このチャートは 2D でのみ可能です。バブルチャート及びそのディスプレイオプションに関しては 7 章セクション 7.9.1 を参照してください。

6.15.1 データマッピング

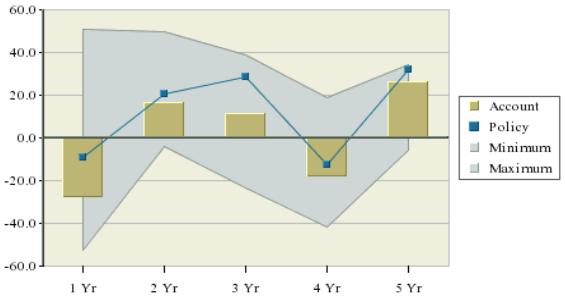
バブルチャートのデータマッピングは 3D 分散チャートのマッピングと同様ですが、z 軸の代わりに Bubble Size （ここでは半径）となる点が異なります。



バブルチャートのマッピングオプション

分散チャートのように、マッピングうまく行われるためにはカラムは数値である必要があります。

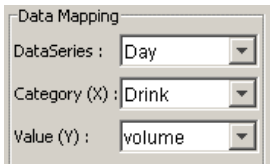
6.16 オーバーレイチャート



オーバーレイチャート

EspressChart は「オーバーレイチャート」と呼ばれる特殊なチャートをサポートします。これは、共通のカテゴリ軸を持つ 2 つ以上のチャートを組み合わせることができるチャートです。データシリーズ中のひとつのエレメントとを表すために異なるチャートを使用することができます。このことによってより自由なチャート作成を可能にすると共に、より多くの情報を表現することができます。オーバーレイチャートでサポートされるチャートタイプはカラムチャート、エリアチャート、およびラインチャートです。オーバーレイチャート内には 2D のチャートのみを含むことができます。

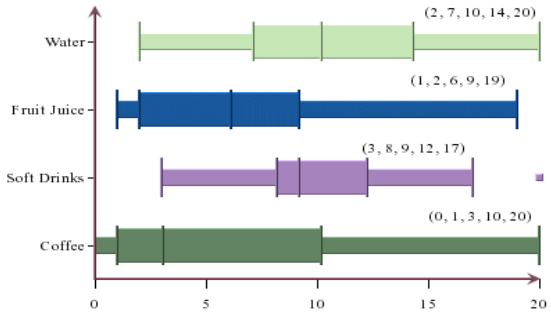
6.16.1 データマッピング



オーバーレイチャートのマッピングオプション

オーバーレイチャートのマッピングはカラムチャートのマッピング（セクション 6.2.1で説明）と同様ですが、オーバーレイチャートはデータシリーズの各エレメントをそれぞれ別のチャートとして描きます。オーバーレイチャートにはセカンドバリュー、セカンドシリーズ、及びコンボオプションは適用されません。 オーバーレイチャートはセカンドバリューをサポートしていません。しかしながら異なるシリーズエレメントとは別の軸上に描かれます。オーバーレイチャートのプロッティングオプション については、セクション6.9.3を参照してください。

6.17 ボックスチャート



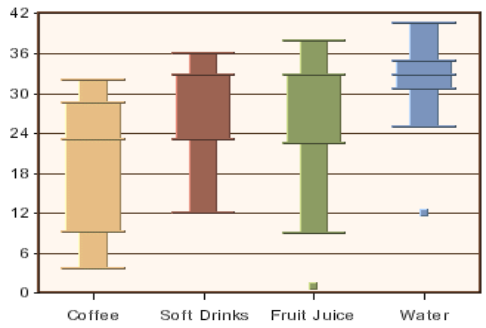
ボックスチャート

EspressChart はボックスチャート、あるいはボックスアンドウィスカーチャートと呼ばれる統計型のチャートを作成することができます。基本的に、ボックスチャートは統計の要点が一目で見て取れるような表現を可能にします。ヒストグラムによって観察されるべき値の分布が示されるため、値のピークや最小値、最大値、データの固まり方などを確認することができます。その一方で、ボックスチャートはデータ分布

の概要を 4 分割の割合（最小のデータポイント、25 パーセント、中間点、75 パーセント、および最大のデータポイント）で表現します。

ボックス内の中心にある線は中間点を示します。その両側にある他の線は 25 パーセントずつの増加（または減少）を表します。最小および最大ポイントは確認された最小および最大ポイントで、分離点とは異なります。ボックスの端から最小または最大の分離点に向かって延びる線はウィスカー（ひげ）と呼ばれることがあるため、ボックスアンドウィスカーチャートと呼ばれるのです。ボックスの長さの 1.5 倍（またはそれ以上）の値、つまり 25 パーセントと 75 パーセントのポイントの差にあたる値は分離点と呼ばれます。分離点は計算によって導かれ、ボックスから離れた点として表示されます。他にボックスによる表現の利点としては、ひとつのチャート内でボックスを積み上げることによって異なるデータセットの概要を比較できる点があります。

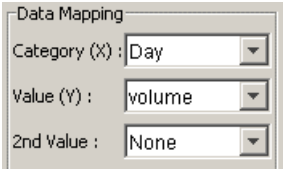
EspressChart では、ボックスチャートを既定値の水平方向に加えて垂直に表示することもできます。チャートデザイナー でこの機能を使用するには、ボックスチャートをデザインした後、フォーマットメニューからチャートオプションを選択し、Vrtical（垂直）ます。



垂直のボックスチャート

このチャートは 2D 形式でのみ使用することができます。

6.17.1 データマッピング



ボックスチャートのマッピングオプション

ボックスチャートのマッピングは次のようになります。

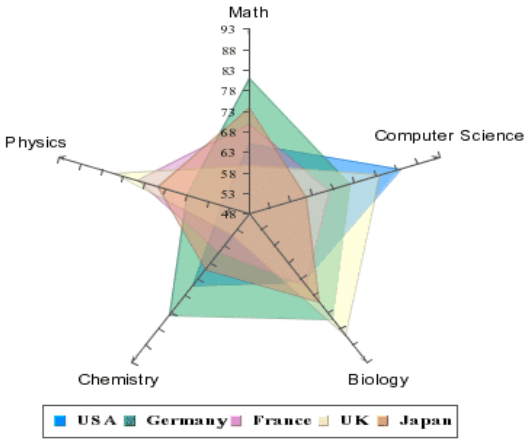
- **Category (X)（カテゴリ）：** カテゴリを決定する固有の値を持つデータカラムを選択します。

- **Value (Y)（バリュー）：**各カテゴリに値を供給するデータを選択します。これらの値は最小データポイント、２５パーセントのポイント、中間ポイント、75 パーセントのポイント、最大データポイントを計算するのに使われます。

- **2nd value（セカンドバリュー）：**コンビネーションチャートを作成するためのセカンドバリューを追加します。

セカンドシリーズ、とコンボオプションはボックスチャートでは使用できません。これはボックスチャートと組み合わせられるのがラインしかないからです。

6.18 レーダチャート



レーダーチャート

レーダーチャートは、異なるソースから得られる業績や測定値、統計などの比較を行うときに便利なチャートです。例えば上のような、工業国における子供達の数学、コンピュータサイエンス、生物学などの科目のテスト結果の中間値を比較するチャートを作成することができます。レーダーチャートは複数の軸を持っており、それに沿ってデータが描画されます。各軸がカテゴリに相当します。データはそれらの軸上のポイントとして表示されます。ひとつのデータシリーズに属しているポイントは結合またはエリア化され、その内側が色で塗りつぶされます。軸の中心に近いポイントは低い値を示し、軸の先端に近いポイントは高い値を表します。チャートが表す内容によっては、所得に対する物価、売上に対する物価、株の帳簿価格に対する物価など、高い値よりも低い値のほうが好ましい結果を表す場合もあります。

このチャートは 2D でのみ使用することができます。

6.18.1 データマッピング

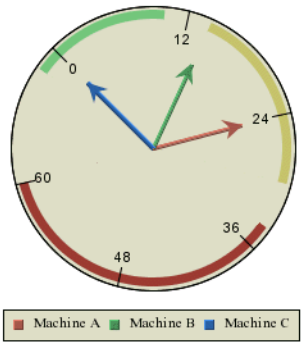
レーダーチャートのマッピングオプション

レーダーチャートのマッピングはつぎのようになります。

- **Data Series（データシリーズ）：**チャートのデータシリーズの数を決定する固有の値を持つデータカラムを選択します。データシリーズの各エレメントは各軸にそって同じ色を使って描かれます。.
- **Category (X)（カテゴリ）：**カテゴリを決定する固有の値を持つデータカラムを選択します。このカラムの独自のエレメントの数がレーダーチャートの軸の数を決定します。
- **Value (Y)（バリュー）：**カテゴリに対して描く数値を供給するデータカラムを選択します。

レーダーチャートにはセカンドバリュー、セカンドシリーズ、コンボオプションは使用できません。レーダーチャートはセカンダリバリューをサポートしていません。

6.19 ダイアルチャート

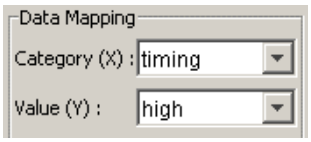


ダイアルチャート

ダイアルチャートは温度計や速度計、時計などに似た円形のチャートを作成するために使用され、ダッシュボードや均衡化されたスコアカードのアプリケーションを作成することができます。データは「ダイアル」上の「針」として表現されます。時計のようにダイアル全体を使用することも、または燃料計のような半円形のチャートとしてダイアルの一部のみを使用することもできます。ダイアルチャートはポップアップラベル、ドリルダウン、および針やダイアルの目盛を回転するといったユーザによる操作をサポートします。

このチャートタイプは 2D でのみ使用することができます。。

6.19.1 データマッピング

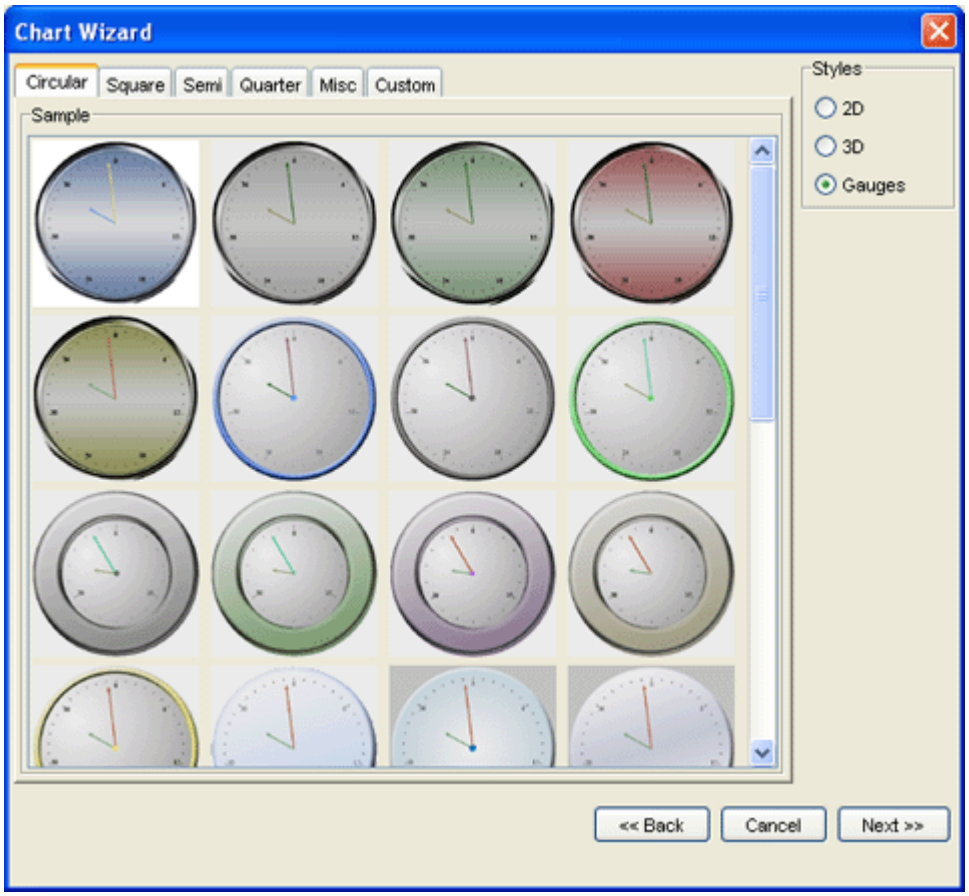


ダイアルチャートのマッピングオプション

ダイアルチャートのデータマッピングはパイチャートと同様です。（詳細はセクション 6.8.1）ただし、パイチャートではくさび（V 字）形だったカテゴリが指針盤の針になります。ダイアルチャートもデータシリーズまたはセカンダリバリューをサポートしません。

6.19.2 ゲージ

ゲージは、ダイアルチャートの特殊な種類で、プロットバックグラウンドイメージまたはプロットフォアグラウンドイメージ、あるいはその両方を表示します。新しいゲージを作成するには、チャートタイプ選択ダイアログで Gauges ラジオボタンを選択します。



ゲージ選択テンプレート

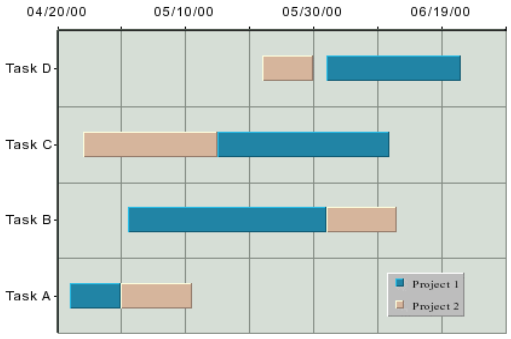
ゲージを迅速に作成するのに役立つように事前に定義されたゲージテンプレートのタブが複数あります。テンプレートを作成し、<ERES Install>/gauges/templates/Custom/フォルダに tpl ファイルを保存することによって、専用のゲージを作成することもできます。さらに、このテンプレートを Custom タブに追加するには、<ERES Install>/gauges/screenshots/selected/Custom/内にあるテンプレートのスクリーンショットと<ERES Install>/gauges/screenshots/unselected/Custom/内にある同一のテンプレートの dimmer バージョンの 2 つのイメージを指定する必要があります。スクリーンショットを簡単に作成するには、テンプレートを正規のスクリーンショット用の gif にエクスポートした後、サイズを 100 ピクセル×100 ピクセルに変更します。その後、dimmer バージョンについて、バックグラウンドを暗い色に変更して、再度エクスポートとサイズ変更を行います。

さらに、既存のダイアルチャートにゲージテンプレートを適用することもできます。現在のチャートがダイアルチャートのときに適用するテンプレートを選択すると、新しいチャートを作成する場合と同様に、ゲージタブが表示されます。

ゲージ適用テンプレート

ダイアルのバックグラウンドおよびフォアグランドイメージの詳細については、ダイアルチャート固有オプションを参照してください。

6.20 ガントチャート



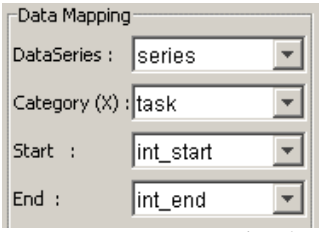
ガントチャート

ガントまたはタイムチャートは計られた時間因子をデータとして表現するのに使われます。これはよくプロジェクトや時間管理などに使われます。ガントチャートは y 軸にカテゴリ、x 軸にバリューを取るバーチャートと似ています。バリュー軸は日付、時間、タイムスタンプ日付（数値の場合もあり）に使います。各データポイントは始まる時間と終る時間があります。

このチャートタイプは 2D でのみ使用できます。

6.20.1 データマッピング

ガントチャートのデータマッピングは High-Low チャートと同様ですが、High と Low の値を選択する代わりに、ここでは始まりと終わりの値を選択します。



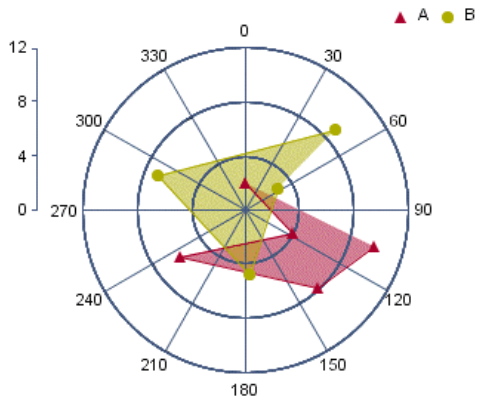
ガントチャートのマッピングオプション

マッピングは次の通りです。

- **Data Series（データシリーズ）：** チャートのデータシリーズの数を決定する固有の値を持つでデータカラムを選択する。
- **Category (X)（カテゴリ）：** カテゴリを決定する固有の値を持つデータカラムを選択する。
- **Start:** カテゴリの時間間隔の始まりを表す値を持つデータカラムを選択します。

- **End:** カテゴリの時間間隔の終わりを表す値を持つデータカラムを選択します。

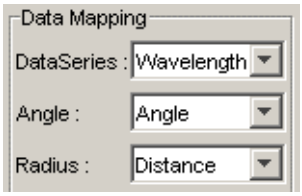
6.21 ポーラチャート



ポーラチャート

2D 分散チャートのように、ポーラチャートは平面上にポイントを描きます。ただし、ポーラーチャートは長方形の座標ではなく、円の中心からの距離 *r* とプロットの中心から放射してポイントを通過する角度を表す *θ*、極座標（*r* ,*θ*）を使って描きます。分散チャート同様、ポーラチャートの座標のデータは数値である必要があります。 *θ* の値は度かラジアンで与えることができます。第 3 のデータシリーズカラムはデータポイントをグループに分けるのに使われます。

6.21.1 データマッピング



ポーラチャートのデータマッピング

ポーラチャートのデータマッピングは分散チャートと同様です。角度と半径オプションで極座標を作る値を持つカラムを選択します。両方とも数値である必要があります。データシリーズボックスではチャートのデータシリーズの数を決定する値を持つデータカラムを選択します。ひとつのデータシリーズ中の各エレメントは、同一のカラーやマーカーなどの描画属性のセットを伴って描画されます。

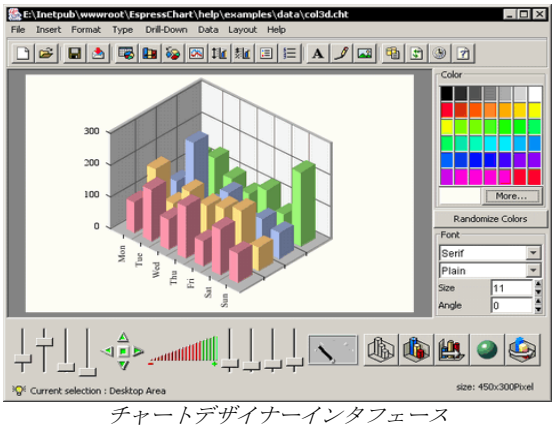
6.22 データマッピングの変更

マッピングオプションの設定が完了したら、メインチャートデザイナインタフェースが表示されます。チャートデザイナの中では、データメニューの'Modify Data

Mapping' を選択することによってデータマッピンウィンドウへ戻ることができます。これでデータマッピング画面に戻りチャートのマッピングを調整することができます。

7 デザイナーインターフェース

ウィザードで全てのステップを完了すると、メインチャートデザイナインタフェースが表示されます。ここではプロパティを変更してチャート表示のカスタマイズや編集を行たり、また新しいエレメントを追加したりすることができます。



チャートデザイナーインタフェース

7.1 デザイナーメニュー

チャートデザイナー機能のほとんどはデザイナーウィンドウの上の方にあるメニューバーでコントロールできます。このセクションでは簡単にどのようなオプションがあるか説明します。機能についてはこの章の後半で説明します。

7.1.1 ファイルメニュー

このメニューは基本的なファイル操作のファイルを開く、閉じる、保存の実行をします。 ファイルオプションの詳細については、第 9 章を参照してください。

New(新規): 新規チャートの作成するデータソースマネージャーを返します。カレントチャートを保存していない場合はプロンプトメッセージが出されます。

Open（開く）: 保存されてたチャート定義を開きます。.cht、 .tpl、.xml チャート定義ファイルがチャートデザイナーによってロードされます。

Close(閉じる): カレントチャートを閉じます。

Apply Template(テンプレートの適用): このオプションはカレントチャートにテンプレートを適用します。.tpl または.xml フォーマットのチャートテンプレートが適用できます。

Save（保存）: カレントチャートを保存します。

Save As（名前をつけて保存）: カレントチャートを保存します。.cht または .tpl（チャートまたはテンプレートファイル）として保存することができます。その

他のオプションに XML チャート定義ファイルの作成やとチャートビューワアプレットを埋め込む HTML の作成があります。

Export（エクスポート）：T チャートを静的イメージフォーマットにエクスポートします。チャートデータは XML フォーマットのテキストにもエクスポートすることができます。

7.1.2 インサートメニュー

このメニューはチャートにいろいろなエレメントを追加します。

Titles(タイトル): このオプションはチャートタイトルを自動的に追加します。メインタイトルは各軸のタイトルと同様に指定することができます。注釈テキストとは違い、タイトルはチャートと共に自動的にサイズや位置が決められます。

Text(テキスト): チャートに注釈テキストを追加することがきます。テキストはチャートのどこにでも追加でき、様々なフォーマットプロパティがあります。ランタイム代用のテキストに変数を追加することができます。チャートへのテキスト追加に関してはセクション 7.6.1 を参照してください。.

Background(背景):背景として使うイメージを選択します。背景イメージは、キャンバスに並べて表示、中央に表示、拡大して表示が選べます。

Link（リンク）：データポイントまたはチャートのデータポイントにハイパーリンクができます。チャートにハイパーリンクを使うには、イメージに加えてマップファイルをエクスポートする必要があります。

Line（線）：チャートに任意/フローティングラインを追加することができます。線はどこにでも入れられ、形を描くのに使うことも可能です。フローティングラインはポインタとしてしばしば用いられ、矢印で描かれます。

TrendLine（トレンドライン）：チャートにトレンドラインを追加することができます。EspressChart は、直線、いかなる次数の多項式、累乗 、指数関数、対数、移動平均、指数関数移動平均（exponential moving average）、三角移動平均（triangular moving average）、立方 B-スプライン（cubic B-spline）、正規分布曲線を含むさまざまなタイプのトレンドラインを引くことができます。

Hort/Vert Line（水平/垂直線）：チャートに固定した水平あるいは垂直線をひくことができます。一定の線（ｘ軸かｙ軸に固定の値で引かれる線）を追加するかまたはコントロールライン（平均、最小、最大、複数標準偏差のある値を基にした線を引く）を追加するかのどちらかを選択します。

Control Area（コントロール領域）:チャートのデータ値に対して比較するための固定領域を（2D チャートのプロットエリアかダイアルチャートの文字面に）描くことができます。

7.1.3 フォーマットメニュー

このメニューは異なるチャートのコンポーネントのプロパティを編集、修正することができます。

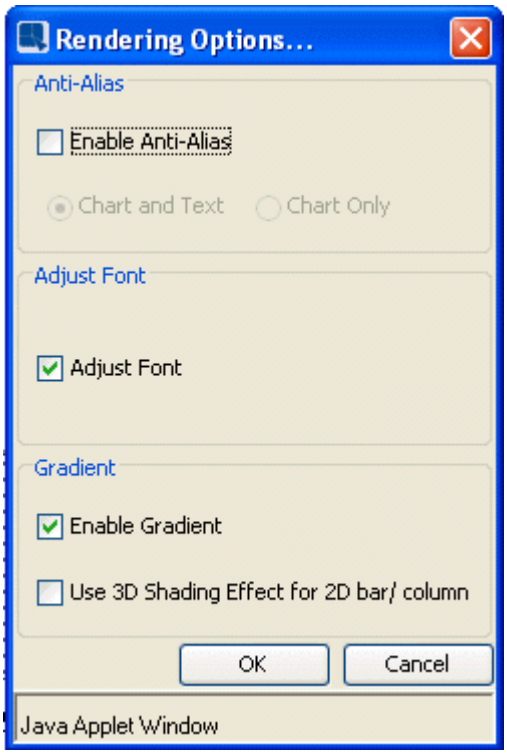
Data Properties(データプロパティ): データをどのように表示するかをいくつかのオプション選択で調節することができます。カラムやバーの太さ、Null データの表示方法、データの頂点にラベルを入れるか入れないか、負の頂点のラベルを色づけするかどうか、およびその色の選択を調節することができます。

Histogram Options（ヒストグラムオプション）: チャートをヒストグラム(度数分布)として描くかどうか、そして頻度数を指定する追加オプションで表示します。

Zoom Options（ズームオプション）:時間のズームを可/不可にし、オプションを設定することができます。このオプションはカテゴリ軸に日付/時間 D データをマッピングしたときのみ適用します。

3D Display Options（3D 表示オプション）: 3D チャートの表示を調節するいくつかのオプションを設定します。3D カラム（または類似）チャートのインラインシリーズを指定することができ、3D 分散、サーフェスチャートのレンダリング（描写）の近似を指定することもできます。（これはたくさんのデータポイントを持つチャートにとってはパフォーマンスの向上になります。）

Rendering Options（レンダリングオプション）: チャートのアンチエイリアスを指定することができます。アンチエイリアスはチャートのテキストと線を滑らかにし、よりはっきり見えるようにします。この機能は、ファイル全体に適用することもできるし、チャートだけに適用してテキストはそのままにしておくこともできます。このダイアログでは、画面の解像度に合わせてチャートのテキストフォントサイズも設定することができます。この機能を使用すると、チャートのさまざまなエクスポートフォーマットへの変換およびインストレーション間での変換をより正確に行うことができます。 このダイアログではさらに、一部の 2D チャート（カラム、バー、スタックカラム、スタックバー、パーセンテージカラムおよびオーバーレイ）に対して 3D シェーディングを有効にすることもできます。また、データポイントおよびバックグラウンドに使用できる勾配を有効にすることもできます。



レンダリングオプションダイアログ

Font Mapping(フォントマッピング): システム（true タイプ）フォントファイルを PDF エクスポートにマップすることができます。この機能の詳細は 9 章セクション 9.2.1 を参照してください。

Chart Options（チャートオプション）: 使用しているチャートタイプにある特定のオプションを実行します。オプションはチャートタイプにより異なります。

Axis Scale（軸目盛）: 値軸の目盛を調節できます。規定値では自動（最適）目盛が使われます。

Axis Elements（軸エレメント）: T 軸と軸ラベルの表現を編集することができます。ここでのオプションは軸の太さ、グリッド線、ラベルステップ、データ形式を含みます。

Canvas（キャンバス）: 背景のキャンバスサイズを調節できます。キャンパスがウィンドウのサイズより大きい場合にスクロールを使用するかしないかを指定することができます。

Legend(凡例): チャート凡例のディスプレイプロパティを編集できます。

Lighting Model（ライティングモデル）: T３D チャートの照明（明暗）オプションの編集ができます。ライトアンビエントカラー（光の周囲のカラー）とそのインテンシティ（強さ）を編集することができます。

Line and Point（線とポイント）: チャートの線ならどれでもディスプレイプロパティを編集することができます。２D チャートではどのデータセットの線でもポイントでもディスプレイするように選択することができ、それらの外観をカスタマイズすることもできます。また、どのトレンド(傾向)線、フローティングライン、水平線、垂直線でも編集するにはこのメニューアイテムを使用することができます。

Plot Area(プロット領域): ２D チャートのｘ軸とｙ軸による境界領域の外観をカストマイズすることができます。

Table(テーブル): チャートデータをディスプレイするテーブルを追加、コンフィグすることができます。

Text Properties（テキストプロパティ）:チャートの中のどのテキストでも比率サイズの変更を設定することができます。このオプションからは Java2D 回転テキストの使用を指定することもできます。このオプションで回転テキストにすっきりとしたきれいな外観になります。また、テキストの置換オプションも指定できます。

Viewer Options（ビューワオプション）: チャートがビューされている場合、チャートビューワアプレットにコンフィギュレーションオプションを指定することができます。ビューワポップメニューでどのオプションが利用可能かコントロールできます。これらのオプションは HTML パラメータでコントロールすることも可能です。

7.1.4 タイプメニュー

このメニューはカレントチャートと次元を変更することができます。それぞれの次元で画像をサポートしているチャートタイプでは、２D と３D 間で変更をすることができます。また、チャートタイプも変更できます。ただし、チャートタイプを変更すると、ある形式の情報を失ってしまうことがありますので気をつけてください。また、ガントチャートとその他のチャートタイプ間では変更することができませんので注意してください。

7.1.5 ドリルダウンメニュー

このメニューオプションはチャートのドリルダウンレイヤーを追加、ナビゲートすることができます。ドリルダウン機能は 8 章で説明しています。ドリルダウンはチャートがレポートから独立してデプロイされている場合のみ有効な機能です。

Add（追加）: データドリルダウンのレイヤーを追加します。

Remove This（カレントの削除）: データドリルダウンのカレントレベルを削除します。

Remove All（全ての削除）: データドリルダウンの全てのレベルを削除します。

Previous（前へ）: データドリルダウンの前のレイヤーへナビゲートします。

Next（次へ）: データドリルダウンの次のレイヤーへナビゲートします。

Go To Top Level（トップレベルへ）: データドリルダウンのチャートのトップレベルへナビゲートします。

Dynamic（ダイナミック）: ダイナミックデータドリルダウンを可能にします。.

Parameter Drill Down（パラメータドリルダウン）: パラメータドリルダウンナビゲーションウィンドウを起動し、パラメータドリルダウンのレイヤーの編集、追加、削除を行うことができます。

7.1.6 データメニュー

このメニューにはチャートデータをリフレッシュ、データをリロード、全データの変更といったオプションがあります。

Modify Data Mapping（データマッピングの編集）: データマッピングウィンドウを立ち上がり、チャートのデータマッピングを変更することができます。

Modify Data Source(データソースの編集):データソースマネージャに戻り、チャートの新しいデータソースを選択することができます。

Modify Database（データベースの編集）: チャートがデータベースをデータソースとして使用している場合、そのチャートが使っているデータベースコネクションを編集することができます。詳細は 5 章セクション 5.2.4 参照してください。

Modify Query（クエリの編集）: チャートがデータベースをデータソースとして使用している場合、このオプションはチャートデータを取ってくるために使用しているクエリを編集することができます。この機能の詳細は 5 章セクション 5.2.４を参照してください。

Query Parameters（クエリパラメータ）: クエリパラメータの再初期化と、カレントチャートのパラメータ値の変更ができます。このオプションはチャートがデータソースとしてパラメータ化したクエリを使用している場合のみ使用できます。

Ordering（順序付け）: チャートのデータポイントに順序付けをし直すことができます。カテゴリエレメントのどれでも順序を変更することができます。または、カテゴリを値によって順序付けることができます.

Refresh（リフレッシュ）: 最新データのカレントチャートを更新します。このオプションにはオリジナルデータソースが利用可能でなければなりません。

Schedule Refresh(スケジュールリフレッシュ): T データをリフレッシュするスケジュールを設定することができます。このオプションはチャートビューワアプレットでチャートをデプロイするためのものです。リフレッシュを行う間隔を設

定することができるので、チャートは自動的に最新データに更新されることになります。

View Table（ビューワテーブル）:カレントチャートが作成されたデータテーブルを表示するウィンドウが開きます。テーブルは最初の 20 レコードを表示します。'Show All Records' チェックボックスをクリックすると全てのレコードを表示するようになります。

View Chart Data（チャートデータのビュー）:データテーブル全体をビューするのではなく、カレントチャートに描かれているデータポイントをビューします。

View Data Source Info（データソース情報のビュー）:チャートを作成するために使用されるデータソースの情報を持つダイアログが表示されます。データソースタイプとロケーションがディスプレイされます。

Go Back（戻る）:リンクで来た場合、オリジナルチャートに戻ります。

7.1.7 レイアウトメニュー

このメニューのオプションではチャートデザイナインタフェースのさまざまなエレメントをトグル切り替えすることができます。このメニューからフォントとカラーパネル、チャートデザインツールバー、３D チャートのナビゲーションパネルのオンとオフを切り替えることができます。

7.2 デザイナツールバー

チャートデザイナウィンドウの上方にあるツールバーは EspressChart のよく使われる機能に簡単にアクセスできるようにしたものです。それぞれのボタンには次のような機能があります。

- 新規チャートの開始
- 既存チャートを開く
- カレントチャートの保存
- カレントチャートのエクスポート
- (チャート)データマッピングの変更
- データディスプレイプロパティの編集
- chart-specific オプションの編集
- 線とポイントの編集
- 軸目盛の変更
- 軸エレメントの編集
- 凡例表示の編集
- データ順序の変更
- 注釈テキストの挿入
- フローティングラインの挿入
- 背景イメージの追加/変更
- チャートデータソースの編集
- チャートデータのリフレッシュ
- データリフレッシュスケジュール

7.3 カラー、パターンおよびフォントパネル

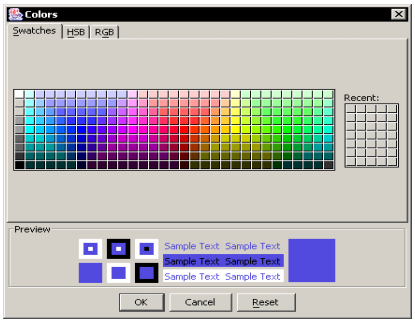
カラーパネルとフォントパネルはチャートデザイナウィンドウの右側にあり、チャーとオブジェクトのカラーを変更したり、チャートのテキストやラベルのフォントサイズやスタイルを変更したりすることができます。レイアウトメニューの'Show font/color panel' オプションのトグル切り替えによってこれらのパネルを表示しないようにすることもできます。

7.3.1 カラーパネル

チャートのエレメントのカラーを変更するのにカラーパネルを使うことができます。エレメントのカラーを変更するにはまずそのエレメントをクリックします。チャートデザイナの下方にあるステータスバーに、どのエレメントを選んでいるか表示されます。エレメントを選んだら、カラーパネルのどれか１色をクリックします。選んだ色がそのエレメントに反映されてカラーが変更します。

カラーパネル

オブジェクトのためにカスタムカラーを作成するには、まずそのオブジェクトを選びます。次に'More' ボタンをクリックします。より大きなパレットが表示され、その中から１色選ぶか、または新たなカラーを作成することができます。

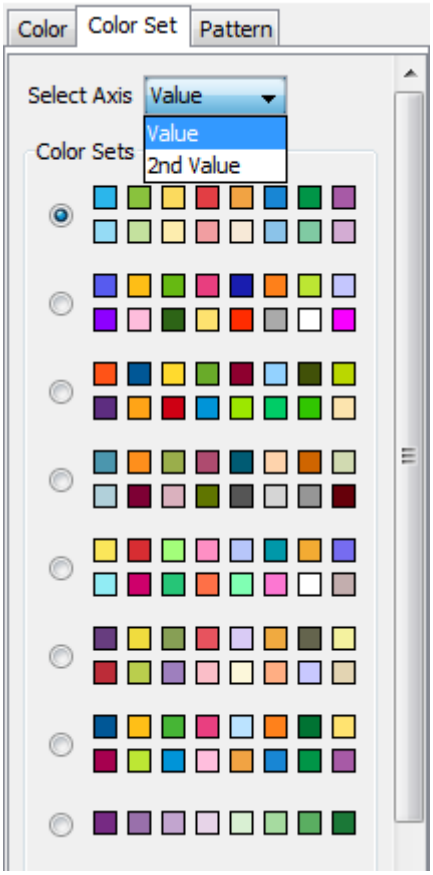


追加カラーダイアログ

このダイアログではスワッチ（見本）から新しいカラーを選ぶかまたは HSB 値、RGB 値を使ってカスタムカラーを作成することができます。'Randomize Colors'ボタンをクリックして、チャートのデータエレメントに新しいカラーのセットを選択することができます。これでチャートの規定値となるカラーを決めることができます。

7.3.2 カラーセットパネル

カラー・セット・パネルで、チャートのカラースキームを設定することが可能になります。チャートの値でのデータポイントまたは２番目値の軸に適用する所定のカラー・セットを含みます。



カラー・セット・パネル

値の、または、２番目値の軸のカラー・セットを変更するには、最初に Color Set タブを選択します。次、「Select Axis」 選択ボックスから軸(Value か 2nd Value)を選択し、該当するカラー・セットのラジオボタンをクリックします。結果としては、選んだカラーセットに従い、チャート・データ・ポイントに色が反映されます。「Select Axis」選択ボックスは、チャートは２番目の値の軸があるだけで表示可能になります。さらに、デフォルトで、値の軸は１番目のカラー・セットを使用し、２番目値の軸は７番目のを使用しています。

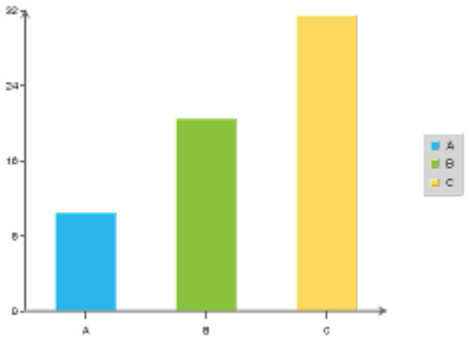
データポイントの色を手動で変更する際に、該当するカラーセットの選択は自動的に外されます。これは、カラーセットがチャートのデータポイントの色に合わなくなったためです。さらに、チャートのデータポイントの色はカラーセットに合わなかったら、自動的に次のカラーセットの最初の色、またはその次の色などを利用します。次

のカラーセットは利用できなかったら、一番目のカラーセットを利用することになります。

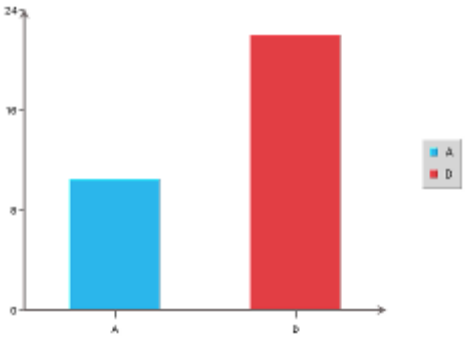
7.3.2.1 6.3.2.1. “Save Colors for Categories”の特徴

データ・ポイントの色は、Save As ダイアログの“Save Colors for Categories”（カテゴリー色の指定）特徴に関連します（Save As ダイアログについては、Section 8.1 - Saving Chartsをご参照ください。）。この特徴が有効になった場合、チャートのデータポイントの色はカテゴリー名（または、チャート・シリーズ）に与えます。そのようなカテゴリー（または、シリーズ）がチャートに表示された場合、該当する色を使用することになります。一方で、この特徴が（デフォルトで）無効になっている場合、データ・ポイントの色はデータ・ポイントの順にカラーセットから取得されます（例：一番目のデータ・ポイントの色はカラーセットの一番目色になり、二番目のデータ・ポイントの色はカラーセットの二番目色になることなどです。）。特徴の有効設定、または無効設定の該当するシナリオを以下の事例に示します：

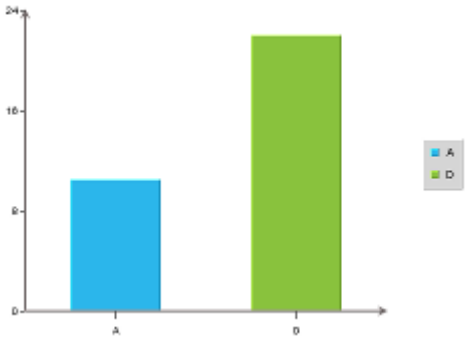
以下の事例で、チャートは三つのカテゴリー(“A”、 “B”、“C”)があり、該当する色を(青，緑，黄，赤)カラーセットから取得します。



イメージ１－ サンプルチャート



イメージ２－ “Save Colors for Categories”特徴が有効の場合



イメージ３ － “Save Colors for Categories”特徴が無効の場合

“Save Colors for Categories”特徴が有効の場合

“Save Colors for Categories”特徴が有効の設定(イメージ１をご参照)でチャートを保存しますので、下記のカテゴリーと色はカテゴリー保存のリストに保存されます：

カテゴリー Ａ ... 青色

カテゴリー Ｂ ... 緑色

カテゴリー Ｃ ... 黄色

チャートとデータ変更をオープンしたら(例：イメージ２のように、データにカテゴリーA と D だけ含みます)、カテゴリーの色は下記のようになります：

カテゴリー Ａ ... 青色(カテゴリー名“A”はカテゴリー保存リストに含むためです。)

カテゴリー Ｄ ... 赤色(カテゴリー名“Ｄ”はカテゴリー保存リストに存在しないため、カテゴリーＤの色はカラーセットの次の利用できる色になります。青色、緑色、黄色がカテゴリーＡ、Ｂ，Ｃに割り当てられているため、カテゴリーＤは赤色になります。

このシナリオで、データポイントの色をカラーセットに対応しないため、カラーセットはカラーセットタブから選択されません。

“Save Colors for Categories”特徴が無効の場合

今回のシナリオは同じ状況ですが、“Save Colors for Categories”の特徴が

無効の設定でチャートを保存します（イメージ３）。今回の保存カテゴリーリストは 空白になります。

チャートをオープンした後、各カテゴリーの色は下記のようになります:

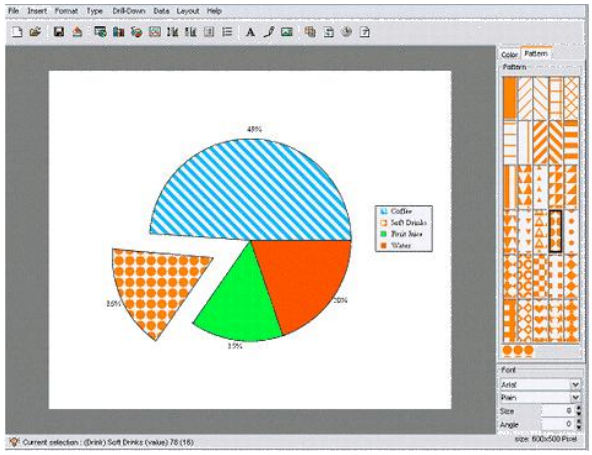
カテゴリーＡ ... 青色（青色はカラーセットの一番目の色のため、カテゴリーＡの色になります。）

カテゴリー Ｄ ... 緑色（カラーセットの最初の色はカテゴリーＡに割り当てられているため、カテゴリーＤの色をカラーセットの二番目色にします。

このシナリオで、データポイントの色はカラーセットに対応されているため、カラーセットをカラーセットタブから選択することになります。

7.3.3 パターンパネル

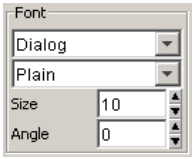
カラーおよびフォントパネルとは異なり、パターンパネルは、データポイントにのみ適用できます。あらかじめ定義されたパターンパレットがあり、これを利用することができます。使用方法はカラーパネルと同様で、まず変更したいデータポイントを選択した後、パターンパレットからパターンを選択する必要があります。パターンは、データポイントに直接適用されます。
データポイントに対してパターンがすでに定義されている場合でも、カラーパネルタブを選択して、カラーパレットから別の色を選択して色を変更することができます。パターンの色が即座に新しい色に変更されます。パターンパレットに表示されるパターンも新しい色に変更されます。



パターンの例

7.3.4 フォントパネル

フォント、フォントスタイル、フォントサイズ、フォントアングルの変更にはフォントパネルを使用することができます。フォントを変更するにはまず変更したいテキストをクリックします。チャートデザインウィンドウの下方のステータスバーにどのオブジェクトを選んでいるのか表示されます。



フォントパネル

最初のドロップダウンボックスではフォントを選びます。２番目のドロップダウンボックスではプレーン、太字、イタリック、太字のイタリック、いずれかのフォントスタイルを選びます。３つ目のドロップダウンボックスではフォントサイズを選びます。最後のボックスではテキストのアングルを指定します。

テキストのグループ（例えば軸ラベルやデータトップラベル）は同じプロパティになります。つまり、グループの中の１つを選択してフォントプロパティを変更するとそのグループ全てに適用されます。

EspressChart はテキストの外観をすっきりさせることができる Java グラフィックライブラリを使用することができます。 普通のテキストはフォーマットメニューの 'Rendering Options' を選択してチャートアンチエリアスを使うことができます。回転テキスト（つまり０度ではないテキスト）はフォーマットメニューの 'Text Properties' を選択することによって、Java２D 回転テキスト機能使うことができます。これらを実行するには Java1.2 以上が必要で、アプレットにチャートをデプロイしてクライアントが Java プラグインを使わない場合はアンチエリアステキストはディスプレイできません。

7.4 ナビゲーションパネル

ナビゲーションパネルは３D チャートに特別なプロパティをコントロールするためのオプションを提供します。このパネルは、２D チャートでは現れません。３D チャートでもレイアウトメニューの'Show Navigation Panel'オプションのトグル切り替えでこのパネルを隠すことができます。

ナビゲーションパネル

ナビゲーションパネルには 6 つのコントロールと 5 つのボタンがあります。左からつステップサイズ、X,Y,Z 軸（それぞれにバー）のライトポジション、ナビゲーション、ズーム、目盛、ナビゲーションスピードの 6 つのコントロールです。ボタンは右から

ワイヤー・フレーム/ソリッド・モードのコントロール、アウトライン挿入のオン/オフ、インラインシリーズのオン/オフ（シリーズ付きカラムナーチャートとバーチャート）、グーローシェーディング、アニメーションのオン/オフとなっています。

Step size（ステップサイズ）：このスライドバーは３Ｄチャートの回転や動きをナビゲーションコントロールで増減を設定することができます。上にすればするほど、ステップの増加が大きくなります。

Light position for X, Y, and Z（x,y,z 軸のライトポジション）：ユーザ定義のライトポジションは照明の位置をコントロールする便利な機能です。つまり、各軸上でその方向から照明があてられるようになります。

Navigation（ナビゲーション）：このコントロールはチャートを回転または translate することができます。真ん中のボタンはトグル切り替えスイッチです。トグルスイッチが押された状態のとき（ボタンが赤く表示されます）は、4 つの三角のボタンをクリックすると、それぞれボタンの向いている方向にチャートが移動します。真ん中のボタン、トグルスイッチが出っ張っているときは、4 つの三角のボタンをクリックすると、それぞれのボタンが向いてる方向にチャートが回転します。ナビゲーションのスピードはナビゲーションスピードコントロールによってコントロールされます。

Zoom（ズーム）：このコントロールはビューワからチャートを近づけたり、遠ざけたりすることができます。操作するには、マウスをコントロールにポイントして、マウスの左ボタンをクリックします。ボタンを押さえている状態でマウスを左へ、あるいは右へドラッグします。マウスを右へドラグッグすると赤い部分が広がり、チャートより近くなります。（チャートへのズームイン）。マウスを左へドラッグすると、赤い部分が左に移動し、チャートが遠ざかっていきます。（チャートへのズームアウト）。

Scale（目盛）：これは 4 つのスライドバーでセットです。最初の 3 つのバーはそれぞれ X,Y,Z 軸への目盛を変更します。4 つ目のバーは 3D カラム、バー、ライン、パイ各チャートの厚みを決定します。（チャートのタイプによります。）バーの位置が上がれば上がるほど値が大きくなります。２D チャートでは、フォーマットメニューの'Data Properties'を選択すると、バー、カラム、スタックバー、High-Low、HCLO 各チャートのバーの幅を調節することができます。

Animation speed（アニメーションスピード）：チャートナビゲーションのスピードを設定することができます。このコントロールはナビゲーションコントロールとアニメーションのオン/オフスイッチとして便利な機能で、チャートの移動と回転のスピードを決定します。操作するには、指針計の針をクリックし、ドラッグで針の位置を変更させます。針を右へ動かすとスピードが早くなり、左へ動かすと遅くなります。

さらに 5 つのボタンが次のような機能を実行します。

 ワイヤー・フレーム/ソリッド・モードのオン/オフ**:** この切り替えスイッチは、スイッチがオンの時は３D チャートをワイヤーフレーム（線のみで３D を表現）でビューしますが、スイッチがオフの時はソリッド（面の集合として３D を表現）オブジェクトとしてビューします。

 **Border Drawing On/Off(アウトライン挿入のオン/オフ):** この切り替えスイッチは、スイッチがオンの時、チャートの各エッジのアウトラインを黒で描きます。

 **Inline Series On/Off（インラインシリーズのオン/オフ）:** この切り替えスイッチは同じ XY 平面のシリーズカラムを描きます。このオプションはシリーズのあるカラムナーチャートとバーチャートにのみ使用可能で、チャートタイプがこの機能に適用しない場合、このオプションはナビゲーションパネル上には現れません。

 **Gouraud Shading On/Off（グーローシェーディングのオン/オフ）:** グーローシェーディングは３D チャートのための洗練されたリアリスティックなシェーディング（描影/明暗法）機能です。スイッチがオンの時、それぞれの表面に影をつけてチャートを描き始めます。

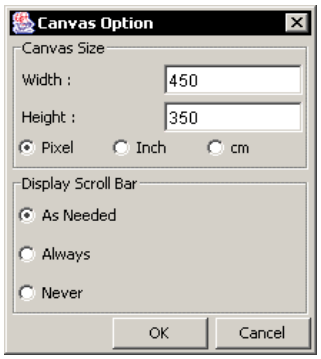
 **Animation On/Off（アニメーションのオン/オフ）:** この切り替えスイッチで３D チャートのアニメーションをスタート/ストップさせることができます。アニメーションのスピードはナビゲーションスピードコントロールを使って設定します。アニメーションを実行中はアニメーションスピードコントロール以外のパネル近とロールは使用不可能となります。

7.5 ビューポート

ビューポートはチャートデザイナーウィンドウの中央の部分を共有しています。ビューポートの中では、キャンバス上のさまざまなチャートエレメントを選択、移動、サイズ変更することができます。

7.5.1 チャートキャンバス

チャートキャンバスはチャートエレメント全てが描かれている背景です。キャンバスの面積/大きさが完成したチャートのサイズになります。フォーマットメニューの'Canvas'を選択して、チャートのサイズを変更することができます。これにより新しいキャンバスの大きさを指定するダイアログが表示されます。



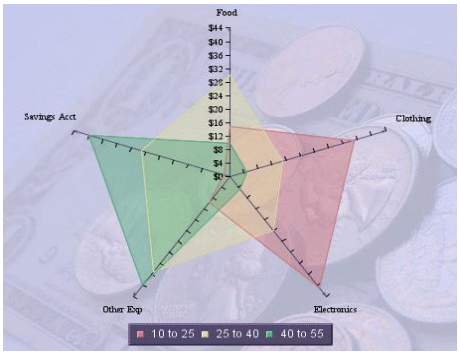
キャンバスフォーマッティングダイアログ

キャンバスサイズはピクセル、インチ、センチメートルで指定できます。このダイアログでは、ビューポートでいつスクロールバーを使用するかを指定することもできます。規定値ではキャンバスがウィンドウより大きいときスクロールバーが表示されます。キャンバスがビューポートウィンドウより小さいときはキャンバスの回りはダークグレイになります。

規定値ではキャンバスはレポートデザイナのチャートに定義したスペースに合わせてサイズを決めます。例えば、レポートで３インチ×4 インチでスペースを設定した場合、次にチャートを編集するときに、キャンバスサイズは自動的に 3 インチ×4 インチになります。

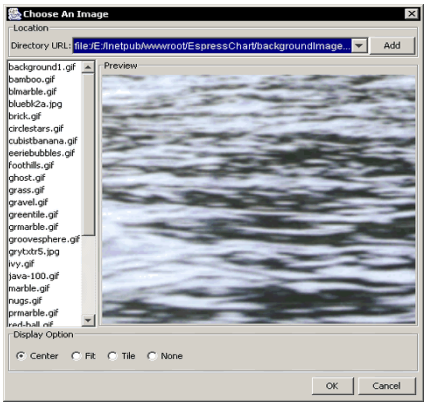
7.5.1.1 背景イメージ

チャートの背景には、シンプルなプレーン、または色付きのキャンバスの代わりにイメージを追加することができます。



背景イメージのレーダーチャート

挿入メニューの'Background'を選択するか、またはツールバーの背景ボタンをクリックして背景イメージを追加することができます。ダイアログが立ち上がり、チャートの背景イメージを指定することができます。



背景イメージの追加ダイアログ

このダイアログでは使いたいイメージを指定することができます。ファイルパス、または URL を使ってファイルを取ってくることができます。また、イメージの表示息は中央に表示、拡大して表示、並べて表示のいづれかを指定することができます。インストレーションの/backgroundimages/ ディレクトリにはサンプル背景イメージが入っています。

*注意 – 背景イメージを追加した場合イメージそれ自身はチャートには保存されません。パス名または URL だけが保存されます。チャートを移動する場合は、指定されているパス名でそのイメージにアクセスできるか確認する必要があります。

特に、チャートファイルをある Web サーバから別の Web サーバに移動する場合は背景イメージも必ず移動先の Web サーバにコピーして、同じ相対位置に置くようにします。例えばマシーンからマシーン BB.\にチャートを移動して、チャートデザイナで設定した URL が http://machineA/ images/test.jpg になっていたとします。EspressChart がマシーン B に保存されたチャートをロードしにいく場合、EspressChart は背景イメージ http://machineB/images/test.jpg をロードしようとします。EspressChart はこれに失敗すると、オリジナルの URL へアクセスしに行きますが、これは Security Exception を発生させます。

7.5.2 チャートエレメントの移動とサイズ合わせ

チャートの中のどのエレメントもクリックで選択することができます。チャートデザイナウィンドウの下方にあるステータスバーが今どのオブジェクトを選択しているのかを示しています。オブジェクと上をクリックしてドラッグすればチャートキャンバス内を移動させることができます。ただし、中には軸やデータトップラベルなどはタンデムで移動するものもあれば、凡例などその他のオブジェクトは単独で移動します。

プロットエリアでマウスをクリックしてドラッグするとチャート全体が移動できます。これによって、キャンバス内をチャート全体が移動させられます。チャートをリサイズ（サイズの変更）するにはプロットエリア内で右クリックしてドラッグします。こ

れによりプロットが拡大したり、縮小したりします。ナビゲーションパネルのズームコントロールを使えば、３D チャートのサイズあわせもできます。

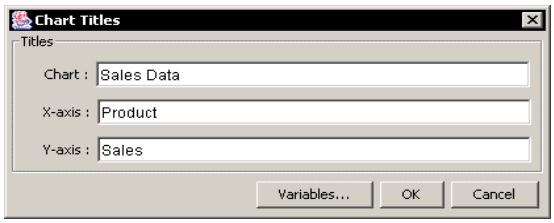
7.6 チャートエレメントの追加

EspressChart では規定値のチャートエレメントに加え、チャートに追加できるいくつかの追加エレメントを提供しています。

7.6.1 テキストの追加

チャートへテキストを追加するには 2 通りの方法があります。テキストはタイトルとして追加されるかまたは普通のテキストエレメントとして追加されるかです。

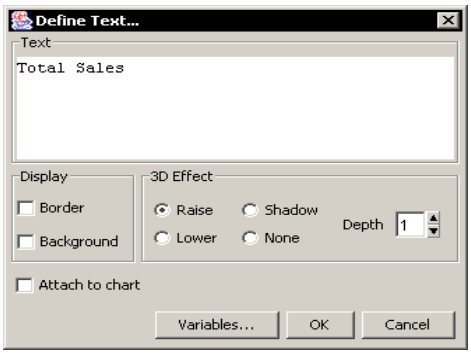
タイトルの追加: チャートにタイトルを追加するには挿入（インサート）メニューから'Titles' を選びます。これによってチャートにタイトルを入れるダイアログが立ち上がります。



タイトル追加ダイアログ

ダイアログはチャートにメインタイトル、各軸にタイトルを指定できます。軸のないパイ、ダイアルチャートではチャートタイトルだけ指定するようになります。タイトルをしてしたら、'OK'をクリックしそれらはチャートに追加されます。タイトルは自動的に配置され大きさも調節されますが、移動したりフォントを変えたりできます。

テキストの追加: テキストフィールドをチャートに追加するには、挿入（インサート）メニューから'Text' を選ぶか、ツールバーのテキスト追加ボタンをクリックします。これによってチャートにテキストを追加することができるダイアログが立ち上がります。



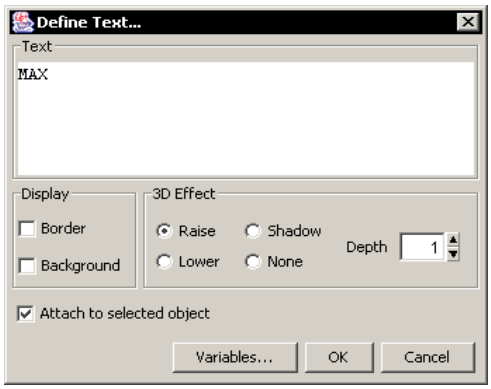
テキスト追加ダイアログ

このダイアログでテキストを指定し、ディスプレイオプションを設定することができます。テキストの周りの境界線や背景を描くかどうかを指定できます。またどんな効果を背景に適用するかも指定することができます。

テキストを指定し終えたら、'OK'をクリックします。これでチャートの中にテキストを配置することができます。チャートキャンバスの中ではポインタに小さい四角が付いています。テキストを配置したいところでマウスをクリックします。

注釈テキスト: EspressChart は注釈もサポートしています。注釈はラベルやテキストフィールドを線やチャートプロットのような特別なエレメントに添付することができます。例えば最大値を示すチャートにコントロールラインを挿入して、"MAX" というテキストラベルをこのコントロールラインに添付することができます。毎回最大値が変更するとそのラベルもコントロールラインと一緒にその位置も調節します。コントロールラインに関する詳細はセクション 7.6.2.3 を参照してください。

テキストを注釈で指定するには 2 通りの方法があります。テキストをチャートプロットに添付するには、テキストを追加するときに"Attach to Chart" を選択します。テキストをコントロールラインのような特定のオブジェクトに添付するには、まずオブジェクトを選択し、挿入（インサート）メニューから'Text'を選択するか、ツールバーのテキスト追加ボタンをクリックします。"Attach to selected object" のオプションは自動的にチェックされます。チェックが終わると追加したテキストが自動的にオブジェクトに添付されます。



テキスト（注釈）の追加ダイアログ

7.6.1.1 テキスト変数

EspressChart ではチャートのあるバリュー/オブジェクトを基にしたランタイム置換を可能にする変数をテキストの中に指定することができます。例えば、チャートがデータソースとしてパラメータクエリを使っている場合、ランタイム時にどのパラメータ値が選択されたのか表示する"&paramInfo" という変数を使うことができます。

タイトル挿入ダイアログもテキスト追加ダイアログもどちらも下方に'Variables' と書かれたボタンがあります。このボタンをクリックすると、使用できる変数のリストを表示され、このダイアログでテキストやタイトルに追加する変数をひとつ選びます。

変数ダイアログ

次の変数がサポートされています。

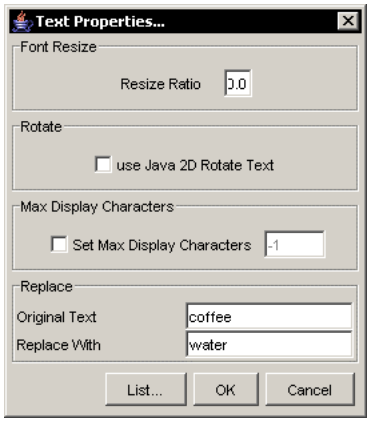
- "*&drillInfo*": どのデータポイントがドリルダウンチャートでドリルされているかを表示します。この変数はパラメータドリルダウンには使用できません。
- "*&paramInfo*": 選択されたパラメータ値を表示します。ドリルダウンチャートの&drillInfo の代わりに使用できます。
- "*&date*": 最後にチャートが描かれた日付を表示します。
- "*&time*": 最後にチャートが描かれた時間を表示します。

- "*&category*": カテゴリカラムの名前を表示します。
- "*&series*": データシリーズカラムの名前を表示します。
- "*&sumby*": 総計（sum-by）カラムの名前を表示します。
- "*&value*": バリューカラムの名前を表示します。
- "*&subvalue*": セカンドバリューカラムの名前を表示します。
- "*&xaxis*": x 軸にマッピングされているカラムの名前を表示します。これは分散チャートやバブルチャートのように x 軸にカテゴリではなくバリューをマップングしたチャートに使います。
- "*&yaxis*": y 軸にマッピングされているカラムの名前を表示します。これは分散チャート、バブルチャート、サーフェスチャートなどに使います。
- "*&zaxis*": z 軸にマッピングされているカラムの名前を表示します。これは分散チャート、バブルチャート、サーフェスチャートなどに使います。
- "*&paramInfoName<index>*": チャートにパラメータが含まれている場合、指定したインデックス*<index>番号*のパラメータ名を表示します。
- "*&paramInfoValue<index>*": この名前のパラメータがチャートに含まれている場合、そのパラメータに対して選択された値を表示します。

7.6.1.2 テキスト置換

EspressChart ではチャートの中の特定のテキストをオーバーライトすることができます。データソースが認識できるカラム名を使用していない場合、便利な機能です。この機能はテキストの全てを置換します。例えば、シリーズのない x 軸ラベルと凡例のアイテムの両方のカラム名を表示するカラムチャートの場合、ラベルだけテキスト置換を行い、凡例アイテムは置換しないということはできません。

テキスト置換を使用するにはフォーマット（Format）メニューから'Text Properties'を選択します。これにより置換するテキストを指定するダイアログが立ち上がります。



テキストプロパティダイアログ

同じテキストに連続的に変更を加える場合、オリジナルのテキストを使用します。例えば、"coffee" を"water"に置換したとします。ここで"water"を"soft drink" に置換したい場合、テキスト置換ではオリジナルテキスト、つまり"coffee"を "soft drink"に変換

します。テキスト置換を削除するには単純にオリジナルの文字列をそれ自身に変換します。つまり、先の例で言うと、"coffee" を"coffee" に置換します。

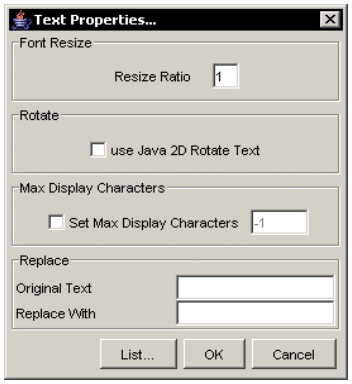
下方の'List'ボタンをクリックするとオリジナルテキストのリストと置換するテキストを見ることができます。これによりダイアログが表示され、チャートのテキスト置換全てのリストが表示されます。

置換テキストのリスト

このダイアログでは、ダイアログの下方のボタンをクリックすることによって、どの置換テキストでも選択（Chose）でき、修正（Modify）、元に戻す（Undo）こともできます。

7.6.1.3 テキスト自動サイズ変更

EspressChart では、チャートのテキストのフォントサイズを調整して、自動的にサイズを変更することがことができます。この機能は同じチャートテンプレートを使って、いくつかの異なるサイズのチャートを作成するのに便利です。チャートキャンバスの変更により調節するようにフォントサイズの変更比率を指定することができます。サイズ変更比率（Re-size Ratio）を指定するにはフォーマット（Format）メニューから'Text Properties' を選択します。



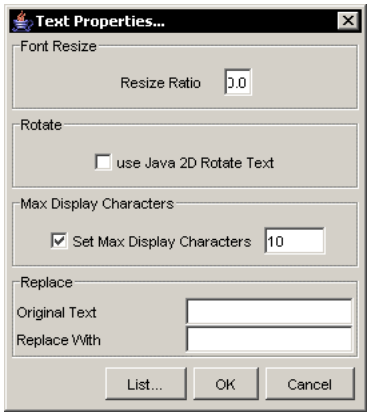
テキストプロパティダイアログ

比率（Rtio）はフォントがキャンバスに対して変更されるサイズの相対割合を指示しています。例えば、チャートが 500 x 500 ピクセルから 250 x 250 に変更されたとします。サイズ変更比率が 1 の場合、12 ポイントのフォントは、キャンバスの縮小と

同じ割合で縮小させて 6 ポイントになります。しかし、サイズ変更比率が 0.5 だった場合は縮小はキャンバスの縮小率の半分となり、１２ポイントが 9 ポイントになります。

7.6.1.4 テキスト切り詰め

チャートに長いラベルやテキストがあると、チャートプロットにスペースが空きすぎたり、実際のチャートに十分な領域がとれなかったりすることがあります。このような場合、EspressChart はチャートテキストのテキスト切り詰め機能（Cropping）を提供します。テキストがユーザ指定の限界値より長い場合、 "..."で切り詰められます。チャートのヒントボックスがラベルの全文字列を表示します。限界値を指定するにはフォーマット（Format）メニューの'Text Properties' を選択します。



Text Properties Dialog

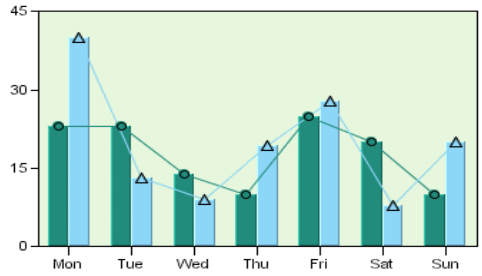
テキスト切り詰めを有効にするには、'Set Max Display Characters' オプションにチェックを入れ最大文字数を指定します。ここで指定した文字数より長いテキストは切り詰められます。

7.6.2 線の追加

EspressChart ではさまざまなタイプの線をチャートに追加、フォーマットすることができます。

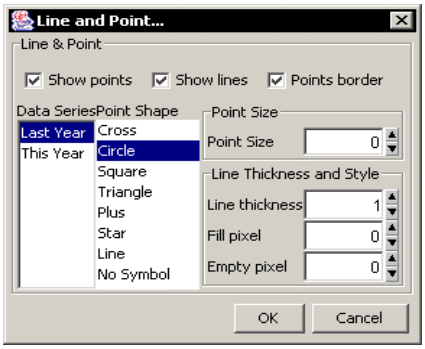
7.6.2.1 線と点のフォーマット

２D チャートではどんなチャートタイプでもデータポイントの線と点を表示する選択ができます。いくつかのチャートタイプは既にこの表現を使っています。（ライン、分散チャート）



線と点の表示されたカラムチャート

線と点の表示はフォーマット（Format）メニューから'Line and Point'を選択するか、またはツールバーの'Line and Point'ボタンをクリックすることによってコントロールされます。これによりいくつかのオプションを表示するダイアログが立ち上がります。



線と点ダイアログ

最初の 3 つのオプションは線、点、ポイントの境界線を表示するかどうか指定します。レーダー、分散、ポーラの各チャートではエリアの表示選択もあります。レーダーチャートとポーラチャートではエリアオプションはデータポイントで囲まれているエリアを塗りつぶします。分散チャートでは x 軸からデータポイントへカラムを描きます。残りのオプションでは、データシリーズの各エレメントのために線と点をカスタマイズすることができます。

各データシリーズエレメントでは点の形を指定することができます。また、点の大きさもコントロールできます。規定値では点の大きさはゼロ。点の大きさは-1, -2, -3 と指定でき、それぞれ 0.75, 0.5, 0.25 という大きさになります。-3 (0.25)では、点は選んだ形にかかわらずドットで描かれます。

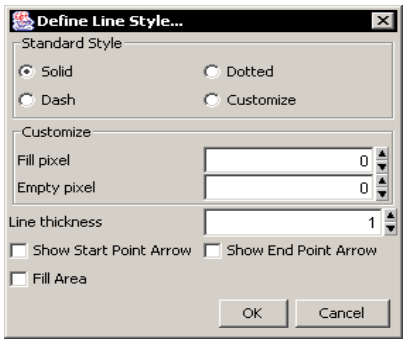
線には太さを指定することができ、またダッシュパターンもカスタマイズできます。ダッシュパターンは、塗りつぶされたピクセルの数と空白のピクセルの数（0 と 255 の間）を指定することで作成されます。線はセグメントに分けることによって描かれます。-塗りつぶされたピクセルの数の次に空白なピクセルの数を指定します。両方とも 0 で設定すると実線です。両方 255 は線が引かれません。

7.6.2.2 フローティングライン

フローティングラインはフリーフォームの線で、チャートキャンバス上のどこにでも自由に追加できます。フローティングラインはチャートの特定のエレメントを指し示すのにしばしば使用されます。フローティングラインを追加するには挿入（Insert）メニューから'Line'を選択するか、ツールバーの'Insert Line'をクリックします。

このオプションを選ぶとマウスポインターが十字に変わります。チャートキャンバス内の線を開始したい地点でクリックします。クリックするごとに線のポイントが追加され、セグメントができます。このようにしてフローティングラインを使って好きな形を描くことができます。描き終えたら、右クリックで終了させます。線が追加されます。

一度キャンバスに線が引かれたら、それを個別に移動することはできません。注釈テキストのようにキャンバスと一緒に動くようになります。フローティングラインへオプション指定するにはまずラインを選択し、フォーマット（Format）メニューから'Line and Point' をクリックするか、ツールバーの'Line and Point'ボタンをクリックします。これによりダイアログが立ち上がり、フローティングラインに対してさまざまなプロパティをフォーマットすることができます。



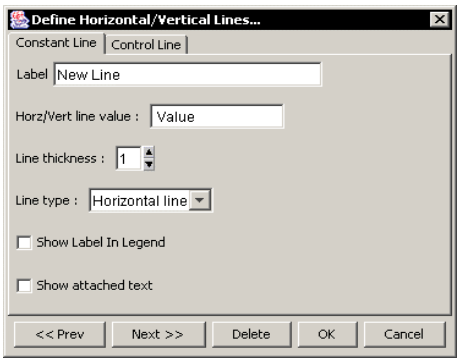
フローティングラインの線と点のダイアログ

ダイアログでは標準ラインスタイルの指定、または線と点フォーマットと同じ要領でダッシュパターンにカスタマイズすることができます。また、ピクセルで線の太さも指定できます。ダイアログの一番下には線の開始と終わりに矢印をつけることができるチェックボックスがあり、同様に実線で隙間なく囲われた領域も塗りつぶすことができます。

7.6.2.3 固定水平垂直線

固定水平や垂直の線はチャート軸上に描かれます。これらの線は平面上の３D チャートにも引くことができます。固定線にはコンスタントラインとコントロールラインの２つタイプがあります。コンスタントラインは軸のある値に固定された線です。コントロールラインは、ある値の外側にあるデータポイントを見分けることができるようにするための線で、計算された値を基に描かれます。どちらのタイプの線もチャート

に追加するには挿入（Insert）メニューから'Hort\Vert Line' を選択します。これにより線のオプションをしていするダイアログが表示されます。



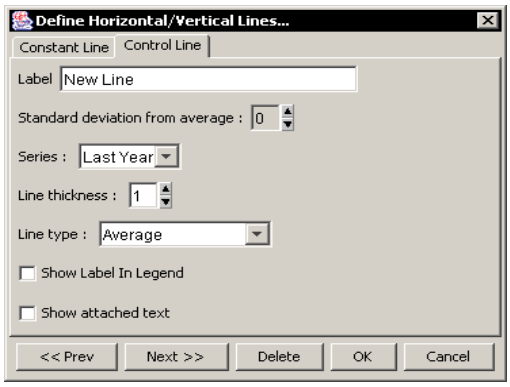
コンスタントラインダイアログ

コンスタントラインには線に使われる数字の値と同様にラベルを指定する必要があります。カテゴリ軸ではデータポイントが 0.5 で始まるようになっています。線の太さ、線は水平方向か垂直方向かも指定します。最後の２つオプションはこの線のためのアイテムを凡例に追加するか、線に対する注釈を表示するかどうかを指定します。

レーダーチャートの場合、水平/垂直オプションは無効です。レーダーチャートはすべてのチャート軸の同一点にコンスタント/コントロールラインを描画します（レーダーグリッドと同様の方法です）。さらに、レーダーチャートの場合、”Circular Style"という名前の追加オプションが表示されます。デフォルトでは、レーダーチャートのラインはセグメント方式で（各軸上の点を直線で結んで）描画されます。ただし、このオプションを選択すると、コンスタント/コントロールラインは円として描画されます。

*注意 – 線に対して注釈を指定していない場合は、最後のプションを選択しても何も表示されません。チャートへの注釈の追加に関する詳細はセクション 7.6.1 を参照してください。

コントロールラインを追加するにはダイアログの"Control Line" タブをクリックします。



コントロールラインダイアログ

コントロールラインはどのエレメントで値を計算して欲しいか、値をどう計算するかを指定する必要があります。（シリーズがない場合、このオプションは表示されません。コントロールラインに対するオプションは平均（Avarage）、最小値（Minimum）、最大値（Maxmum）、標準偏差の倍数（Multiples of standard deviation）です。

全てのオプションを指定すると、線はチャートに追加されます。ひとつ前のダイアログでプロパティを変更するには、もう一度挿入（Insert）メニューから'Hort\Vert Line'を選択し、'Prev' と 'Next'ボタンを使って、必要な線を見つけます。変更したい線の上をダブルクリックしてもできます。

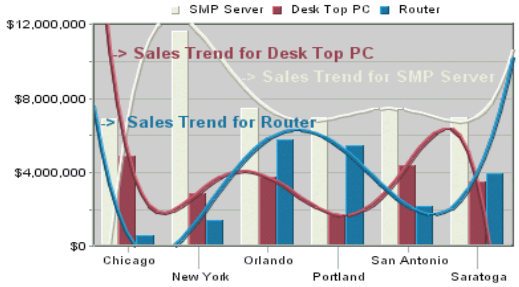
最初に線を選択しておいて、フォーマット（Format）メニューから'Line and Point'を選択しても、ツールバーの'Line and Point'アイコンをクリックしても、固定ラインの表現をコントロールすることができます。これによって線をカスタマイズできるダイアログが立ち上がります。

固定ラインの線と点ダイアログ

このダイアログでは標準ラインスタイルを指定、または線と点フォーマットと同じ要領でダッシュパターンにカスタマイズすることができます。

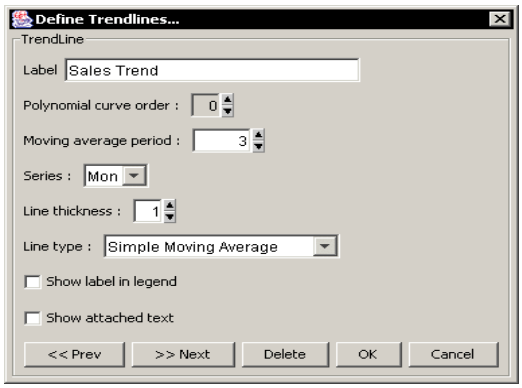
7.6.2.4 トレンドライン

EspressChart の効果的な機能の一つはトレンドラインを入れられということです。トレンドラインはある傾向を明らかにし、強調することによってチャートのデータのより詳細を見せる手助けをします。



トレンドライン付きチャート

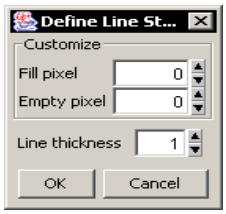
チャートにトレンドラインを追加するには、インサート（Insert）メニューから 'Trendline' を選択します。これによりトレンドラインのオプションを指定できるダイアログが立ち上がります。



トレンドラインダイアログ

このダイアログでは、データシリーズのどのエレメントを計算の基にするかわかりやすいように線のラベルを指定します。EspressChart はトレンドラインの次のタイプをサポートしています。直線、多項式、 累乗、指数関数、対数、 移動平均、指数関数移動平均、三角移動平均、3 次 B スプライン、正規分布曲線。移動平均では平均期間、多項式ではカーブオーダー（curve order）を指定する必要があります。

トレンドラインのオプションを指定し終わったら OK をクリックします。これでトレンドラインがチャートに追加されます。トレンドラインの外観を変更するには、まずトレンドラインを選択しておいて、フォーマット（Format）メニューから'Line and Point'を選ぶか、ツールバーの'Line and Point'ボタンをクリックすることによって行えます。これによってラインをカスタマイズするダイアログが立ち上がります。

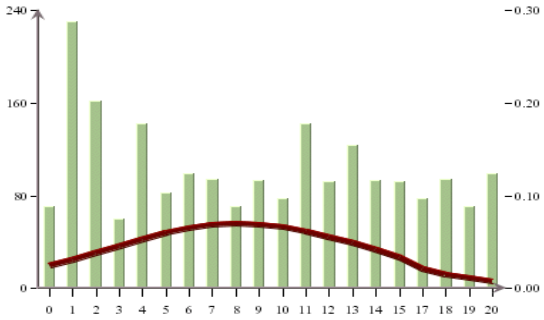


トレンドラインの線と点ダイアログ

このダイアログで、線と点フォーマットと同じ要領でカスタムダッシュパターンを作成することができます。

7.6.2.4.1 正規分布曲線

トレンドラインの特別なタイプはチャートのデータに正規分布曲線を描くことができます。正規分布曲線を描くためには、チャートが２D カラムチャートまたはバーチャートでなくてはならず、データシリーズがなくカテゴリは数値でなければなりません。これらの条件が揃っていれば、トレンドオプションのひとつとして正規分布曲線を指定することができます。

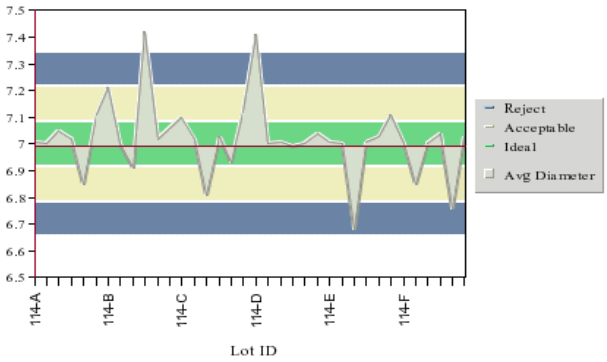


正規分布曲線付きチャート

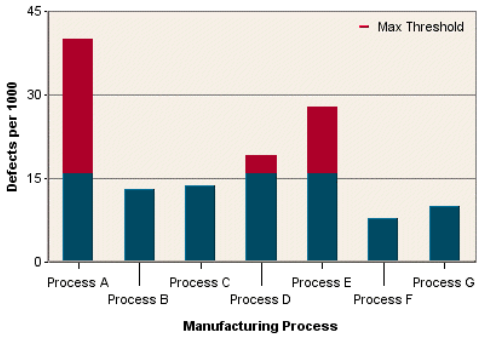
通常曲線の目盛は値軸の目盛と違っているので、曲線はセカンダリ軸で表されます。セカンダリ軸の目盛の変更によって目盛を修正できます。

7.6.3 コントロールエリアの追加

コントロールエリアはデータのある範囲に対してチャートデータを比較するのにとても便利です。ほとんどの２D チャートでは、コントロールエリアは値軸上の値の範囲を塗りつぶしたエリアとしてチャートプロット上に描かれます。データポイントはコントロールエリアを越えて描かれ、どのデータポイントが特定範囲内に入っているかひと目でわかります。

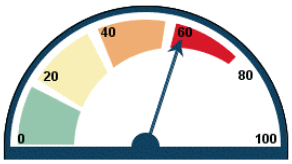


コントロールエリア付き 2D アリアチャート



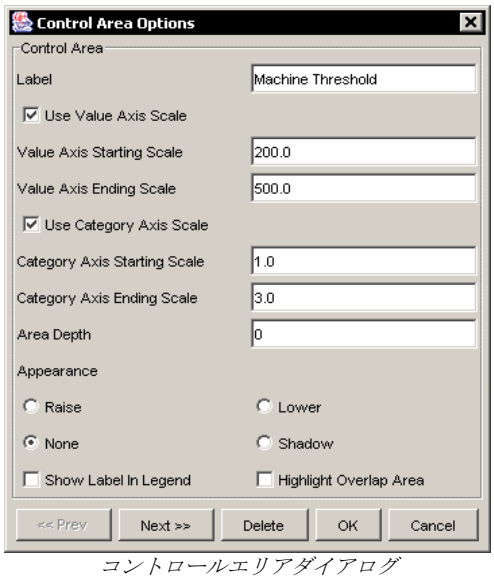
データポイントに描かれたコントロールエリア付きからむチャート

コントロールエリアの特例はダイアルチャートに使用されます。ダイアルチャートのコントロールエリアはダイアルの文字面に弧で描かれていて、ダイアルの針がその範囲ないに入ってるかどうかがわかります。コントロールエリアはレーダーチャートとパイチャートでは使用できません。



コントロールエリア付きダイアルチャート

コントロールエリアをチャートに追加するには、インサート（Insert）メニューから'Control Area'選択します。ダイアルチャートでない場合は次のようなダイアログが表示され、コントロールエリアのオプションを指定することができます。



コントロールエリアダイアログ

コントロールエリアには次のオプションがあります。

Label（ラベル）：コントロールエリアに対するラベルを指定します。

Use Value Axis Scale（値軸目盛の使用）：コントロールエリアがチャートの値軸上の値に挟まれるかどうか指定します。

Value Axis Starting Scale（値軸上の開始点）：値軸上のコントロールエリアの低い方の境界線の値を指定します。

Value Axis Ending Scale（値軸上の終了点）：値軸上のコントロールエリアの高い方の境界線の値を指定します。

Use Category Axis（カテゴリ軸の使用）：コントロールエリアが、チャートのカテゴリ軸上の値に挟まれるかどうか指定します。

Category Axis Starting Scale（カテゴリ軸上の開始点）：カテゴリ軸上の低い方の境界線の値を指定します。

Category Axis Ending Scale（カテゴリ軸上の終了点）：カテゴリ軸上の高い方の境界線の値を指定します。

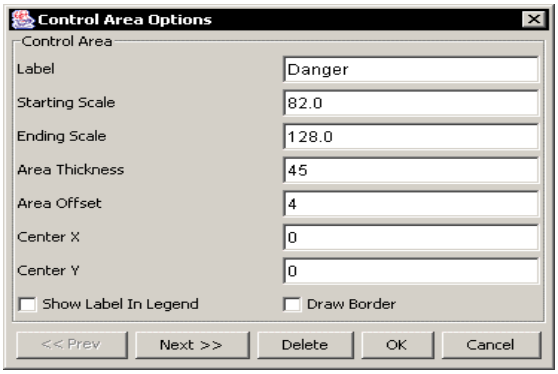
Area Depth（エリアの深さ）：外観スタイルの深さを指定します。スタイルを選択していない場合は、この深さはまったく影響しません。

Appearance（外観）：コントロールエリアに３D またはシャドー（影）効果を指定することができます。エリアの深さが指定された場合、外観が 0 というのはなにも作用しません。

Show Label In Legend（凡例にラベルを表示）：チャートの凡例にコントロールエリアのラベルを表示するかどうか指定します。

Highlight Overlap Area（オーバーラップエリアの強調）：データポイントがコンロトールエリアの境界線をオーバーラップ（はみ出）した部分のエリアだけを強調して表します。

チャートがダイアルチャートの場合、挿入（Insert）チャートメニューから'Control Area'を選択すると、異なるダイアログが表示されます。



ダイアルチャートのコントロールエリアダイアログ

ダイアルチャートのコントロールエリアには次のオプションがあります。

Label（ラベル）：コントロールエリアに対するラベルを指定します。

Starting Scale（開始の目盛）：コントロールエリア開始の値を指定します。

Ending Scale（終了の目盛）：コントロールエリア終了の値を指定します。

Area Thickness(エリアの太さ): コントロールエリアの太さを設定します。

Area Offset（エリアオフセット）：ダイアルチャートのエッジ（端）からのオフセット（補正値）をピクセルで指定します。

Center X（センターX）：コントロールエリアの中心の X 座標を設定します。0 は文字面の中心と同じ位置を意味します。ダイアルの中心からのオフセット位置をピクセル単位の数値（マイナス、プラスどちらでも）を指定します。

Center Y（センターY）：コントロールエリアの中心の Y 座標を設定します。指定方法は上のオプションと同じ。

Show Label in Legend（凡例でラベル表示）：チャートの凡例にコントロールエリアのラベルを表示するかどうか指定します。

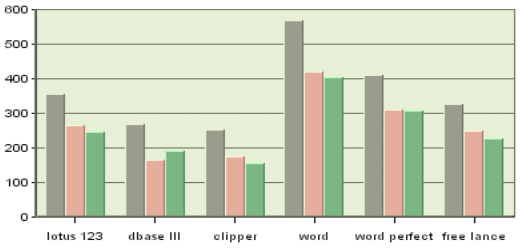
Draw Border（境界線の描画）：コントロールエリアの回りに境界線を描くかどうか指定します。

コントロールエリアのオプションの指定を完了すると、チャートにコントロールエリアがチャートに追加されます。挿入(Insert)メニューから'Control Area'を選択、また

はチャートのコントロールエリアをダブルクリックすることによってコントロールエリアの編集ができます。

7.6.4 テーブルの追加

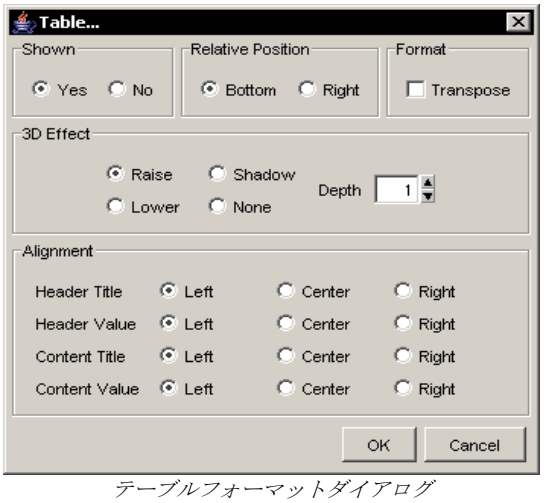
EspressChart はチャートの表示だけではなく、チャートのデータポイントの示すテーブルを表示することができます。テーブルはチャートの下または右に配置されます。



PERIOD_END	lotus 123	dbase III	clipper	word	word perfect	free lance
1987-03-31	355	270	253	569	410	327
1987-02-28	267	165	177	420	311	251
1987-01-31	248	193	155	405	307	229

テーブル付きチャート

チャートにテーブルを追加もしくはテーブルの表示オプションを編集するにはフォーマット（Format）メニューから'Table' を選択します。これによりテーブル表示のためのさまざまなオプションをカスタマイズするダイアログが立ち上がります。



テーブルフォーマットダイアログ

このダイアログではテーブルを表示するか否か、またテーブルの相対的な配置はチャートプロットの下方か右かも指定します。テーブルに対する３D 効果とその深さも指定します。

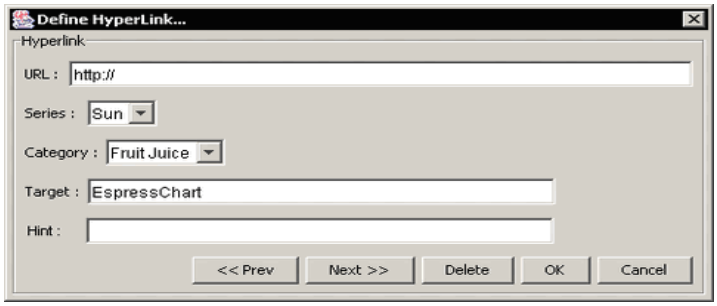
'Transpose' チェックボックスはテーブルのカラムと行をスワップさせることができます。規定値では、カテゴリエレメントはカラムとして描かれ、データシリーズエレメントは行として描かれます。

'Alignment' オプションはテーブルセルの中のテキストの水平方向のアラインメント（右ぞろえ、左ぞろえ、またはセンタリング）を指定します。アラインメントはテーブルセルの中と同様、行とカラムのヘッダーにも設定されます。

*注意 – チャートキャンバスに十分なスペースがない場合は、すべてのデータポイントがテーブルに表示されるということになりません。テーブルサイズはキャンバスサイズによって調整されます。また、テーブルセル中のテキストのフォントサイズによっても調整されます。

7.6.5 ハイパーリンクの追加

EspressChart はチャートのデータポイントにハイパーリンクを追加することができます。リンクは１データポイントでも複数エレメントでも設定することができます。追加されたハイパーリンクはすべて、チャート内のデータポイントと凡例上のデータポイントのそれぞれのフィールドの両方に適用されます。チャートにハイパーリンクを追加するには挿入（Insert）メニューから'Link'を選択するか、またはデータポイント上を右クリックして。、ポップアップメニューから'Insert Link'を選択します。



リンク挿入ダイアログ

URL フィールドにリンクしたい Web ページを指定します。

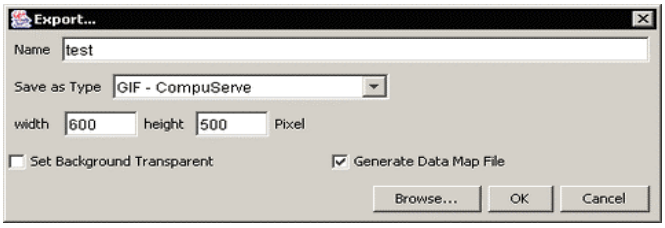
'Series'と'Category'のドロップダウンメニューの使って、データシリーズのエレメントを選択し、ハイパーリンクのためのカテゴリエレメントを選択します。全てのデータシリーズエレメントや全てのカテゴリエレメントにリンクを張ることもでもできます。

データポイントやデータシリーズにリンクを張るとき、HTML で認識される'Target'パラメータをしていすることができます。これは新たな HTML のページを開くのに新しいブラウザウィンドウを開いて表示するか、同じブラウザウィンドウで表示するか、また、カレントページのある部分と同じところを使って新しいページを表示するのかを決定するのに使われます。

'Hint' フィールドは、マウスがデータポイントに置かれたときにポップアップするテキストを入力します。ハイパーリンクなしでポップアップラベルを作成したい場合には URL フィールドを空白にして Hint だけ指定することができます。

7.6.5.1 ハイパーリンクのビュー

チャートにハイパーリンクを指定した場合、チャートを HTML または DHTML フォーマットにエクスポートする場合のみハイパーリンクがアクティブになります。チャートをエクスポートするとき、リンクに関する情報が入ったイメージマップファイルが自動的に生成されます。



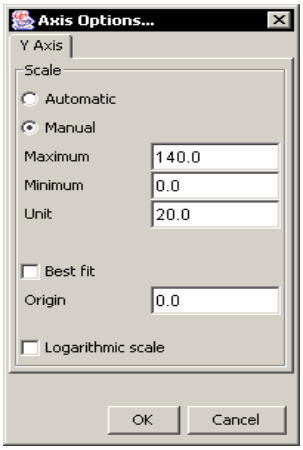
エクスポートダイアログ

7.7 チャート軸のフォーマット

EspressChart はチャート軸に対して幅広いフォーマッティングが行えるようになっています。ユーザは軸目盛から軸ラベルの表示方法に至るまで全てカスタマイズすることができます。

7.7.1 軸目盛

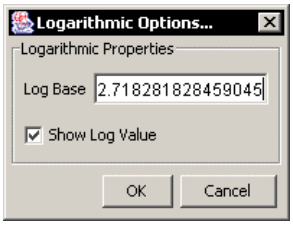
EspressChart の規定値ではチャートに描かれるデータにとって最適な大きさになるように計算して値軸の目盛をとります。これは、表示されるデータが完全に変更されてしまった場合にとても便利な機能です。しかし、手動で軸目盛を設定したいことも実際には多いでしょう。軸目盛を変更するには、フォーマット（Format）メニューから'Axis Scale' を選択するか、またはツールバーの'Axis Scale' ボタンをクリックします。これによりチャートの値軸の目盛をフォーマットするダイアログが立ち上がります。



軸目盛ダイアログ

オプションは次の通りです。

- **Automatic（自動）：**軸の自動スケーリング（目盛をとる）をオンにします。このオプションが規定値になっています。
- **Manual（手動）：**手動スケーリング（目盛をとる）をオンにして軸目盛にオプションを設定します。
- **Maximum（最大値）：**軸目盛の最大値を指定します。
- **Minimum（最小値）：**軸目盛の最小値を指定します。
- **Add Padding（パディングの追加）：データの最大値とチャートの上端の間に余白【訳注：要チェック】を設けるために、軸の最大値を増加します。**
- **Best Fit（最適化）：**データの最小値と最大値を基にチャートの原点を自動的に配置します。
- **Origin（原点）：**X 軸と Y 軸の交差する点を指定します。通常は 0 です。
- **Logarithmic（対数）：**割り当てられた軸に対数の目盛を設定します。これは該当する軸が正の値のみをとるとき有効です。このオプションを選択すると対数オプションを設定する追加ダイアログが立ち上がります。

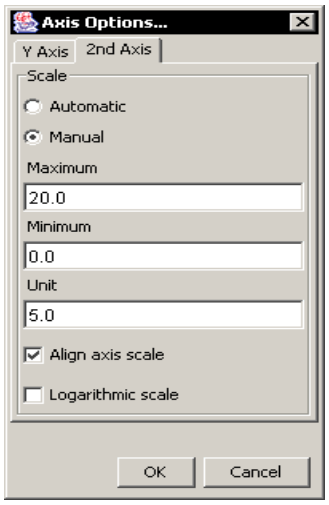


対数オプションダイアログ

このダイアログの２つオプションは次の通りです。

- **Log Base（対数の底）：**log の値の底を指定します。
- **Show Log Value（対数の値の表示）：**軸ラベルに対数の値を表示します。

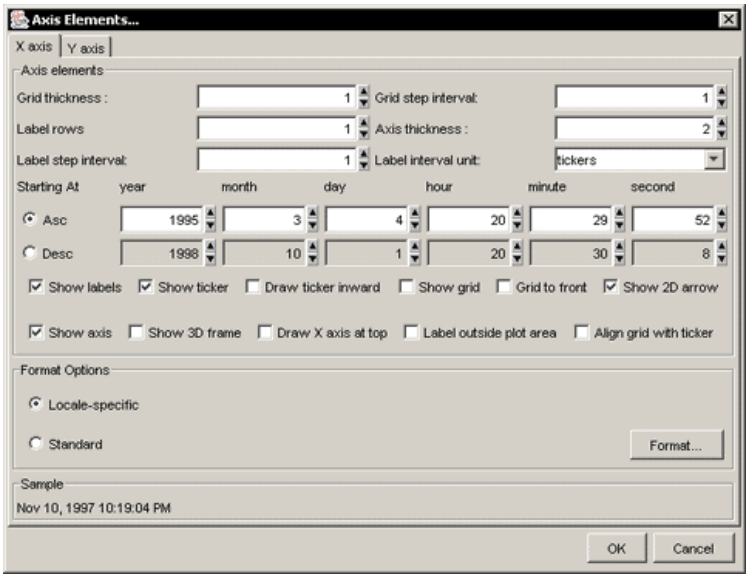
軸目盛ダイアログはチャート各値軸にタブがあります。他の軸に目盛を設定するのはその軸軸のタブをクリックします。セカンダリ軸に利用できる特別なオプションは軸目盛を揃えることができます。これはプライマリ軸からの全てのオプションをセカンダリ軸に適用するので、両軸は全く同じ目盛になります。



セカンダリ軸の軸目盛ダイアログ

7.7.2 軸エレメント

軸の外観プロパティと軸ラベルは軸エレメントダイアログからコントロールされます。軸エレメントダイアログを立ち上げるにはフォーマット（Format）メニューから 'Axis Elements' を選択するか、ツールバーの'Format Value Elements' ボタンをクリックします。これにより、チャートの全ての軸のエレメントをカスタマイズできるダイアログが立ち上がります。このダイアログではダイアルチャート、パイチャートの外観もコントロールできます。



軸エレメントダイアログ

このダイアログにはチャートの各軸ごとにタブがあります。実行するオプションは次のとおりです。なお、いくつかのオプションは特定のチャートタイプや特定のデータ、特定の軸にのみ有効になっているものもあります。

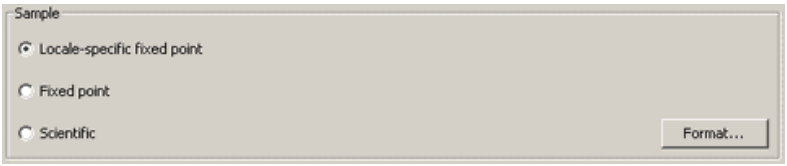
- **Grid thickness（グリッドの太さ）：**軸に沿っているグリッドラインの太さを指定します。
- **Grid step interval（グリッド間隔）：**軸に沿っているグリッドラインのグリッド間隔を設定します。
- **Label rows（ラベルの交互列）：**ラベルを交互の列で表示できます。互いに重なり合ってしまうことを防ぎます。このオプションは X 軸にのみ使用できます。
- **Axis thickness（軸の太さ）：**軸の太さをピクセルで設定します。この設定はチャートの全ての軸に適用されるので注意してください。
- **Label step interval（ラベル間隔）：**データのラベル間隔を設定します。例えば、2 を設定するとチャートの隔データポイント（データポイントをひとつ置き）にラベルを表示します。
- **Label interval unit（ラベル間隔の単位）:**時間に基づくデータ（日付、時刻、タイムスタンプ）を分類して表示する場合、使用される単位を選択します。“tickers”を選択するとデータはチャートデザイナーによって読み込まれているようにそのデータを使います。
- **Ascending/Descending（昇順/降順）：**T 時間ベースのデータをフィルタリングして並べます。データは昇順または降順に並び替えられます。また、データの開始点（または終了点）を指定することもできます。
- **Show labels（ラベルの表示）：**各軸ラベルを表示または非表示にします。
- **Show ticker（ティッカーの表示）：**軸のティッカーを表示または非表示にします。
- **Draw ticker inward（ティッカーを内側に描く）：**軸ティッカーをプロットの外側（規定値）ではなく内側に描きます。
- **Show sub-ticker（サブティッカーを描く）：**この機能は対数の底を 10 に設定したときの値軸にのみ使用することができます。値軸上のポイント間に区間ティッカーを描きます。.
- **Show grid（グリッドの表示）：**各軸のグリッドを表示または非表示にします。
- **Grid to fron（グリッドを前面へ）：**チャートのデータエレメントの上からグリッド線をを引きます。規定値ではデータポイントがグリッドの上から描かれます。

- **Show 2D arrow(2D 矢印表示):** 軸の先端に矢印を表示または非表示にします。このオプションは全てのチャート軸に適用しますので注意してください。また、このオプションは２D チャートにのみ有効です。
- **Show axis（軸の表示）:** 軸（２D チャート）または壁（３D チャート）を表示または非表示にします。
- **Show 3D frame（３D フレームの表示）:** チャートの周りのフレームを表示または非表示にします。このオプションは全てのチャート軸に適用しますので注意してください。また、このオプションは３D チャートにのみ有効です。
- **Label outside plot area（ラベル外側表示）:** 軸がどこにあるかにかかわらず、ラベルをプロットエリアの外側に配置します。正と負の両方の値を持つデータを描く場合、この機能をカテゴリ軸ラベルに使うと便利です。
- **Align grid with ticker（ティッカーにグリッドを合わせる）:**グリッド線をティッカーの間に配置するのではなく、グリッド線をティッカーに合わせます。つまり、ティッカーとグリッド線が同一線上に一致して配置されます。このオプションはカラムタイプチャートのカテゴリ軸にのみ適用されます。
- **Swap Y-axis position（Y 軸のスワップ）:** プライマリ値軸とセカンダリ値軸をスワップします。このオプションはセカンド軸のタブにしかありません。
- **Draw X-axis at top（X 軸を上部に描く）:** X 軸を規定値の下方にではなく、チャートの上部に配置します。このオプションは２D のカラム、バー、分散、Hight-Low、HLCO、バブル、ガントの各チャートに有効です。

7.7.2.1 軸ラベルのフォーマット

軸エレメントダイアログは、軸上にプロットするデータタイプによる軸ラベルの外観をフォーマットすることもできます。軸エレメントダイアログの"Format Options"にラベルフォーマットダイアログが含まれています。

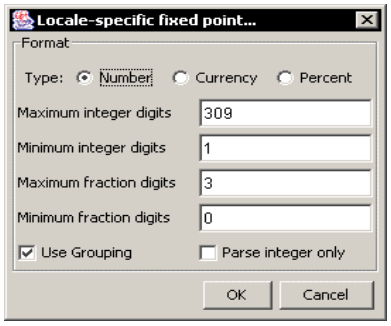
Formatting Numeric Data（数値データのフォーマット）: 数値データには３つの主な表示形式のオプションがあります。locale specific fixed point（地域固有固定小数点）、fixed point（固定小数点）、scientific （科学的）。オプションを選択したら、'Format'ボタンをクリックして、追加オプションへ行きます。



数値データフォーマットオプション

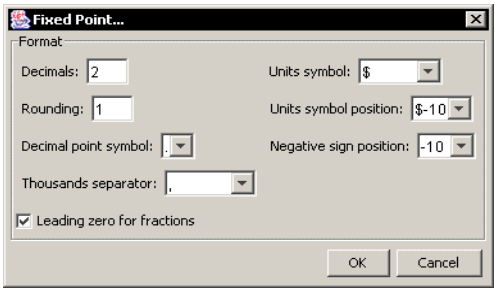
Locale-Specific Fixed Point（地域固有固定小数点）: データがビューされる地域によってデータ形式が変更します。このオプションの追加フォーマットは、データが数値か通貨かパーセントのどれで表示するかを指定します。

さらに、整数と分数の最大数と最小数を設定します。表示属性は地域によっていろいろあります。



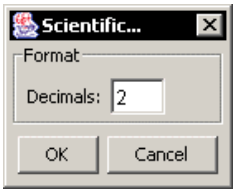
（ロケール固有フォーマットおオプション）

Fixed Point（固定小数点）：データ形式は地域によって変わることはありません。このオプションの追加フォーマットは Decimals （小数点以下の桁数）Rrounding（四捨五入する桁数）、Unit symbols （単位）Negative signs（負の号）、Decimal and Thousands separator （小数点と 1000 の区切り記号）、Leading zeros for fractions（分数（＋1 と－１の間の小数）の小数点の前に０をつける。（例えば.012→0.012）



固定小数フォーマットオプション

Scientific（科学的）：科学的表記法のデータを表示します。このオプションの追加フォーマットは小数点以下の桁数を指定します。



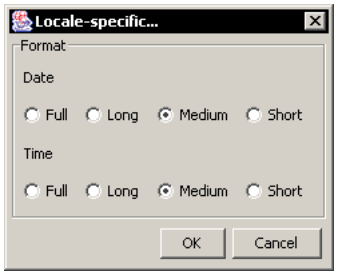
科学的フォーマットオプション

Formatting Date/Time Data（日付/時間データのフォーマット）：日付/時間データには表示フォーマットとして 2 つのオプションがあります。地域固有と標準の 2 つです。どちらかのオプションを選択して'Format'ボタンをクリックして追加オプションへ行きます。データの性質によって可能なオプションが変わります。日

付、時間、タイムスタンプデータは日付、時間、日付と時間オプションにそれぞれ対応します。

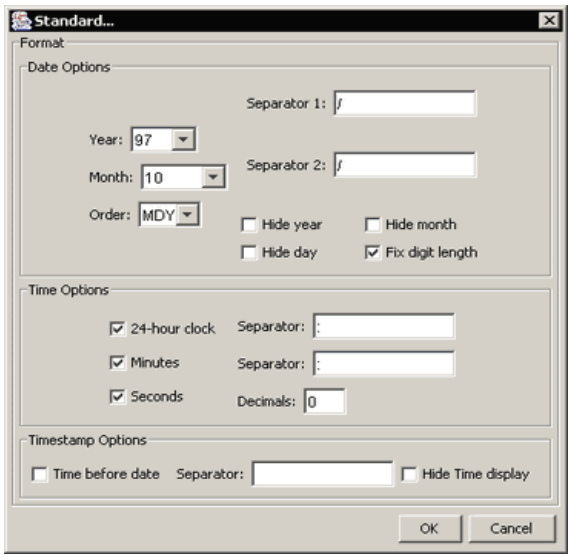
日付/時間データフォーマットオプション

Locale-Specific（地域固有）：データがビューされる地域によってデータ形式が変更します。このオプションの追加フォーマットは日付と時間情報をFull(全て)、Long（長く）、Medium（中位）、Short（短く）表示のいづれかを選びます。他の表示属性は地域によって変わります。



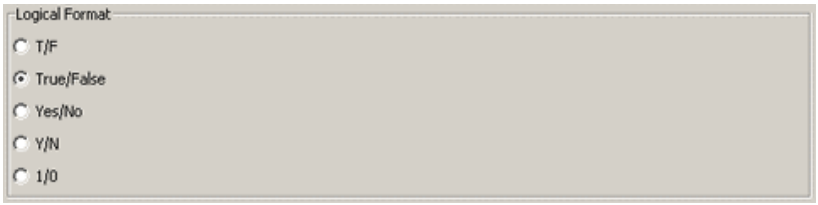
地域固有フォーマットオプション

Standard（標準）：データ形式は地域によって変わることはありません。このオプションの追加フォーマットは年と月の表示を選び、表示される月、日、年の順序も同様に選択します。セパレーター（区切りに使われる記号）も選択します。時間オプションは時、分、そして/または秒を選び、それらのセパレーターも選びます。タイムスタンプデータはその日付の前または後の時間を表示することができ、セパレーターも選択できます。



標準フォーマットオプション

Formatting Logical Data（論理データのフォーマット）：T 論理またはブーリアンデータには 5 つのオプションがあります。T/F、 True/False、Yes/No、 Y/N、1/0。



論理データフォーマットオプション

軸エレメントダイアログで'OK'をクリックすることによって、データフォーマットに加えた変更が反映されます。文字列データには追加フォーマットオプションはありません。

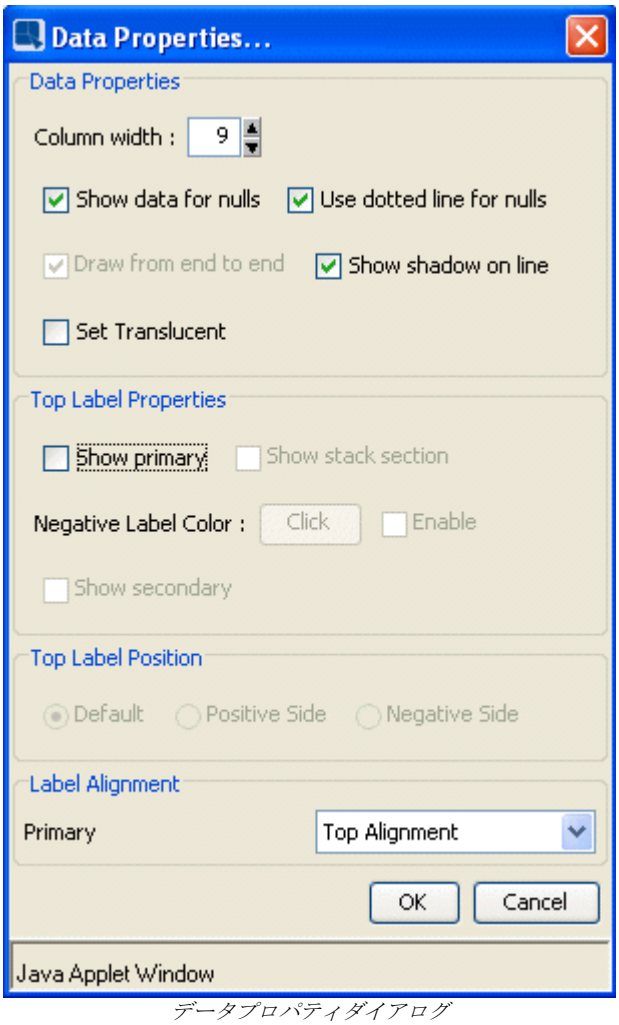
7.8 プロット/データ エレメントのフォーマット

EspressChart では、チャートそれ自体を描くのと同様に、データポイントをどのようにチャート上で描いたり、注釈をつけたりするか設定するカスタマイズやコンフィグができるようになっています。

7.8.1 データプロパティ

ほとんどのデータ表示オプションはデータプロパティダイアログでコントロールされています。このダイアログからバー/カラムのサイズをコントロールでき、表示オプションに Null 値を設定し、データラベルをオプションに指定することができます。データプロパティダイアログを立ち上げるにはフォーマット（Format）メニューの

'Data Properties' を選択するか、またはツールバーの'Data Properties'ボタンをクリックします。これにより次のようなダイアログが立ち上がります。



データプロパティダイアログ

このデータプロパティダイアログには次のようなオプションがあります。

- **Column width（カラムの幅）：**チャートの連続するバーの間の隙間に対するバー/カラム幅の割合を指定します。単位は 1/10 でデータポイントとデータポイントの間のスペースを 10 等分します。つまり、”9”と入れるとデータポイント間のスペースの 10%を隙間に割り当てることになり、”10”と入れた場合はバーとバーの間、カラムとカラムの間に隙間が全く入らないことになり

ます。このオプションは２D バー、カラム、スタックバー、スタックカラム、High-Low、HCLO、ガントの各チャートにのみ適用されます。３D チャートのカラムの厚さ（幅）コントロールするにはナビゲータパネルのシェイプスライダーの Thickness（厚さ）を使います。

- **Show data for nulls（Null のデータ表示）：**Null データがあった場合ラインをつなげます。例えば、3 ポイントあって、ポイント 1 の値が 4、ポイント 2 が Null、ポイント 3 が 6 だった場合、EspressChart は 3 ポイント間を 4 から 6 へラインを引きます。このプロパティはラインチャートまたはライン付き２D チャートでのみ有効です。他のチャートタイプは Null のデータは描きません。

- **Use dotted lines for nulls（Null にドットライン）：**チャートの Null 値に完全なラインを引くのではなく、値のないところにはドットラインを引くようにこのオプションを使って指定することができます。Show data for null オプションのようにこのプロパティはラインにのみ有効です。

- **Draw from end to end（端から端まで使って描画）：**2D のラインチャートとエリアチャートを最初と最後のデータポイントにオフセットをとって描くのではなくプロットエリアの全体にチャートを描くことができます。

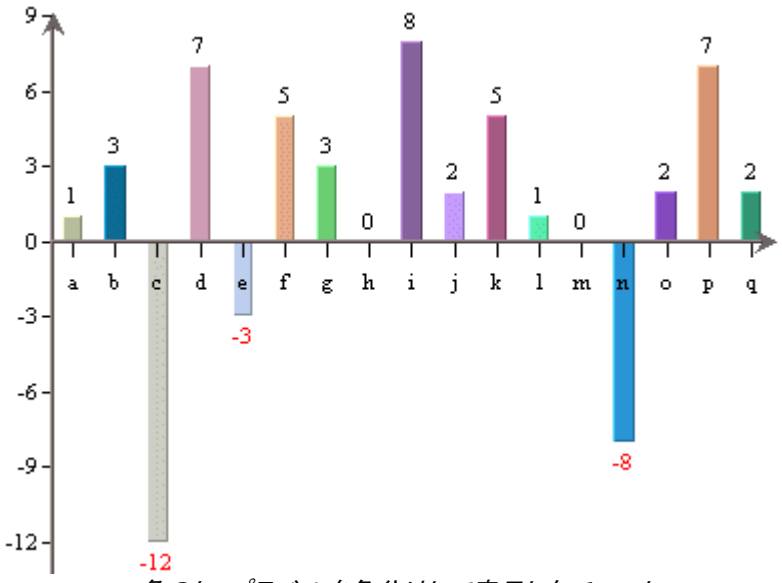
- **Show shadow on line（線上に影）：** ２D ライン上に影を描くかどうか指定します。影を適用するにはラインが 1 ピクセより太くなければなりません。

- **Set translucent（半透明の設定）：**エレメントが重なっても見えるように全てのデータポイントを半透明にします。このオプションはレーダーチャートとデータシリーズ付き２D エリアチャートには効果的です。また、バブル、ガント両チャートにも有効です。この機能は Java1.2 またはそれ以上を必要とするので、アプレットでチャートをデプロイしてクライアントが Java プラグインを使用していない場合は半透明エレメントは表示できません。

- **Show primary（プライマリの表示）：**チャートのプライマリの値にデータトップラベルを表示します。

- **Show stack section（スタックセクションの表示）：**T スタックバー、スタックカラム、スタックエリアチャートの各スタックセクションにそれぞれのラベルを表示します。

- **Negative Color Label（負のカラーラベル）:**オリジナルの値よりも小さい値を持つトップラベルを別の色で表示します。色は、この機能を有効にした後、”Click”ボタンを使用して選択することができます。



負のトップラベルを色分けして表示したチャート

- **Show secondary（セカンダリ表示）：**チャートのセカンダリの値にデータトップラベルを表示します。

- **Top label position（トップラベルの位置）：**データトップラベルをどこに描くかを指定します。規定値はデータが正の場合はポイントの上に、負の場合はデータポイントの下に描きます。規定値以外のオプションを使うと、データポイントを正の側に、または負の側に表示させることができます。

- **Label Alignment（ラベルのアラインメント）：**データトップラベルのアラインメントを設定します。データトップラベルをデータポイントの上、下または中央に描くことができます。更にラベルをデータポイントの内側の上または下に描くことができます。追加オプションではスタックチャートのスタックセクションラベルのアラインメントを設定することができます。

7.8.2 日付/時間ベースのズーム

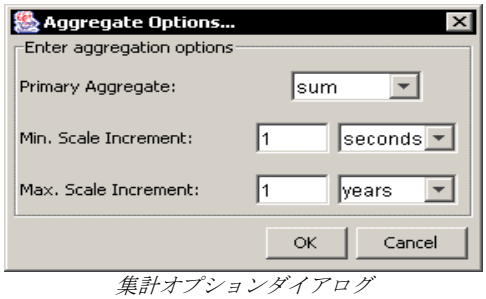
EspressChart では、カテゴリ軸の日付または時間を表示する場合、日付/時間ベースのズームを実行できるユニークな機能を提供しています。この機能を使用するにはカテゴリエレメントをユーザ定義の期間（インターバル）でグループ化して、各グループのデータポイントの集計をとります。結果の上限、下限を指定してデータをフィルタリングすることもできます。

例えば、過去 2 年間の毎日の売り上げデータがあるとします。ズームを使うにはデータを集計して月、4 半期あるいは年ごとの平均を出します。上限下限を使うと、特定の 4 半期の週の売上げに対して範囲を狭めてフィルタリングするというようなことができます。

ズームは High-Low、HLCO、分散、サーフェス、ボックス、ダイアル、レーダー、バブル、ポーラ、ガントの各チャート以外のチャートで使用できます。

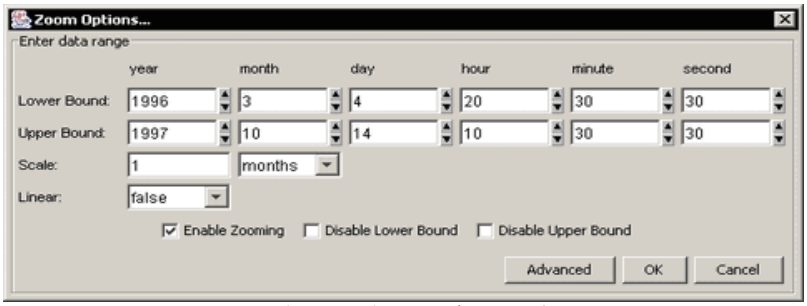
7.8.2.1 ズームの追加

日付、時間またはタイムスタンプデータがカテゴリ軸にある新しいチャート作成するとき、自動的にズームオプションが指定できるプロンプトが出されます。手動でズームオプションダイアログを出すには、フォーマット（Format）メニューから'Zoom Options' を選択します。
ズーム追加を最初に選択すると、ダイアログが現れてグループ化されたデータポイントの集計オプションを設定するようにプロンプトが出されます。



集計オプションダイアログ

このダイアログで合計、最小値、最大値、平均または総数（カウント）の集計を指定できます。また、データをズームするときに使われる増加スケールの最大と最小もしていすることができます。オプションを指定したら、'OK'をクリックすると、新たなダイアログが出力されます。そこで追加ズームオプションを指定することができます。



ズームオプションダイアログ

このダイアログではデータの上限、下限とデータのグループ化したい期間（インターバル）を指定できます。スケールは集計オプションダイアログで指定した最大スケールと最小スケール以内でなければなりません。

このオプションはチャートのリニアスケールも保存できます。"Linear" オプションに True を設定すると、特定のグループに関連するデータがない場合でも、チャートはグループ化した期間のポイントを常に表示します。例えば、4、５、６月の 3 ヶ月間の売上げデータがあったとします。もし 5 月の入力データがなかった場合、そして、“Linear”オプションを ture の設定すると 5 月の値は０で描かれます。"Linear" オプション を False にすると、4 月データポイントの後すぐ 6 月が続きます。

ダイアログの下方にあるチェックボックスを使用して、ズームに加えて下限および上限の制限を有効/無効にするきおとができます。すべてのオプションを指定したら、'OK'をクリックます。ズームがチャートに適用されます。

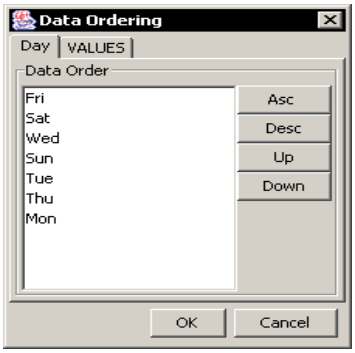
7.8.2.2 チャートビューでのズーム

チャートビューワを使ってチャートをデプロイする場合、エンドユーザはダイナミックズームを実行することができます。チャートビューワで時間シリーズズームを実行するには CTRL キーを押しながらチャート上のポイントをクリックし、他のポイントにドラッグします。これにより自動的にマウスを使って選択された下限上限を基にズームします。集計はデザイン時にセットされたオプションにより実行されます。CTRL を押しながら右クリックすることによってズーム解除ができます。

スケール期間はチャートビューワで ALT+ Z を押すことで変更できます。これによりダイアログが立ち上がりズーム設定の変更ができます。

7.8.3 データの順序

EspressChart はカテゴリとシリーズエレメントがチャートで描かれる順序を変更することができます。順序を変更するにはデータ（Data）メニューから'Ordering' を選択するかツールバーの'Data Ordering' ボタンをクリックします。これにより次のダイアログが立ち上がります。

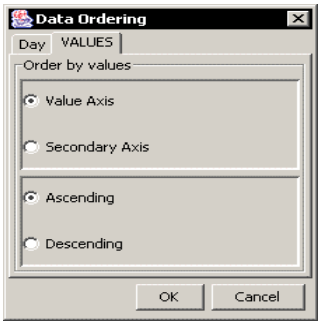


データ順序ダイアログ

ダイアログのはカテゴリエレメントのタブ、シリーズエレメントのタブ、"VALUES" と書かれたタブがあります。カテゴリとシリーズエレメントには次のようなオプションがあります。

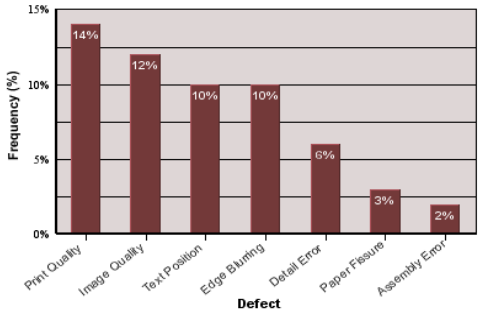
- **Asc（昇順）：**カテゴリ/シリーズエレメントを昇順に並べ替えます。例えば、カテゴリエレメントが文字列ならば、アルファベット順に並べ替えられます。
- **Desc（降順）：**カテゴリ/シリーズエレメントを降順に並べ替えます。
- **Up（アップ）：**リスト中のカテゴリ/シリーズエレメントをひとつ上に移動して、特別の順番にします。
- **Down（ダウン）：**リスト中のカテゴリ/シリーズエレメントひとつ下に移動します。.

カテゴリエレメントの値を基にそれらをソートすることもできます。それにはデータ順序ダイアログで"VALUES" タブを選びます。



値によるデータ順序ダイアログ

このダイアログから値軸またはセカンダリ値軸で値を基にカテゴリエレメントのソートを指定します。昇順でそーツするか降順でソートするか指定します。値チャートによってソートされたこの結果はパレットチャートと呼ばれ、工程管理アプリケーションにしばしば使われます。



パレットチャート

7.8.4 ヒストグラム

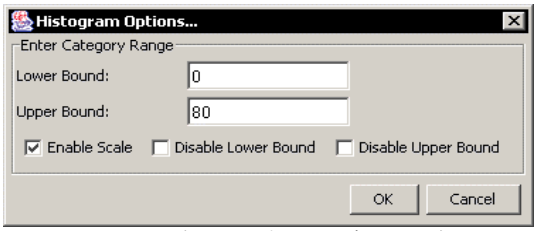
ヒストグラムはイベントが起きる頻度を追跡したり、特定の範囲にデータが分布する様子を表したりする便利な分析ツールです。EspressChart ではチャートのカテゴリエレメントを基にヒストグラムを描くことができます。時間ベースデータ（日付、時

間、タイムスタンプ）以外の全てのカテゴリデータにヒストグラムを描くことができます。

ヒストグラムはデータポイントやあるいは各カテゴリエレメントの実例を数えることによって計算されます。数値カテゴリの場合は頻度の範囲を作るためのスケールと同様に上限と下限を指定することができます。
ヒストグラムを作成するには、フォーマット（Format）メニューから'Histogram Options' を選択します。ダイアログが表示されヒストグラム描画を選択します。

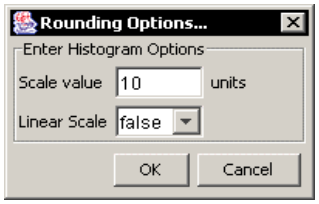
ヒストグラム選択ダイアログ

'Draw Histogram' オプションを選択すると、ヒストグラムを描くためのオプションを指定する別のダイアログが表示されます。このダイアログで描かれるデータの下限または上限を選択します。限度を設定するとヒストグラムでは下限（Lower Bound）と上限（Upper Bound）によって指定された範囲より外側のデータは数なくなります。通常は限度を設定するのは数値カテゴリエレメントとなるのが順当です。



ヒストグラムオプションダイアログ

数値カテゴリエレメントでは、頻度数を計算するための値の範囲やグループ分けのためにスケールを指定します。スケールを指定するのは'Enable Scale' オプションを選択します。これにより、スケールを指定することができる新しいダイアログが立ち上がります。



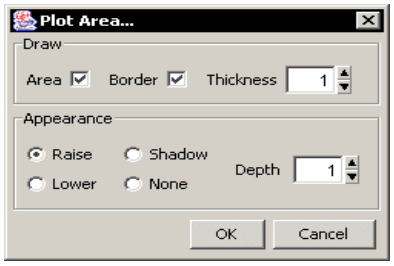
ヒストグラムスケールオプション

ここでは各スケールステップ中にある単位の数を指定します。リニアスケールを保存するかどうかも指定します。 これは、たとえ対応するデータポイントがなかった場合でも、カテゴリ軸のその範囲を描画します。例えば、頻度数が 20 と 30 の間では０だった場合、"30" というイエレメントは総数０でカテゴリ軸に表示されます。しか

し、リニアスケールをオフにしていると"30" というエレメントはチャートに描かれません。

7.8.5 プロットエリアのフォーマット

２Ｄチャートでは、プロットエリアはデータポイントが描かれる平面です。フォーマット（Format）メニューから'Plot Area' を選択してプロットエリアの外観をカスタマイズすることができます。カレントチャートが２Ｄチャートだとすれば、次のようなダイアログが出力されます。

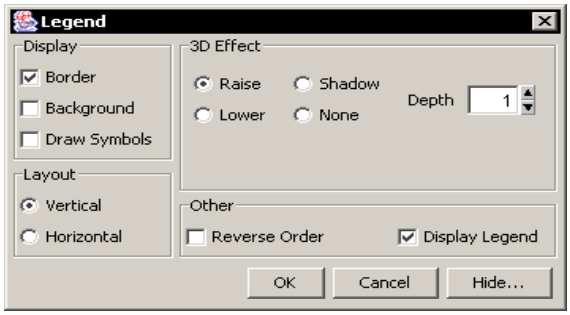


プロットエリアダイアログ

このダイアログではプロットエリアの周りの境界線を引いたり、背景色をつけたりすることができます。エリアを塗りつぶす場合、Raise（隆起）、Lower（へこます）、Shadow をつけることによって、3D 効果を出す指定もできます。

7.8.6 チャート凡例のフォーマット

フォーマット（Format）メニューから'Legend'を選択するか、またはツールボタンの'Format Legend'ボタンをクリックすることによって凡例の表示を変更することができます。これによって次のようなダイアログが立ち上がり、凡例プロパティをカスタマイズすることができます。



凡例フォーマットダイアログ

ダイアログには次のオプションがあります。

- **Display（表示）：** 凡例の境界線のオンまたはオフ、背景のオンまたはオフができます。そして、凡例の中に線やブロックの変わりにポイントシンボルを表示することもできます。

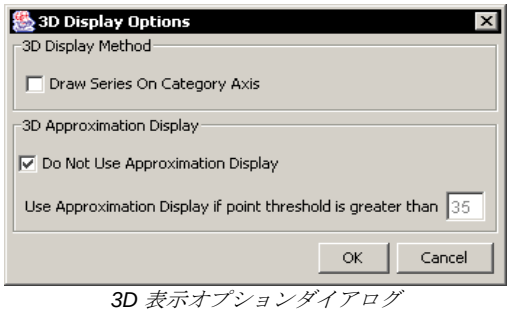
- **3D Effect（3D 効果）：** 凡例に 3D 効果を追加することができます。隆起させるか、へこませるかまたは影を描くことができます。
- **Layout(レイアウト):** 凡例を縦から横のレイアウトに変更します。
- **Other（その他）：** 凡例を表示するかしないか、また凡例を逆順に描くかを指定できます。

さらに、'Hide'ボタンをクリックして、特定のカテゴリ/シリーズエレメントを凡例から取り去ることができます。この操作により凡例アイテムのリストが出力され、隠したいエレメントを選ぶことができます。

7.8.7 完全に近い 3 次元化（3D Rendering Approximation）

EspressChart ではとても少ないソース修正で、パンしたり、ズームしたり、回転したりする実際の 3 次元のようなチャートの 3 次元化をします。しかし、3 次元化はメモリをたくさん使い、CPU も集中的に使用します。チャートにデータポイントがたくさんある場合（３D 分散チャート、サーフェスチャートなど）、チャートを作成するときにメモリー不足になる可能性があります。これを解決するのに EspressChart には完全に近い描画（rendering approximation）機能があります。これを使うとチャートは完全に描画するのではなく、しかし多くのポイントが表示されて容認できる程度で描かれます。

この機能がオンになる閾値の規定値は 100 ポイントです。これはどういう意味かと言うと、３D チャートに 100 データポイント以上あるとき、この完全に近い描画機能が使われます。フォーマット（Format）メニューから'3D Display Options' を選択してこの描画機能をオフにしたり、閾値を変更したりすることができます。これには次のようなダイアログが立ちあがります。



3D 表示オプションダイアログ

D3 Approximation には 2 つのオプションがあり、1 つは Approximation 機能のオン/オフで、もう１つは閾値の設定です。

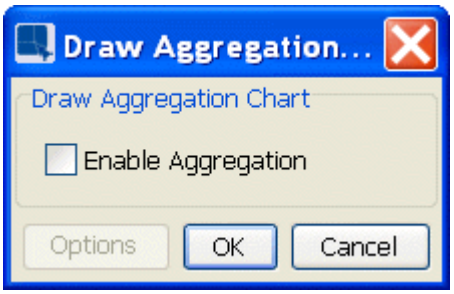
7.8.8 カラムの境界線

カラム、バー、スタックカラム、スタックバー、各チャートは、EspressChart では、カラムの周りに境界線を描くことが できます。境界線オプションを設定するにはフォーマット（Format）メニューから'Data Border' を選択します。そうすると境界線オプションを設定できるダイアログを立ち上げることができます。

7.8.9 集計

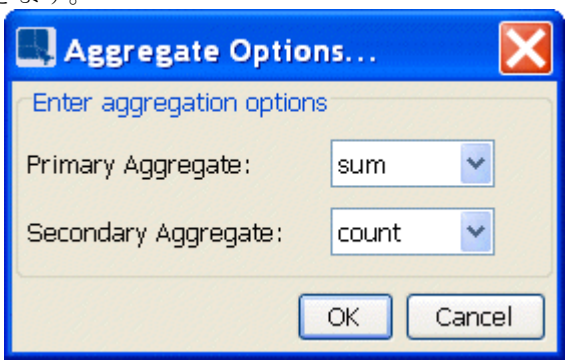
EspressChart は、カテゴリー（もし、存在する場合、シリーズおよびスタック）に該当する一つ以上のデータポイントを集計することができます。この機能で、（大勢の内の）一つのデータポイントからではなく、広い視点からデータを見ることができます。

データを集計するには、Format → Aggregation Options を選択します。ダイアログが表示され、集計を行なうことになります。

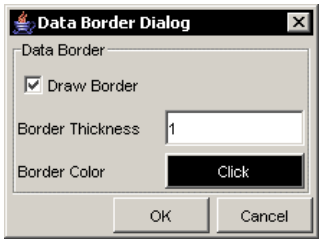


集計の選択ダイアログ

集計機能を有効したら、二番目のダイアログが表示され、適用する集計方法を選択することになります。集計方法を最大限、最低限、平均、合計、計算から選ぶことができます。プライマリー集計（プライマリー軸にマップされているコラムに適用）または、第二軸が存在する場合、第二集計（第二軸にマップされているコラムに適用）を指定することができます。



集計オプションダイアログ



データ境界線ダイアログ

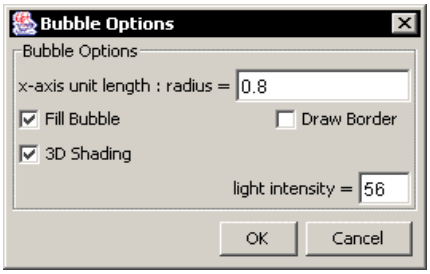
最初のオプションでは、データ境界線のオン/オフを設定できます。２番目のオプションでは、チャート内の白色領域を囲む黒色の境界線を設定できます。３番目と 4 番目のオプションでは、境界線の太さと色を設定できます。カラーボタンをクリックすると、新しい色を選択または入力できるダイアログが表示されます。

7.9 チャートの特定オプション

特定のチャートタイプだけのためのフォーマットオプションがいくつかあります。これらのオプションはフォーマット（Format）メニューからの'Chart Options' を選択または、ツールバーの'Chart Options'ボタンをクリックすることによって変更することができます。これによってカレントチャートのタイプによって異なるダイアログが表示されます。チャートタイプによっては追加オプションのないものもあります。

7.9.1 バブルチャート

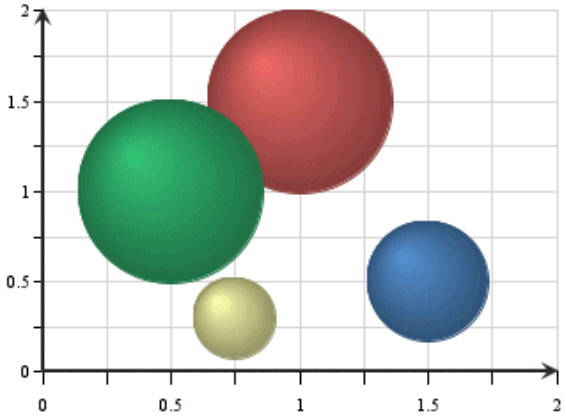
バブルチャートは次のようなダイアログを表示します。



バブルオプションダイアログ

次のオプションはバブルチャートに有効です。

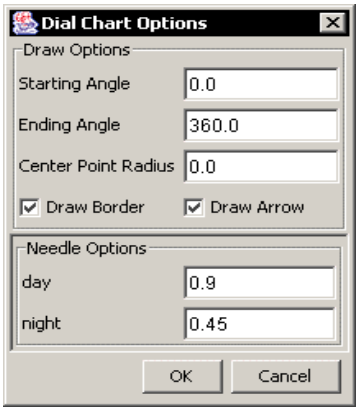
- **x-axis unit length （X軸単位の長さ）: radius（半径）:** これはX軸単位の長さ対バブルの半径を指定します。
- **Fill Bubble（バブル塗りつぶし）:** バブルの中を塗りつぶすかどうか指定します。
- **Draw Border（境界線を描く）:** バブルの周りの境界線を描くかどうか指定します。
- **3D Shading（3D の影）:** 3D の影をバブルに追加することができます。また、影の光強度も指定できます。



3D 影付きバブルチャート

7.9.2 ダイアルチャート

ダイアルチャートには次のようなダイアログが表示されます。



ダイアルオプショ h ンダイアログ

ダイアルチャートには次のオプションが有効です。

- **Starting Angle（開始角度）：**最初の軸ラベルが設定される角度を指定します。さらに、Draw Full Circle オプションがチェックされていない場合、このプロパティは境界線とダイアル領域の開始位置も決定します。角度は度で表され規定値は０。ダイアルチャートが時計の面だとすれば、０度は１２時の位置になります。
- **Ending Angle（終了角度）：**最後の軸ラベルが設定される角度を指定します。さらに、Draw Full Circle オプションがチェックされていない場合、このプロパティは境界線とダイアル領域の開始位置も決定します。角度は度で表され規定値は 360。すなわち規定値ではラベル（データポイント）は ダイアルを一周回ることになります。

- **Center Point Radius（中央ポイントの半径）：**ダイアルチャートの中心から始まる内側の丸の半径を指定します。半径はダイアルプロットの半径に対する割合を指定します。つまり値１は内側の丸が全てのダイアルを覆ってしまうことになります。 Draw Full Circle オプションがチェックされていない場合、開始角度と終了角度内にある中央ポイントの部分だけが表示されます。

中央ポイント半径付きのダイアルチャート

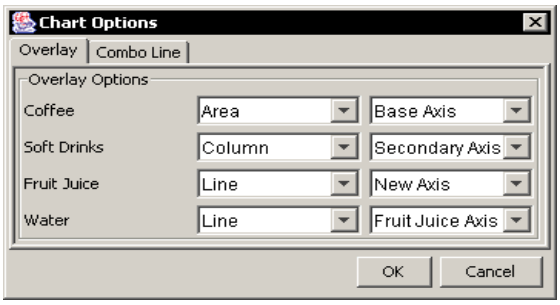
- **Draw Border（境界線の描画）：**ダイアルの周りの境界線を描くかどうか指定します。

- **Draw Arrow（矢印の追加）：**ダイアルの針の先に矢印を描くかどうか指定します。

- **Draw Full Circle（完全な円の描画）：**ダイアルを完全な円（360°）として描画するか、または開始角度と終了角度によって決定される円の一部だけを描画するかを指定します。
- **Needle Options（針のオプション）：**ダイアルの中心から張りの距離を指定します。範囲は０（ダイアルの中心）から１（ダイアルの淵）までです。針の数はカテゴリのエレメントの数と等しくなります。

7.9.3 オーバーレイチャート

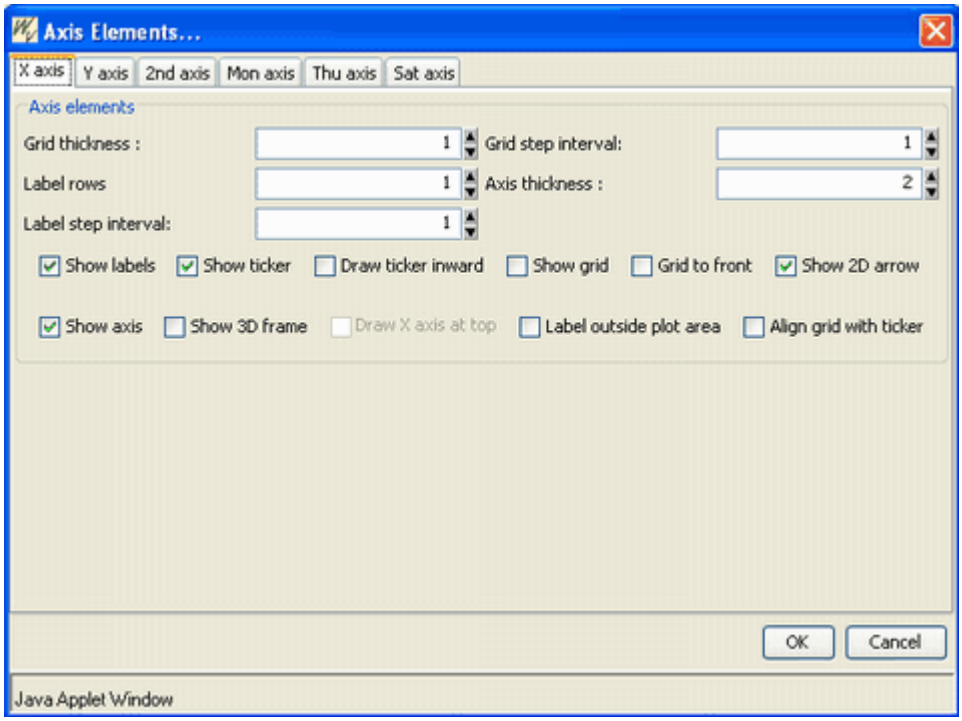
オーバーレイチャートでは以下のダイアログが表示されます。



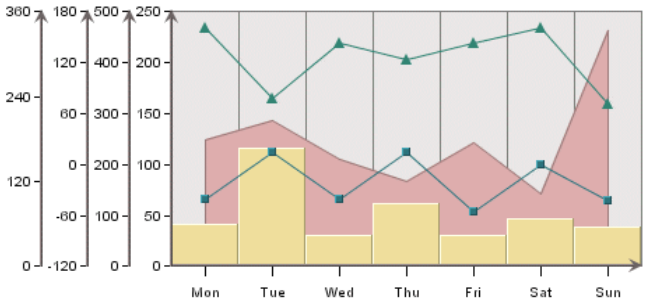
オーバーレイオプションダイアログ

このダイアログではデータシリーズの各エレメントにどのチャートタイプを使用するかを指定することができます。オーバーレイチャート内のシリーズエレメントに使用できるチャートタイプはカラム、エリア、およびラインです。任意のシリーズエレメントを表示しないよう選択することも可能です。

このダイアログではシリーズエレメントを描画するために使用する軸を指定することもできます。エレメントはプライマリまたはセカンダリ軸に配置することができ、シリーズエレメントのための新しい値軸を作成することもできます。新しい値軸を作成するにはドロップダウンメニューから'New Axis'を選択します。新しい軸を使用するよう指定したら、ドロップダウンメニューに他のデータシリーズエレメントのための新しいオプションが追加されます。これによってそのデータシリーズエレメントは既に指定した軸と同じ軸の上に描画することが可能になります。さまざまな軸を使用することによって、異なるデータシリーズエレメントが使用するスケールを正確に調整することができます。これらの軸にはそれぞれ、Axis Elements（軸要素）ウィンドウ内に専用のタブがあり、そのタブで、各軸のラベルステップインターバルを他の軸とは関係なく変更することができます。

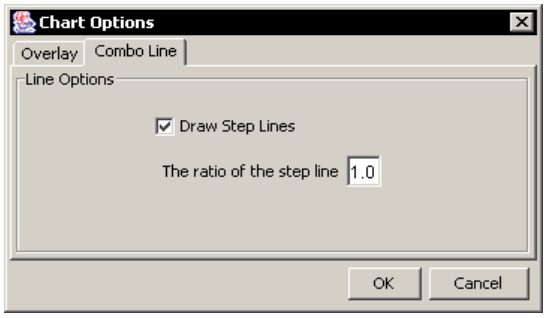


複数の値の軸に使用する Axis Options（軸オプション）ダイアログ



複数の値軸を伴ったオーバーレイチャート

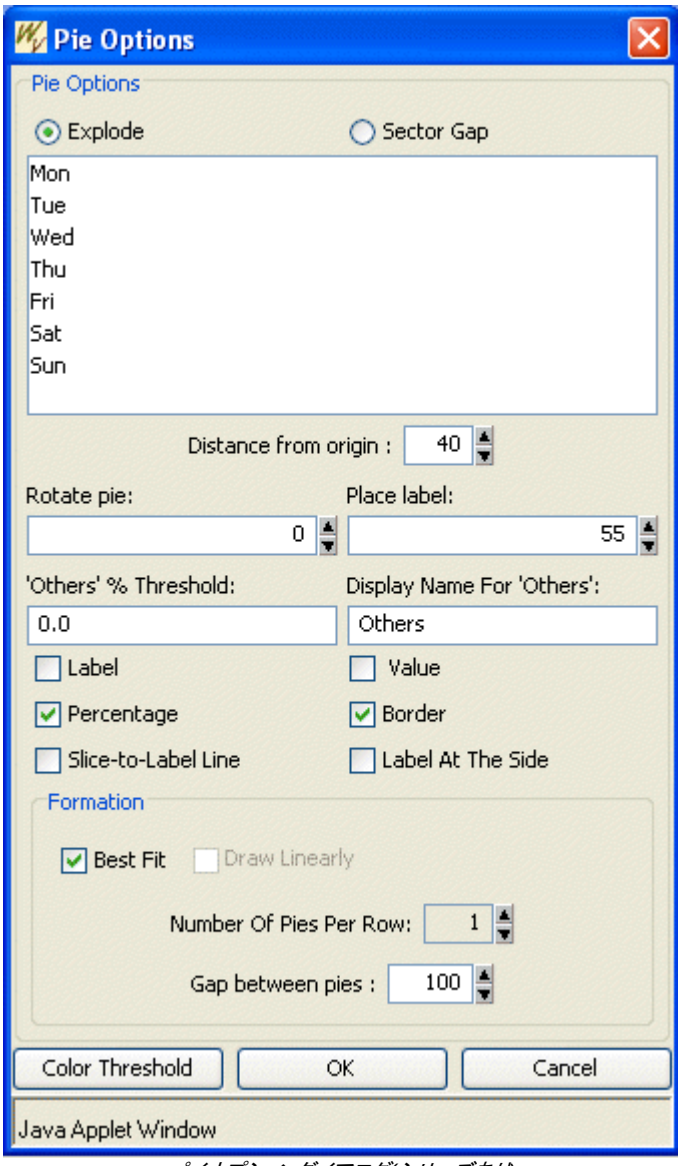
オーバーレイチャートで使用されているチャートタイプのひとつがラインチャートである場合、"Combo Line"タブによって線をステップラインとして描画するかどうかを指定することができます。ステップラインの詳細については、セクション 7.9.5 を参照してください。



オーバーレイチャートの Combo Line オプション

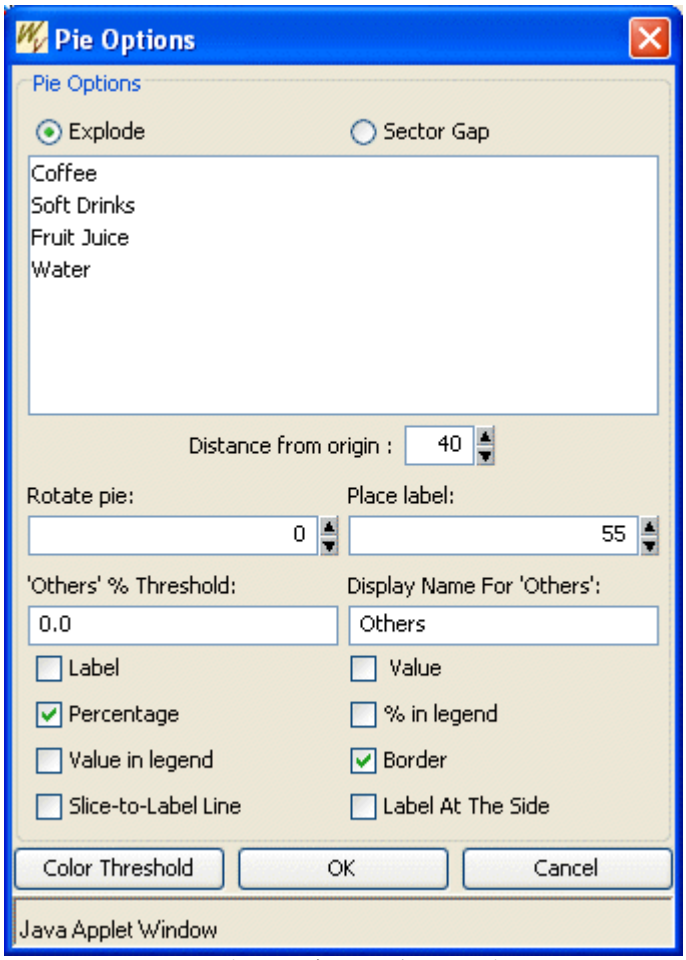
7.9.4 パイチャート

パイチャートでは以下のダイアログが表示されます（チャートがシリーズを持っているか、あるいはチャートが 2D であるか 3D であるかによって異なるオプションが表示されることに注意してください）。ここには、シリーズのある 2D チャートに利用できるオプションが表示されます。



パイオプションダイアログ(シリーズあり)

さらに、ここには、シリーズのない 2D チャートに利用できるオプションも表示されます。Formation オプションが削除され、'% in legend' および'Value in legend'オプションが追加されることに注意してください。



パイオプションダイアログ(シリーズなし)

パイチャートでは以下のオプションを使用することができます：

- **Explode（分散）：**パイの中心から分離して描画されるセクションを持つ 1 つまたは複数のカテゴリ/シリーズエレメントを選択することができます。
- **Sector Gap（セクターギャプ）：**パイの中心から分離して描画されるセクションを持ちつつ、パイのスライスと円の境界線から等しい距離を維持する 1 つまたは複数のカテゴリ/シリーズエレメントを選択することができます。
- **Distance from origin（元からの距離）：**分離/セクターギャップセクションを中心からどれだけ離れて描画するかを指定します。この数字は半径のパー

センテージで表され、中心とエキスポートされるパイスライスの先端との距離を示します。

- **Rotate pie（パイの回転）：**チャートを時計方向に回転させる角度を指定します。0 から 360 までの値を指定できます。

- **Place label（ラベルの配置）：**ラベルのパイの中心からの距離を示します。個々のレーベルの位置はテキストをドラッグすることによって調整できます。

- **'Others' % Threshold（'Others'の%のしきい値）：**この機能は、多数の小さなカテゴリに分かれたパイチャートに有用です。ユーザはカテゴリのスライスを描画するのではなく、しきい値を選択することができます。値カラムの割合(%)がしきい値未満のカテゴリは”Others”スライスにまとめられます。

- **Display Name for 'Others'（'Others'の表示名）：**設定されたしきい値未満のカテゴリ用に作成される”Others”スライスの表示名を設定できます。このラベルは凡例内かまたはスライスラベルとして、あるいはその両方で表示されます。
- **Pie sectors（パイセクター）：**パイを構成するセクターの数を示します。分割数はパイのセクターの数を増減することによって調整できます。より高い数字を選択すると分割数はより多くなります。描画されるパイのセクターの数は 24 から 180 の間で指定することができます。このオプションは 3D のパイチャートでのみ使用することができます。

- **Label（ラベル）：**各スライスごとにカテゴリ/シリーズレーベルが描画されるかどうかを決定します。デフォルトではレーベルは凡例アイテムとしてのみ表示されます。

- **Value（値）：**各スライスの実際の値を表示するかどうかを指定します。

- **Percentage（パーセンテージ）：**パイの各スライスのパーセンテージを表示します。パーセンテージは各セクションの値をすべての値の合計で割ることによって計算されます。

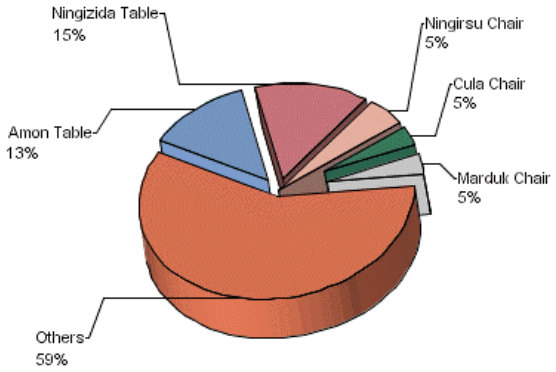
- **% in legend（凡例に表示する%）：**パイの各スライスが表すパーセンテージを凡例に表示できます。スライスが非常に細くなった場合、こうした表示を行うと見やすさが向上します。このオプションはデータシリーズを持たないパイチャートでのみ使用することができます。

- **Value in legend（凡例に表示する値）：**凡例にパイの各スライスが表す値を表示できます。スライスが非常に細くなった場合、こうした表示を行うと見やすさが向上します。このオプションはデータシリーズを持たないパイチャートでのみ使用することができます。

- **Border（境界線）:** パイの各スライスに境界線を付けるかどうかを指定します。このオプションは 2D のパイチャートでのみ使用することができます。3D のパイチャートではナビゲーションパネル上の境界線描画オプションを使用することができます。

- **Slice-to-Label Line（スライスとラベルを結ぶ線）:** ラベルから対応するスライスへ線を描画します。このオプションは 2D のパイチャートにのみ使用することができます。

- **Label at the Side（横にラベル）:** チャートの外のプロットのパイチャートから離れたところにラベルをつけます。"Slick-to-Label Line" オプションを使うとき、小さいカテゴリのあるチャートに、テキストが重なり合わないようにしてパイラベルを表示することができます。



サイドラベルと線付きのパイチャート

- **Best Fit（最適化）:** 複数のパイがチャートのキャンバス内に適切に収まるようアレンジします。データシリーズを伴うパイチャートにのみ使用することができます。

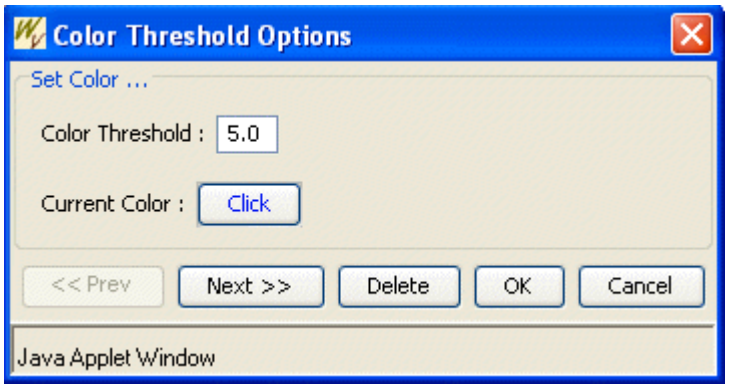
- **Draw Linearly（直線描画）:**複数のパイを水平な直線上に並ぶようアレンジします。データシリーズを伴うパイチャートにのみ使用することができます。

- **Number of Pies Per Row（複数パイの横並び）:**複数のパイのカスタムアレンジメントを作成し、アレンジメントの各横一列ごとに任意の数のパイを並べることができます。

- **Gap between pies（パイとパイの間）:** 複数のパイの間のギャップを指定することができます。数はパイの半径の倍数なので、チャートのサイズに合わせてギャップも調整されます。

- **Color Threshold（色の閾値）:** ”Click”ボタンを選択すると、パイスライスが小さいほど、より明るい色を選択することができます。下に表示されるフィールド"Color Threshold"には数字を指定します。パイスライスの割合がこ

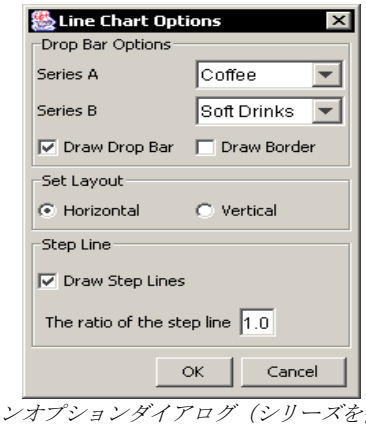
の数字よりも小さい場合、その色が適用されます。たとえば、この数字が 5 である場合、3% のパイスライスは、指定された色に色づけされます。
"next" をクリックすると、2 番目の色を選択できますが、色の閾値の数字は、すべての色に対して同一でなければなりません。したがって、2 番目の色のダイアログで、閾値の数字を変更したら、最初の色の閾値も変更されます。
複数の色を選択する場合、これらの色は、該当するパイスライスから連続して左回りに循環します。



Color Threshold （色の閾値）ダイアログ

7.9.5 ラインチャート

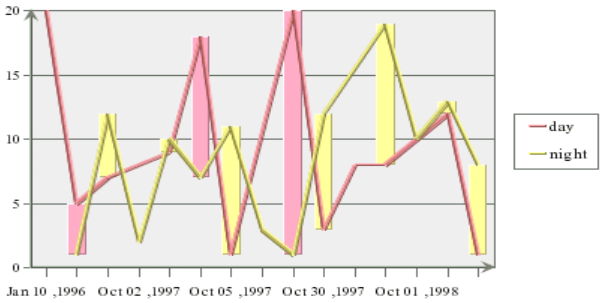
2D のラインチャートでは、チャートがデータシリーズを持っているか否かによって 2 つの異なるダイアログが表示されます。チャートがデータシリーズを持っている場合、以下のダイアログが表示されます。



ラインオプションダイアログ（シリーズを伴う）

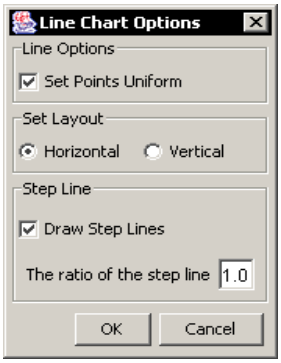
データシリーズを伴うラインチャートには、2 つのシリーズエレメントの間にドロップバーを描画するための固有のオプションがあります。ダイアログのオプションは以下のようになります：

- **Series A（シリーズ A）:**ドロップバーに使用する第 1 のシリーズエレメントを指定します。
- **Series B（シリーズ B）:**ドロップバーに使用する第 2 のシリーズエレメントを指定します。
- **Draw Drop Bar（ドロップバーの描画）:**ドロップバーを描画するか否かを指定します。
- **Draw Border（境界線の描画）:**ドロップバーの周囲に境界線を描画するか否かを指定します。
- **Set Layout（レイアウトの設定）:**ラインチャートを垂直または水平のどちらに描画するかを指定します。
- **Step Line(ステップライン):**ラインチャートをステップラインとして描画します。使用するステップラインの比率を指定することもできます。



ドロップバーを伴うラインチャート

ドロップバーの色はシリーズが持つ任意のポイントの値の高さによって変わることに注意してください。ラインチャートがシリーズを持たない場合は、以下のダイアログが表示されます。



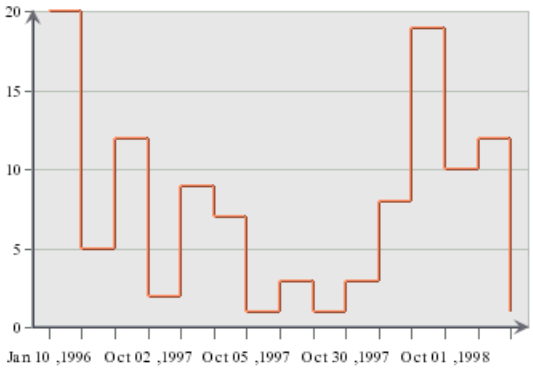
ラインオプションダイアログ（シリーズを伴わない）

ダイアログのオプションは以下のようになります：

- **Set Points Uniform（ポイントユニフォーム設定）:**ポイントの形と色を均一にするかどうかを指定します。このオプションのチェックを外すと、データポイントに複数の色と形を適用するよう設定されます。ポイントは Line and Point ダイアログでカスタマイズすることができます。

- **Set Layout（レイアウト設定）:**ラインチャートを垂直方向または水平方向のどちらに描画するかを指定します。

- **Step Line（ステップライン）:**ラインチャートをステップラインとして描画します。使用するステップラインの比率を指定することもできます。



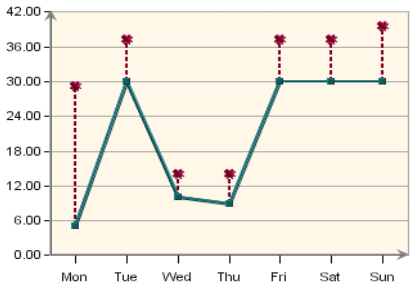
ステップラインを伴うラインチャート

ステップラインの垂直部分が描かれるポイント間の距離はステップラインレシオによって指定します。レシオが 1 の場合、チャート上の隣のポイントへ水平のラインが描画され、次にそのポイントへの垂直のラインが描画されます。レシオが 0.5 の場合、2 つのポイント間で水平のラインが半分まで描画されます。レシオが 0 の場合、ラインの垂直部分が最初に描画され、隣のポイントへ水平方向に結ばれます。レシオで指定できる値は 0 から 1 までです。

3D のラインチャートでは追加のオプションはありません。

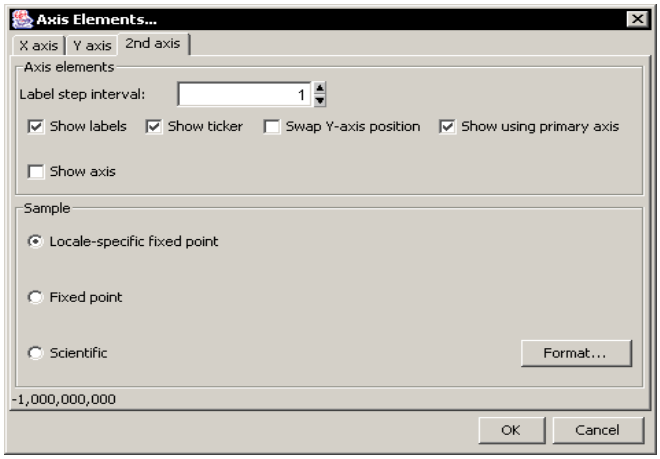
7.9.5.1 二重値ラインチャート

EspressChart はラインチャートで 2 つの値を同じラインで表現することができる特別なオプションを備えています。この場合、第 2 のデータは第 2 の軸を使用して描画され（ライン– ラインコンビネーションと同様）、第 1 軸のラインに組み合わされます。



二重値ラインチャート

二重値ラインチャートを作成するにはライン– ラインコンビネーションチャートをデザインします（第 1 値と第 2 値を持つラインチャート）。次に Format メニューから 'Axis Elements'を選択すると、軸エレメントダイアログが表示されます。

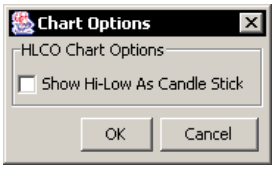


ライン– ラインコンビネーションチャートの軸エレメントダイアログ

"2nd Axis"タブの下に"Show using primary axis"と示されたチェックボックスがあります。これをチェックして'OK'をクリックすると、チャートは二重値ラインチャートとして描画されます。

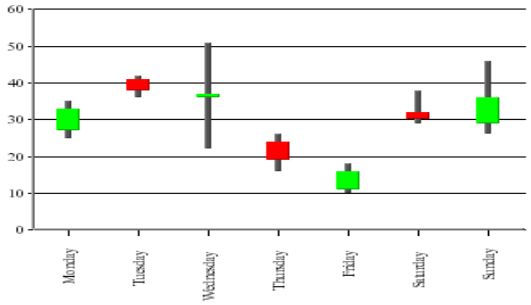
7.9.6 HLCO チャート

HLCO チャートでは次のようなダイアログが表示されます。



HLCO オプションダイアログ

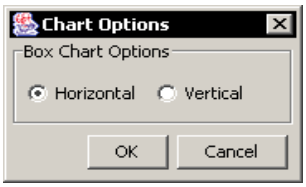
'Show Hi-Low As Candle Stick' オプションは HCLO チャートをキャンドル（ろうそく型）描写に変更します。キャンドル HLCO は高、低、クローズ、オープンのデータを融合させて、キャンドルスティック（ろうそく立て）に似た形で表現します。



HLCO キャンドルスティックチャート

7.9.7 ボックスチャート

ボックスチャートでは次のようなダイアログが表示されます。

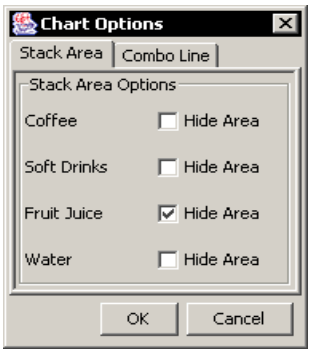


ボックスオプションダイアログ

このダイアログでボックスチャートの表示を Horizontal（水平方向）か Vertical（垂直方向）かを指定します。

7.9.8 スタックエリーアチャート

スタックエリーアチャートでは次のダイアログが表示されます。



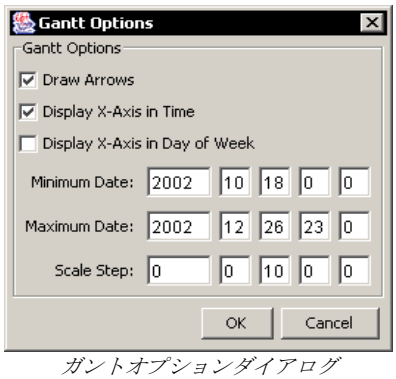
スタックエリーアオプションダイアログ

対応するチェクボックスをクリックすると、チャートのスタックを隠すことができます。"Combo Line" タブは、チャートがラインスタックエリーアコンビネーションの

場合にステップラインを指定するのに使います。3D スタックエリーアチャートには追加オプションはありません。

7.9.9 ガントチャート

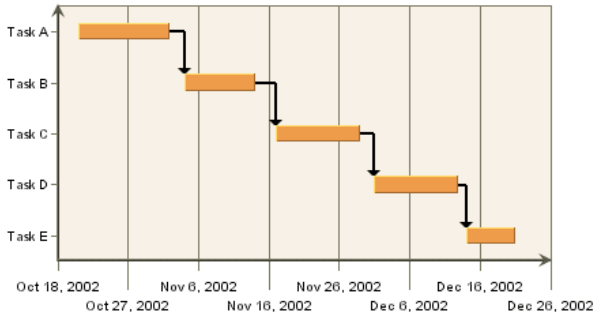
ガントチャートでは次のダイアログが表示されます。



ガントオプションダイアログ

次のオプションはガントチャートに有効なオプションです。

- **Draw Arrows（矢印表現）：**ガントチャートのカテゴリーエレメント間に矢印を描きます。つまり、スケジュールイベント間を矢印で繋いだ表現ができます。矢印はデータソースに現れるカテゴリエレメントの順番で描かれます。



- **Display X-Axis in Time（X 軸の時間表示）：**X 軸に数値の代わりに時間を値としてティッカーラベルを表します。
- **Display X-Axis in Day of Week（X 軸の曜日表示）：**一週間の曜日と日曜日の日付をX軸の値としてティッカーラベルを表します。
- **Minimum Date（日付の最小値）：**X 軸の始まりの日付を指定します。形式は年　月　日　時間　分。
- **Maximum Date（日付の最大値）：**X 軸の終わりの日付を指定します。形式は年　月　日　時間　分。

- **Scale Step（目盛り間隔）：**X 軸の目盛り間隔を指定します。形式は年　月　日　時間　分。

7.9.10 レーダーチャート

レーダーチャートでは次のダイアログが表示されます。.

レーダーオプションダイアログ

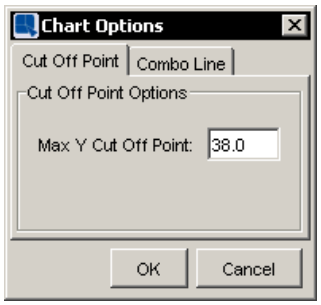
全てのレーダーチャートの X 軸の目盛りは、デフォルトで同じになるように設定されています。"Synchronize All Axes"（全てのX軸と同調）オプションのチックを外すとそのレーダーチャートの各軸を独自に目盛り設定することができます。各軸に対してオートスケーリング（自動目盛り）を使うか、またはマニュアルで目盛りを設定するかを軸目盛り（axis scale）ダイアログで選択することができます。

2 番目のオプションを使用すると、レーダーチャートのグリッドの描画方法を設定できます。デフォルトでは、グリッドが有効な場合、各軸上の目盛りを結ぶ直線で描画されます。"Draw Circular Grid"オプションを有効にすると、ポーラーチャートのグリッドと同様、グリッドは円で描画されます。

3 番目のオプションを使用すると、レーダーチャート内のデータポイント（領域）のカットオフポイントを指定できます。チャートに表示されなければならない最大値を入力できます。データポイントでバインドされる領域は、指定されたカットオフポイントを超えて描画されません。

7.9.11 分散チャート

分散チャートの場合、次のダイアログが表示されます。



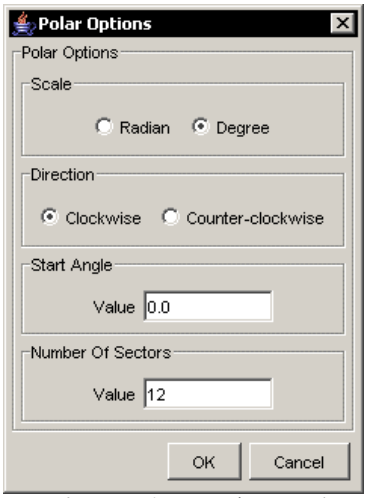
分散オプションダイアログ

"Max Y Cut Off Point"オプションを使用すると、分散座標の Y 点の最大値を指定できます。このしきい値を超える座標はプロットされません。接続線は、しきい値の端まで描画され、次のデータポイントまで続きます。

"Combo Line"オプションを使用すると、接続線を階段状の線として描画することを指定できます。また、階段状の線の割合も指定できます。

7.9.12 ポーラーチャート

ポーラーチャートでは次のダイアログが表示されます。

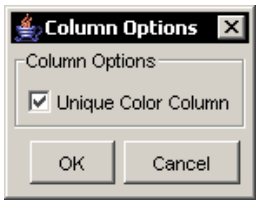


ポーラオプションダイアログ

- **Scale（角度の単位）：** 入力データのデータポイントの角度(*θ*) が Radian（ラジアン）か Degree（ディグリー）かを指定します。チャートは常に 0 から 360 度の角度で表示します。入力データはラジアンで（計算されて）もチャートのディスプレイはディグリー（度数法）で表示されます。
- **Direction（方向）：** 円プロットを Clockwise（時計回り）で描くか Counter-clockwise（時計と逆回り）で描くかを指定します。
- **Start Angle（開始アングル）：** ポーラーチャートの頂点はデフォルトでは 0 度（ディグリー）です。このオプションで、プロットの頂点の角度を指定することができます。ここでの角度の単位は、ディグリーかラジアンのどちらか Scale（角度の単位）で選んだ方になります。
- **Number of Sectors（セクターの数）：** チャートで表示したいセクターの数を指定します。セクターは一定の角度の間隔でポーラー軸ラインを描くことによって作られます。 デフォルトでは 4 つのセクターが表示されます。

7.9.13 シリーズ（データ系列）付きカラムチャート

シリーズ（データ系列）付きカラムチャートでは次のダイアログが表示されます。

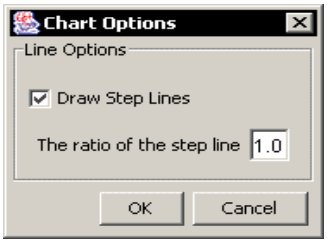


カラムオプションダイアログ

通常、カラムチャーチにデータシリーズ（系列）がある場合、各シリーズ（系列）はチャートでカテゴリーごとに色分けされています。シリーズ（系列）にかかわらず色を指定したい場合、ダイアログの'Unique Color Column' オプションを有効にします。このオプションを有効にすると、各カラムに自由に色を設定することができます。

7.9.14 2D ラインコンビネーションチャート

2D ラインコンビネーションチャートでは次のダイアログが表示されます。

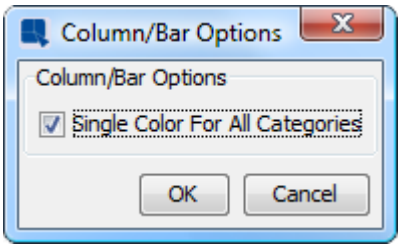


ラインコンビネーションチャートオプション

このダイアログで、ステップラインを組み合わせるかどうか指定します。また、組み合わせる場合のステップラインの比率も指定します。

7.9.15 シリーズ無しのコラム・バーチャート

シリーズ無しのコラム・バーチャートの場合は、以下のダイアログが表示されます：

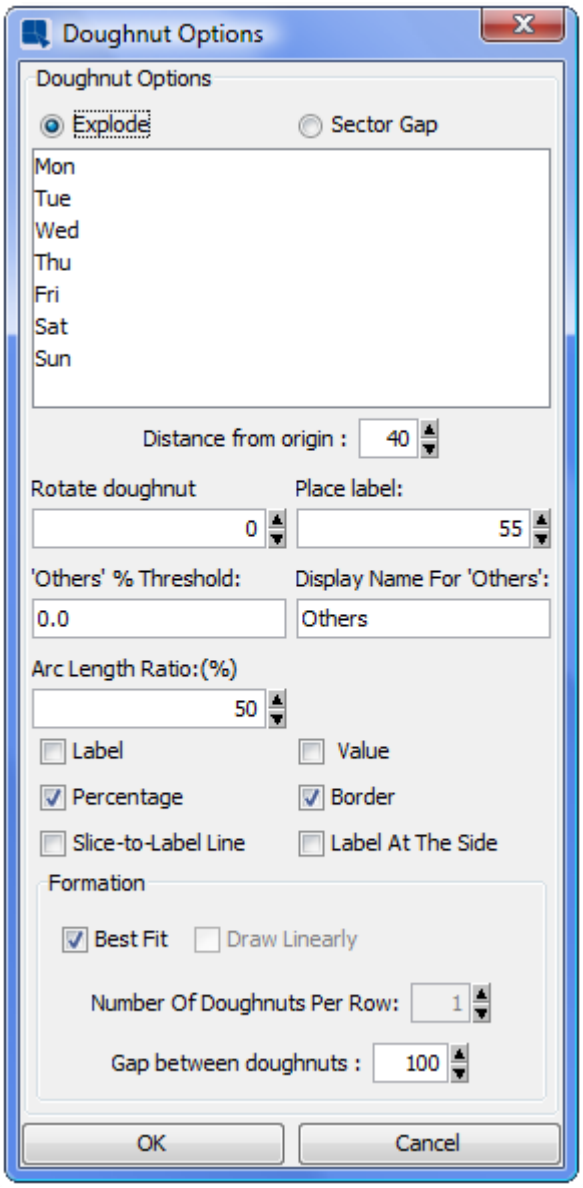


コラム・バーオプションダイアログ

通常に、コラム・バーチャートはデータシリーズがない場合は、全部のカテゴリーは一つの色になります。カテゴリー毎に色を指定する場合、ダイアログの Single Color For All Categories オプションの選択を外します。

7.9.16 ドーナツチャート

ドーナツチャートのオプションは、パイチャートとほとんど同じです。これの詳細については、Section 6.9.4 - Pie Chartsをご参照ください。



ドーナツチャートオプション

ドーナツチャートの特徴オプションとしては、アーク長比率（%）です。これで、チャート中心の穴部のサイズを指定することができます。比率は高い数字になったら、穴のサイズが小さくなる結果になります。

7.10Chart Designer in Mac OS X

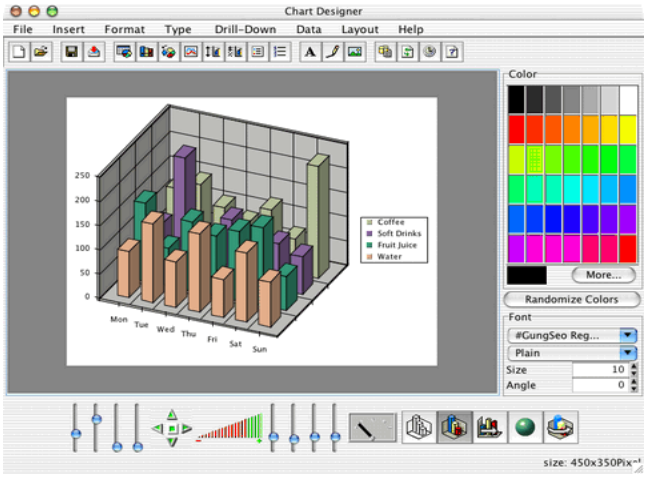


Chart Designer in OS X

When running Chart Designer in OS X most of the controls are the same as for Windows or Unix/Linux, with one exception:

Right-Click: To invoke pop-up menus, and re-size chart plot area, use CTRL+Click instead of right-click when running in OS X.

8 ドリルダウン

チャートはデータセットに含まれる情報の分析と確認を容易にする、優れたツールです。しかしデータ量が非常に多い場合や非常に複雑なデータの場合は、単独のチャートでの描画が困難になる可能性があります。このようなデータをよりわかりやすく表示するためには、大まかなデータのみを表示するトップレベルチャートの作成が有効です。さらにトップレベルチャートの任意のデータポイント、または基本データを表示するチャート凡例内の対応するラベルをクリックすると、該当部分の詳細なデータが表示されます。これがドリルダウンのコンセプトです。ドリルダウンを作成するには、まず個別のチャートを作成し、次に適切なデータポイント、または凡例内の対応するラベルをハイパーリンクします。ただし、この方法が適切なのはデータポイントの数が少ない場合に限られます。例えばトップレベルチャートに 20 のデータポイントがあるとします。1 レベルのドリルダウンのためには各データポイントに対応する 20 のチャートを作成する必要があります。このサブレベルチャートがそれぞれ 15 のデータポイントを持つとすると、さらに次のレベルのドリルダウンのために 20x15 個のチャート、即ち 300 個のチャートを作成する必要が出てきます。このような問題を避けるため、EspressChart ではドリルダウンの各レベルにつき 1 個のチャートのみを作成できる数種類のビルトインのドリルダウンメカニズムが用意されています。すべてのドリルダウンのオプションはチャートデザイナーの Drill-Down メニューで選択できます。

8.1 データドリルダウン

データドリルダウンを使用すると、一つのデータソースに基づく情報をグループ化して表示できます。データドリルダウンはあらゆるデータソースを扱えるという利点があります。しかし、すべてのレベルのドリルダウンが入力データの同じバリューカラムを共有するため、曖昧な関連付けがされた情報は表示できません。

例えば、以下のようなチャートデータがあるとします。

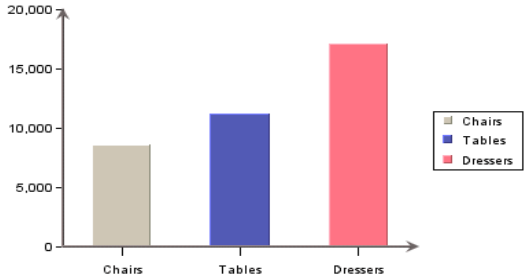
Category	Product	Sales
Chairs	Elm Arm Chair	$8,216
Chairs	Pine Side Chair	$7,611
Chairs	Redwood Arm Chair	$8,625
Tables	Elm Round Table	$10,241
Tables	Pine Oval Table	$9,663
Tables	Oak Oval Table	$11,261
Dressers	Oak Single Dresser	$16,442
Dressers	Elm Double Dresser	$17,148

このデータからは、各製品カテゴリの売上の合計を表示するトップレベルチャートを作成し、さらに各カテゴリ内の製品の個別の売上を表示するサブレベルのチャートを

作成できます。ドリルダウンのレベル数は、入力データにあるグループ分けの数によって決定されます。

8.1.1 データドリルダウンの追加

データドリルダウンのレイヤーをチャートに追加するには、まずトップレベルチャートを作成する必要があります。上記の例の場合、カテゴリ軸に Category を、バリュー軸に Sales をそれぞれマッピングしたチャートを作成します。カラムチャートの場合、以下のように表示されます。



トップレベルチャート

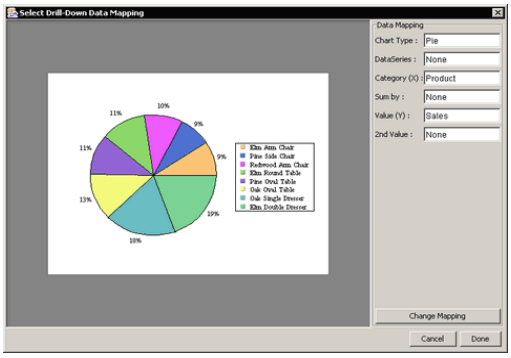
ドリルダウンのレイヤーを追加するには、Drill-Down メニューから Add を選択します。ダイアログが表示されますので、バリュー軸に使用する集計方法を指定します。例えば、sum を選択するとトップレベルチャートに各カテゴリの合計が表示されます。選択できる集計機能は sum（合計）、minimum（最大値）、maximum（最小値）average（平均値）、count（出現回数）です。

集計方法ダイアログ

使用する集計方法を選択し、OK をクリックします。新しいダイアログが表示されますので、ドリルダウンチャートの名前を入力します。この名前はドリルダウンプロセスで作成されるチャートテンプレートに割り当てられます。すべてのサブレベルのドリルダウンテンプレートは/drilltemplates/ディレクトリに保存されます

ドリルダウン名ダイアログ

名前を入力し、OK をクリックします。新しいダイアログが開きますので、サブレベルチャートのチャートタイプとデータマッピングを指定します。



データドリルダウンのマッピングダイアログ

このダイアログでは通常のデータマッピングのオプションと同様のオプションが使用できますが、一番目のオプションでチャートタイプを選択できる点が異なっています。データドリルダウンではカラム、バー、ライン、スタックカラム、スタックバー、パイ、エリア、オーバーレイ、レーダー、ダイアルの各チャートタイプが使用できます。データマッピングオプションは選択したチャートタイプによって変更されます。

マッピングオプションを指定したら'Done'をクリックしてチャートデザイナーに戻り、サブレベルチャートのカスタマイズと修正を行います。もう一度 Drill-Down メニューから'Add'をクリックすると、ドリルダウンにさらにレイヤーを追加できます。

任意のドリルダウンチャートに移動するには、Drill-Down メニューから'Previous'または'Next'を選択するか、データポイントをダブルクリックします。'Previous'を選択すると一つ上のレベルへ移動し、'Next'を選択すると左端にあるカテゴリのデータをドリルダウンして一つ下のレベルへ移動します。また Drill-Down メニューから'Go To Top Level'を選択すると、どのドリルダウンチャートからでもトップレベルチャートへ移動できます。ドリルダウンチャートはトップレベルチャートと同様にカスタマイズでき、色の割り当ておよびタイトル、軸ラベル、背景イメージの追加などが行えます。レベルに変更を加えた後に他のレベルへ移動しようとすると、チャートを保存するダイアログが表示されてます。ドリルダウンレベルの属性を保存したい場合は'yes'をクリックします。'yes'をクリックしない場合、加えた変更が破棄されます。

2 つのドリルダウンチャートの間に、もう一つのドリルダウンチャートを挿入することもできます。これを行うには、2 つのうち上位レベルのドリルダウンチャートに移動して'Add'を選択します。作成したドリルダウンチャートがその直前の 2 つのドリルダウンチャートの間に挿入されます。

ドリルダウンから一つのレベルを削除するには、削除したいレベルに移動して Drill-Down メニューから'Remove This'を選択します。ドリルダウンチャート全体を削除する場合は'Remove All'を選択します。Drill-Down メニューから'Previous'を選択すると、ドリルダウンチャートを一段上のレベルに移動できます。右クリックでポップアップメニューから'Back'を選択しても同様の操作ができます。'Next'を選択すると、ドリル

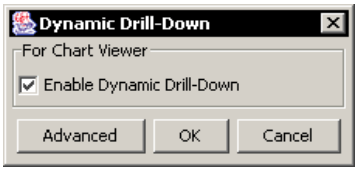
ダウンチャートを一段下のレベルに移動できます。チャート内のアイテムをダブルクリックした場合も同様の操作が可能です。

ドリルダウンチャートの全レベルの作成と編集が終了したら、トップレベルチャートに移動して File メニューから'Save'または'Save As'を選択し、チャートを保存します。

8.2 ダイナミックデータドリルダウン

データドリルダウンでは、通常はデザインを行う時にマッピングとドリルオプションが固定されます。オプションとしてダイナミックドリルダウンを使用すると、エンドユーザによるマッピングの選択が可能になります。この場合、デザイン時に設定されるのはトップレベルチャートと集計方法のみです。

ダイナミックドリルダウンを使用したチャートを作成するには、最初にトップレベルチャートとして使用するチャートを作成します（例えば、前回と同じトップレベルチャートを作成します）。次に Drill-Down メニューから'Dynamic'を選択すると、ダイナミックドリルダウンを有効にするダイアログが表示されます。



ダイナミックドリルダウン有効化ダイアログ

'Enable Dynamic Drill-Down'ボックスをチェックすると新しいダイアログが開きますので、集計方法を選択します。

集計方法ダイアログ

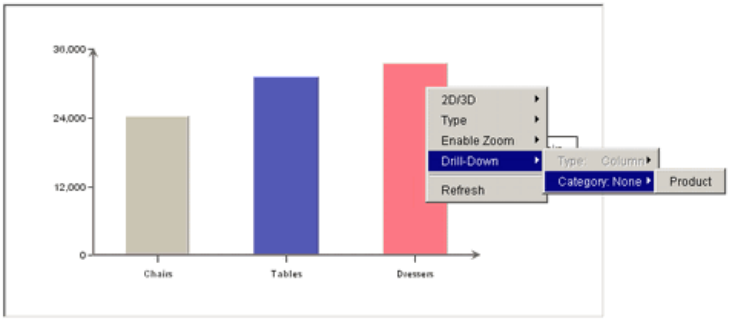
このダイアログで指定できるオプションは通常のデータドリルダウンと同じです。使用する集計方法を指定して'OK'をクリックすると新しいダイアログが開きますので、サブレベルのチャートで使用するテンプレートを指定します。

ダイナミックドリルダウンテンプレートの選択ダイアログ

テンプレートを使用しない場合、サブレベルチャートはデフォルトの外観プロパティで生成されます。すべてのオプションを選択するとチャートの外観が変更され、集計

されたデータがバリュー軸上に表示されます。オプションを変更するには、Drill-Down メニューから'Dynamic'をもう一度選択して'Advanced'ボタンをクリックします。

ダイナミックドリルダウンチャートを表示するには、チャートビューワアプレットの使用が必要です。チャートビューワ内では、デスクトップを右クリックするとポップアップメニューが表示され、次のドリルダウンの設定を行うことができます。ダイナミックドリルダウンが有効で、かつチャートに描画されていない未使用のフィールド（カラム）がある場合、ポップアップメニューに'Drill-Down'が表示されます。この'Drill-Down'を選択すると、次のドリルダウンの現在の設定を表示できます。'Category'に'None'が表示されている場合、次のドリルダウンチャートはまだ設定されていません。次レベルのドリルダウンチャートを設定するには'Category'を選択します（'Category'を選択すると'Type'、'Series'および'SumBy'の選択が可能になります）。ドリルダウンチャートを作成したら、チャートビューワ内でドリルダウンチャートを移動できます。データポイントを左クリックするとレベルが一段下がり、右クリックしてポップアップメニューから'Back'を選択するとレベルが一段上がります。前述の手順を繰り返すことにより、次のレベルにもう一つのドリルダウンチャートを作成することもできます。



チャートビューワにおけるダイナミックデータドリルダウン

8.3 パラメータドリルダウン

EspressChart で使用できる 3 番目のドリルダウンは、パラメータドリルダウンです。パラメータドリルダウンはドリルダウンのレベル間のデータを任意の方法で関連付けることができるため、最も柔軟性の高いドリルダウンが可能です。パラメータドリルダウンでは各レベルに同じバリューエレメントを使用する必要はありません。各チャートレベルはパラメータ化されたクエリ機能によって関連付けられます。パラメータドリルダウンはクエリを使用するため、サブレベルチャートのデータソースはデータベースであることが必要です。

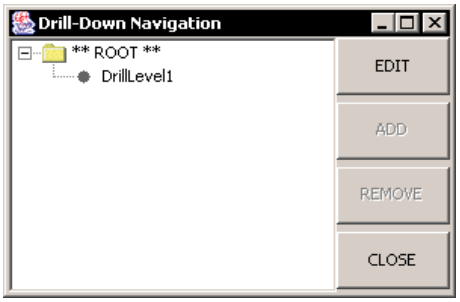
例として上記のデータを使用します。例えば常に売上を表示する代わりに、（上記と同様に）トップレベルチャートにカテゴリごとの売上の集計を表示させたいとします。ただし次のレベルでは、各製品の売上数を表示させます。さらにここから、地域ごと

の各製品の在庫レベルを表示させるとします。このような表示はすべてパラメータドリルダウンによって可能になります。

パラメータドリルダウンでは、ドリルダウンするためにクリックしたエレメントのカテゴリー値はパラメータ値としてサブレベルチャートに引き渡されます。例えば'Chairs'カラムをクリックした場合は'Chairs'という値がクエリに引き渡されます。これにより、データベースでカテゴリ名によってフィルタリングされた結果がすべて引き出されます。

8.3.1 パラメータドリルダウンの追加

パラメータドリルダウンのレイヤーを追加または編集するには、Drill-Down メニューから'Parameter Drill Down'を選択します。ナビゲーションウィンドウが立ち上がり、ドリルダウンの各レベルが表示されます。

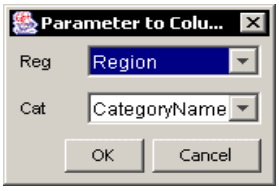


パラメータドリルダウンのナビゲーションウィンドウ

ナビゲーションウィンドウの左側にドリルダウンレベルの階層が表示されます。"ROOT"ノードはトップレベルチャートです。現在編集しているレベルには"**"のマークが表示されます。別のレベルを編集するには、ウィンドウ右側の'EDIT'ボタンをクリックします。選択したチャートが Designer で開きますので、ここで変更を行います。ドリルダウンレベルのレベルを削除するには、ナビゲーションウィンドウで削除したいレベルを選択して'REMOVE'ボタンをクリックします。

新規のドリルダウンレベルを追加するには、下にレイヤーを追加したいチャートを選択して'ADD'をクリックします（トップレベルチャートしかない場合は"ROOT"ノードのみが表示されます）。ダイアログが表示されますので、新規のチャートを作成するか、既存のチャートをドリルダウンレイヤーに使用するかを選択します。どのような既存のチャートでも使用は可能ですが、ドリルダウンレイヤーに使用するチャートのデータベースはパラメータ化されたクエリである必要があります。新規チャートの作成を選択した場合は Data Source Manager が開きますので、チャートのデータソースを選択し、チャートウィザードのステップを実行します。

次に、トップレベルチャートのカテゴリとシリーズカラムをサブレベルチャートのクエリパラメータにマッピングします。ドリルダウンレイヤーに使用する既存のチャートを選択する時、またはドリルダウンレイヤーの新規チャートのデータソースを選択する時にダイアログが表示されますので、ここでフィールドのマッピングを行います。



パラメータのマッピングウィンドウ

ドロップダウンメニューで選択できるオプションはデータタイプによって異なります。例えばパラメータ化されたクエリがストリングをパラメータとして使用する場合、ストリングデータを含むフィールドのみがマッピングされます。トップレベルチャートからサブレベルチャートに引き渡されるデータのタイプに注意してください。カテゴリやシリーズが適切なデータタイプでない場合、またはサブレベルに引き渡すための十分なフィールドがない場合は、ドリルダウンが正しく行われません。

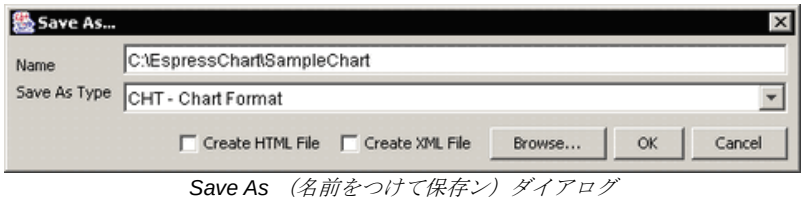
パラメータマッピングを正しく指定したら、ドリルダウンレベルの表示名を指定するダイアログが表示されます。表示名を指定すると Designer にサブレベルチャートが表示され、カスタマイズが可能になります。ドリルダウンのレイヤー間を移動するにはナビゲーションウィンドウを使用します。またはデータドリルダウンと同様に、データポイントをクリックして一つ下のレベルに移動するか、右クリックしてポップアップメニューから'Back'を選択し、一つ上のレベルに移動します。

9 チャートの保存とエクスポート

チャートのデザインが終ったら、チャートとチャートテンプレートの定義、チャートとチャートテンプレートの XML 定義を保存します。また、チャートのスタティック（静的）イメージのエクスポートを作成したり、埋め込みチャートビューワアプレットで HTML ページを作成することもできます。

9.1 チャートの保存

カレントチャートの保存には Fine（ファイル）メニューの'Save' （保存）もしくは 'Save As' （名前を付けて保存）を選択するか、ツールバーの'Save' （保存）ボタンをクリックします。チャートを一度も保存していない場合に保存するとき、または 'Save As'（名前を付けて保存）を選択した場合は次のようなダイアログが表示されます。



Save As （名前をつけて保存ン）ダイアログ

最初のオプションは保存するチャートの名前とファイルパスを指定します。ダイアログの下方にある'Browse' （ブラウズ）ボタンをクリックして、ファイルバスをブラウズすることもできます。次のオプションはチャートを保存するのに使用するフォーマットを指定します。3 章の詳細にあるように、保存されるチャート定義には 2 つの基本的な方法があります。

Chart format: (チャートフォーマット): チャートファイルは filename.cht という名前のバイナリファイルでチャートを保存します。チャートファイルはチャートの定義（タイプ、次元など）とチャートの作成に使用したデータを保存します。

Template format（テンプレートフォーマット）: テンプレートファイルは filename.tpl という名前のバイナリファイルでチャートを保存します。テンプレートファイルはチャートの定義のみ保存し、チャートとデータは保存しません。従って、テンプレートファイルがオープンすると、データをリトリーブするために、常にオリジナルデータソースに接続しようとします

使用したいフォーマットを選んだら、'OK' をクリックしてチャートを保存します。

9.1.1 テンプレートの利用

チャートテンプレートはチャートの定義だけが保存された特別なフォーマットです。チャートの定義とデータの両方が保存されるチャートフォーマット(.cht フォーマット)と違って、テンプレートは定義（例えば、チャートの属性やそれぞれのコンポー

ネントのレイアウトなど）とデータソース情報のみ保存しています。つまり、テンプレートはチャート作成に使用したデータのロケーションまたは接続情報は保存していますが、実際のデータはファイルに保存していません。テンプレートはデータソースが存在しなかった場合にもオープンできるように、バックアップデータとして 10 レコードだけ保持しています。

テンプレートファイルは 2 つの目的で利用されます。ひとつ目は最新データをユーザが見ることができるように、テンプレートファイルがオープンされ、ビューされたり、エクスポートされたりします。EspressChart はファイルに指定されたデータソースにチャートのデータをリトリーブしに行きます。オリジナルデータソースがアクセス不可能な場合は意図するような利用ができません。2 つ目のテンプレートの利用方法は、チャートの属性を他のチャートの属性に適用するというものです。この場合は、が適用されているチャートは、テンプレートの外観のプロパティ（色、フォント、チャートコンポーネントのサイズ、凡例など）を持つようになります。これにより一貫したルック＆フィールのチャートを構築することができます。

JDBC コードまたは他のデータソースを使用し、API を使ってプログラムでチャートを作成している場合、2 つ目の利用方法はとても有益です。コードで外観のプロパティ定義していなくても、またデフォルトチャートのプロパティに依存しなくても、あらかじめ定義されたテンプレートを使って、作成されているチャートのルック&フィールを制御することができます。API のテンプレート適用についての詳細は 11 章のセクション 11.5.2 を参照してください。

File （ファイル）メニューの'Apply Template' （テンプレートの適用）を選択して、チャートデザイナーのテンプレートを適用することができます。使用したいテンプレートファイルを指定するダイアログが立ち上がります。

テンプレート適用ダイアログ

ダイアログでは、テンプレートファイルとそのロケーションが指定できます。'Browse' （ブラウズ）ボタンをクリックして、ファイルをブラウズすることができます。

注意:チャートと適用されているテンプレートのサイズが違うとき（例えば、チャートキャンバスのサイズ）、結果として現れるチャートは正しく出力されないかもしれません。これはテンプレートとテンプレートが適用されるチャート間でテキストサイズが変更されないために起こります。他のコンポーネントが新しいキャンバスのサイズに調整されるのに対して、フォントは調整されないのです。一貫した外観を保つためには、テンプレートのサイズを適応されてるチャートのサイズに近づけることです。

テンプレートに定義されているハイパーリンク、フローティングライン、フローティングテキスト、軸目盛りは、そのまま持ち越されます。必要ならば、チャートのこれらを再定義しなければならないかもしれません。チャートタイプと次元はテンプレートによって変更することはできません。例えば、テンプレートが 2D（2 次元）チャートの場合、3D（3 次元）チャートは 2D（2 次元）には変更されません。同じように、バーチャートのテンプレートが適用されても、パイチャート（円グラフ）はパイチャート（円グラフ）のままです。テンプレートを適用したとき、チャートタイプは変更されなくても、他のチャートタイプのテンプレートから外観のいくつかのプロパティは上手く変換されません。要領としては、同じタイプ同じ次元のテンプレートをチャートに適用するようにしてください。

9.1.2 XML テンプレートの保存

2 つのバイナリチャート保存形式に加え、XML フォーマットのチャート定義を保存することもできます。このオプションによって、チャートデザイナーや API 以外でチャートプロパティを変更するのに使えるテキストベースのチャートテンプレートを保存できます。

チャートを保存するときに XML ファイルを作成するには、チャートを保存するときの Save As（名前を付けて保存） ダイアログで、'Create XML File' （XML ファイルの作成）ボックスをチェックします。これにより、チャート用の XML ファイルが作成されます。XML ファイルは、cht フォーマットではなく.tpl フォーマットで表されることに注意してください。

XML ファイルで変更できる属性のリストは付録 B を参照してください。 .

9.1.3 ビューワページの作成

指定できるその他のオプションはチャートをビューする HTML ページを作成したいかどうかです。HTML ページは埋め込みチャートビューワアプレットが入っていてエンドユーザがチャートをビューしたり、操作したりすることができます。チャートビューワの機能に関しては 10 章に詳細が記述されています。

チャートを保存するとき HTML ページを作成するには、チャートを保存するときの Save As（名前を付けて保存） ダイアログで、'Create HTML File' （HTML ファイルの作成）ボックスにチェックを入れます。これにより EspressChart インストレーション の/html/ ディレクトリにチャートファイルと同じ名前で HTML ページが作成されます。ファイルが書き込まれる前に使用したいビューワーのバージョンを選択するプロンプトが出されます。

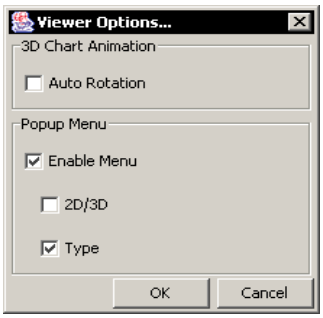
ビューワ選択ダイアログ

EspressChart は、AWT および Viewer(ビューワ)の Swing（スウィング） バージョンをサポートします。でも利用可能です。注意: Viewer を使う場合、クライアントには Java プラグインが必要になります。

HTML ページを作成したら、チャートをビューするためにそのページにブラウザを向けます。注:チャートを表示させるために、http プロトコルで HTML ページをロードする必要があります。ファイルプロトコルでロード（例えば、ファイルにブラウズしてオープンする）を試みた場合、上手くいかないでしょう。ロードが正しく行われるようにするには、EspressChart が Web サーバーにインストールされていなくてはならず、ページには http(例 http:// machinename:port/EspressChart/html/yourfile.html)経由でアクセスしなくてはならないということです

9.1.3.1 ビューワのオプション

Chart Designer(チャートデザイナ)で設定できるチャートビューワのオプションがいくつかあります。それらはチャートに保存されるプロパティであり、チャートがビューワで表示されると、プロパティに設定されているように表されます。ビューワのオプションを変更するには Format（フォーマット）メニューの Viewer Option（ビューワ・オプション）を選びます。すると、次のようなダイアログが表示されます。



Viewer Options（ビューワオプション）ダイアログ

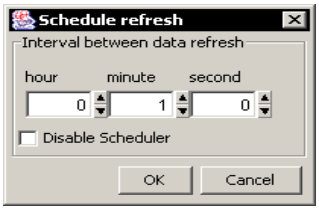
オプションには以下に示すものがあります。

- **Auto Rotation（自動回転）：** 3D チャートを自動的に回転させることが可能です。アプレットがロードされるとチャートが回転し始めます。その回転はナビゲーションパネルのボタンを使って停止させることができます。

- **Enable Menu（メニューの有効化）：**ユーザがチャートビューワでチャートをビューしているときのポップアップメニューを有効にします。

- **2D/3D:**ポップアップメニューのデメンジョントグル（2D と 3D の切り替え）を有効にします。もしこれがオフの設定なら、ユーザはチャートの次元（2D/3D）を切り替えることができません。
- **Type（タイプ）：**ポップアップメニューのサブメニューの Type（タイプ）を有効にします。もしこれがオフの設定なら、ユーザはチャートタイプを変更することができません。

.cht ファイルでは、リフレッシュする時間をスケジュールすることができます。チャートビューワでチャートをビューしているときに、データを定期的にリフレッシュするようになります。リフレッシュの時間をスケジュールするには、Data（データ）メニューから Schedule refresh（リフレッシュのスケジュール）を選びます。すると、ダイアログが表示され、そこでスケジュールオプションを設定することができます。

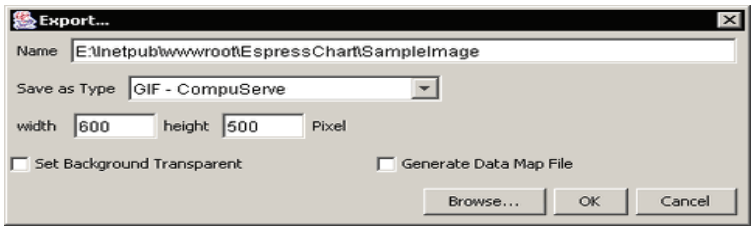


Schedule Refresh（リフレッシュのスケジュール）ダイアログ

このダイアログでリフレッシュのインターバルの時間、分、秒の間隔を設定します。Disable Scheduler（スケジューラの無効）にチェックを入れた場合、スケジューラは作動しません 。

9.2 チャートのエクスポート

チャートデザイナでは、チャートの静止画像エクスポートを作成することができます。カレントのチャートをエクスポートするには File（ファイル）メニューの Export（エクスポート）を選ぶか、ツールボタンの Export（エクスポート）をクリックします。すると、ダイアログボックスが表示され、ファイルを作成するためのオプションが指定できます。



Export Chart（チャートのエクスポート）ダイアログ

最初のオプションは作成するファイルの名前とファイルパスの指定です。ダイアログの下方にある Browse（ブラウズ）ボタンをクリックして、ファイルパスをブラウズすることができます。次のオプションはどのタイプのエクスポート（画像）にするか選択します。オプションには次のものがあります。

- **GIF:** EspressChart は 2 つの圧縮方法（RLE または LZW）のどちらかを使って GIF イメージを作成します。LZW はより早く、より小さいファイルを生成しますが、LZW の使用が特許によって制限されています。LZW の圧縮を解凍するには、まず Unisys からライセンスを取得しなければなりません。デフォルトでは、GIF ファイルは RLE の圧縮法を使って作成されています。

  GIF イメージでは、イメージの背景の透明化（トランスペアレント）の設定を指定できます。Set Background Transparent（背景の透明化設定）チェックボックスをクリックすることによって設定できます。

  *注 - RLE の GIF エンコーディングには、一般に知られている制限があります。現在のところ、チャートがアンチエイリアシング（アンチエイリアシング機能または Java2D 回転テキスト機能）を使うと、RLE GIF エクスポートは不正なチャートを作ってしまいます。 これを避けるためには RLE の GIF をエクスポートするときにアンチエイリアシングをオフにするか、異なるイメージフォーマットにエクスポートするかのどちらかです。

- **JPEG:** JPEG はもうひとつの一般的なイメージフォーマットです。GIF より解像度が高く、特許による制限はありません。JPEG ファイルが作成されるとき、ファイルの画質や圧縮を指定することができます。画質が高ければ高いほど、ファイルは大きくなります。
  エクスポートフォーマットに JPEG を選択した場合、OK をクリックした後画質を指定するようになっています。画質が高ければ高いほど作成されるファイルのサイズは大きくなります。JPEG でエクスポートする場合、画質が低いと期待するような結果はほとんど得られませんので、高画質を指定することが推奨されています。

- **PNG:** PNG はあまり一般的ではありませんが、ほとんどのブラウザで表示できます。高画質で JPEG よりファイルが小さくてすみます。
  エクスポートフォーマットに PNG を選んだ場合、OK をクリックした後、作成されたイメージ画像の圧縮を指定することができます。このフォーマットには 3 つの圧縮オプションがあります。default compression（デフォルト圧縮）、maximum compression（最大圧縮）、no compression（圧縮なし）。圧縮度が高ければ高いほど、作成されるファイルは小さくなりますが、ファイルの作成に時間がかかります。
- **SVG:** SVG (Scalable Vector Graphics) は比較的新しいイメージフォーマットで、XML ベース・テキスト・フォーマットにベクトルとしてイメージを保存

しています。通常このイメージをビューするには、ブラウザ・プラグインが必要です。

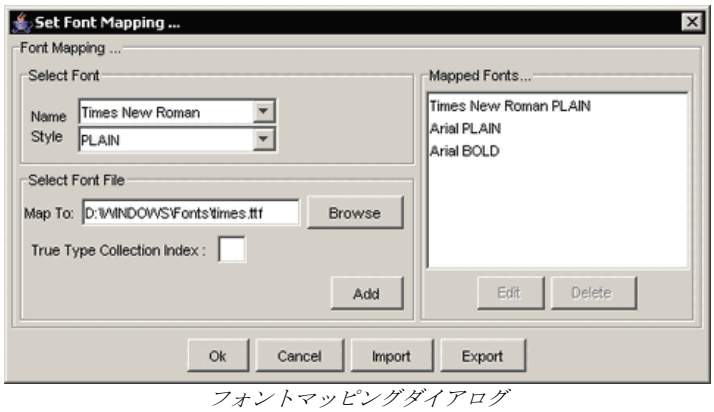
- **SWF:** SWF はマクロメディア ・フラッシュ・ファイルです。フラッシュ・フォーマットはベクトルがベースです。また、エクスポートした後でもチャートのサイズを変更することができます。フラッシュは高画質印刷もでき、ファイルサイズも小さいファイルが作成されます。
- **BMP:** BMP は Windows ビットマップ・フォーマットです。
- **WMF:** WMF は Windows メタファイル・フォーマットです。MS オフィスドキュメントのインポート/エクスポートに使用することができます。
- **PDF:** Adobe ポータブル・ドキュメント・フォーマットでチャートを作成します。チャートは 1 ページの PDF ドキュメントとして作成されます。
- **XML:** チャートデータがデータとして保存される XML データファイルを作成します。XML チャートアトリビュート（属性）ファイルは作成せず、データのみが XML フォーマットに書き込まれます。
- **TXT:**チャートデータがデータとして保存されるテキストデータファイルを作成します。

さらに、もしチャートにハイパーリンクが含まれている場合は、作成されるイメージと共にマップファイルをエクスポートするように選択することができます。マップファイルはチャート作成に必要な HTML イメージマップとリンクを保持しています。マップファイルはイメージファイルと同じロケーションに作成され、同じ名前のファイルになります。マップファイルを作成するには、エクスポートする前に、Generate Data Map File（データマップファイルの作成）ボックスにチェックを入れます。

9.2.1 PDF フォントマッピング

EspressChart ではチャートフォントを使用することができます。よく使われるイメージフォント（GIF/JPEG/PNG）フォントは生成されたイメージで書かれていますが、PDF をチャートで使う場合は、マニアルで ttf (true type font)か、 .ttc (true type collection)か、 .pfb か .afm ファイルであるかを指定しなければなりません。

PDF エクスポートのフォントマッピングを設定するには Option（オプション）メニュー'Font Mapping（フォントマッピング）' を選択します。フォントファイルを指定するダイアログが立ち上がります。



フォントマッピングダイアログ

各フォントとスタイルの組み合わせでそのフォントに対して.ttf、 .ttc、 .pfb、.afm ファイルを指定します。フォントファイルのフルパス名をタイプするか、またはブラウズするかで指定します。.ttc ファイルを使用する場合、指定のボックスにフォントインデックスを指定する必要があります。（.ttc ファイルはひとつ以上のフォントを含みます。）ファイルを指定したら、'Add' （追加）ボタンをクリックしてリストへのマッピングを保存します。リストの中から選択してボタンをクリックすることにより既存のマッピングを編集したり削除したりできます。

9.2.1.1 PDF フォントマッピングのインポート/エクスポート

インポート/エクスポート機能を使って、あるチャートからもうひとつのチャートへフォントマッピングを渡すことができます。フォントマッピングダイアログの 'Export' （エクスポート）ボタンをクリックするとフォントマッピングをエクスポートすることができます。これによりダイログボックスが立ち上がり、ファイル名を指定することができます。フォントマッピングは XML ファイルとして保存されます。ダイアログから'Import'（インポート）オプションを選ぶことによってフォントマッピング XML ファイルをロードすることができます。これによりダイアログボックスが立ち上がりインポートしたい XML ファイルを指定することができます。Click on 'OK' をクリックして XML ファイルに保存されたマッピングはカレントチャートに適用されます。

10 チャートビューワ

チャートデザイナーで作成したすべてのチャートは、他の書類にペースト可能な各種のフォーマットで保存できます。保存可能なフォーマットは BMP、JPG、PNG、PDF、SVG、SWF、WMF、GIF です。さらにオプションとして、.cht、.tpl、.xml の各形式でチャートを保存することもできます。

チャートビューワはウェブブラウザでチャートを動的に表示および操作できるアプレットです。チャートビューワはチャートデザイナーまたはチャート API が出力した.cht または.tpl 形式のファイルを読み込み、チャートを表示します。このデータファイルはサイズが小さいので、チャート画像をウェブ上で配布するのに適しています。データは暗号化されて EspressManager からチャートビューワに転送されますので、ある程度のセキュリティが確保されます。

チャートビューワアプレットは.cht ファイルをインタラクティブに表示することができます。チャートビューワは元データに変更を加えることなくチャートの表示と操作を行うことができます。

チャートビューワは.tpl フォーマットのファイルも表示できます。.tpl ファイルを含むウェブページをブラウザで表示すると、チャートビューワによって自動的に最新のデータが取得されます。これによって、一つのチャートテンプレートを使用して分刻みで最新のチャートをリアルタイム供給することができます。

チャートビューワ内では、チャート、凡例、タイトル、レーベルをオブジェクトとしてドラッグし、配置することができます。チャートやデータポイント上のドリルダウン、およびデータのシリーズをリサイズすることも可能です。3D チャートの場合、ナビゲーションパネルを使用して各方向へのパン、ズーム、回転および平行移動が可能です。また、x、y、z 軸の個別のスケーリング、厚さ比の調整、リアルタイム 3D アニメーションなども簡単に実行できます。ビルトインのコールバックメカニズムを使用すると、データエレメントをクリックすることによって元データを表示したり、関連する URL へのジャンプを行うことができます。チャートビューワは Pure Java で記述されており、Java をサポートするすべてのプラットフォーム上で使用できます。また、チャートビューワはスケジュール化されたリフレッシュとパラメータサービングをサポートしており、Designer で指定した時間ごとにチャートのデータをアップデートしたり、ロード時にチャートのパラメータを供給することができます。

ウェブページに.cht または.tpl 形式のファイルを埋め込むには、以下のシンタックスを使用します：

```
<applet codebase = ".." code =
"quadbase.chartviewer.Viewer.class"width = 640 height = 480>
<PARAM name = "filename" value = "yourchart.cht">
</applet>
```

"filename"パラメータはチャートデータを含むファイルのファイル名を指定します。リモートデータファイルにアクセスする場合は、ファイル名の前に http://を加えます。チャートビューワでチャートを表示する場合、チャート形式（.cht）で保存されたチャートはこのファイルに保存されたデータを使用してチャートを描画します。

テンプレート形式（.tpl）で保存されたチャートは、チャートビューワがデータベースやデータファイルから動的にデータを取得するために使用します。データの取得先は、チャートデザイナーでテンプレートを作成する時に指定したチャートのデータソースによって決定されます（データベース名、ユーザ名、パスワード等はすべて.tpl ファイルの中に保存されます）。

10.1 チャートビューワパラメータ

データやチャート表示のコントロール情報は、チャートデザイナーを使用せずにパラメータを通じてチャートビューワアプレットに引き渡すことが可能です。HTML コードの中で、チャートビューワを使用して直接データファイルを表示したり、データ列形式のデータや他のコントロール情報を直接引き渡すことができます。パラメータが true/false タイプの場合、デフォルトではパラメータは ture になります。以下にパラメータのリストを示します：

チャートパラメータ（2D チャートと 3D チャートに共通）：

- **mainTitle:** チャートのメインタイトル
- **xTitle:** x 軸タイトル
- **yTitle:** y 軸タイトル
- **zTitle:** z 軸タイトル
- **RefreshInterval:**リフレッシュ間隔（秒単位）のスケジュール
- **DragLegend:** false の場合、凡例は移動不可
- **DragChart:** false の場合、チャートの移動やサイズ変更は不可
- **ShowDataHint:** false の場合、チャートデータを左クリックしてもデータ情報ボックスは表示されない
- **ShowLinkHint:** false の場合、チャートデータを右クリックしてもハイパーリンク情報ボックスは表示されない
- **DataHintBgColor:** データ情報ボックスの背景色を設定
- **LinkHintBgColor:** ハイパーリンク情報ボックスの背景色を設定
- **DataHintFontColor:** データ情報ボックスのフォント色を設定
- **LinkHintFontColor:** ハイパーリンク情報ボックスのフォント色を設定
- **DataHintFont:** データ情報ボックスのフォントを設定
- **LinkHintFont:** リンク情報ボックスのフォントを設定
- **DataHintOffsetX:** データ情報ボックスの x オフセットを設定
- **DataHintOffsetY:** データ情報ボックスの y オフセットを設定
- **LinkHintOffsetX:** リンク情報ボックスの x オフセットを設定

- **LinkHintOffsetY:** リンク情報ボックスの y オフセットを設定
- **Printing:** false の場合、ブラウザで CTRL-P または CTRL-J によるチャートのエクスポートは無効
- **filename:** テンプレートファイル名をチャートに適用
- **xAxisRuler:** true の場合、x 軸目盛を表示（2D チャートのみ）
- **yAxisRuler:** true の場合、y 軸目盛を表示（2D チャートのみ）
- **sAxisRuler:** true の場合、二次軸目盛を表示（2D チャートのみ）
- **ResizeChart:** false の場合、チャートのサイズ変更は不可
- **ResizeCanvas:** false の場合、キャンバスのサイズ変更は不可
- **comm_protocol:** ファイアウォールを使用している場合に使用するプロトコル
- **comm_url:** ファイアウォールを使用している場合に EspressManager への接続に使用する URL
- **RefreshData:** false の場合、チャートデータのリフレッシュは不可
- **PopupMenu:** false の場合、ポップアップメニューは非表示
- **TypeMenu:** false の場合、ポップアップメニューで type サブメニューは非表示
- **DimensionMenu:** false の場合、ポップアップメニューで dimension サブメニューは非表示

3D チャートのみ

- **Toggle3Dpanel:** false の場合、ナビゲーションパネルの表示/非表示の切り換え不可
- **Drawmode:** 3D チャートの描画モードを設定。設定可能なモードは"Flat"（デフォルト）"WireFrame"、"Flat Border"（Flat の周囲に黒い境界線を描画）、"Gouraud"および"Gouraud Border"
- **NavColor:** ナビゲーションパネルの色を設定
- **navpanel:** false の場合、3D チャートの表示中にナビゲーションパネルの表示は不可。2D チャートの表示中は、いずれの場合でもナビゲーションパネルは表示されません。
- **GouraudButton:** false の場合、ナビゲーションパネル内の gouraud shading ボタンは非表示
- **AnimateButton:** false の場合、ナビゲーションパネル内の animation speed control は非表示
- **SpeedControlButton:** false の場合、ナビゲーションパネル内の speed control ボタンは非表示

データ入力パラメータ

- **sourceDB:** チャート生成のためのデータベース情報の設定
- **sourceData:** チャート生成のためのデータ情報の設定
- **sourceFile:** チャート生成のためのデータファイル情報の設定

- **datamap:** チャートのカラムマッピングを設定
- **TransposeData:**チャート生成に使用される前に置き換えられるデータを設定
- **chartType:** 生成されたチャートのチャートタイプを設定
- **EspressManagerUsed:** EspressManager の使用を設定
- **ParameterServer:** チャート内のデータを動的にアップデート
- **transposeData:** true の場合、データを置き換え
- **server_address:** Espress Manager 接続の IP アドレス
- **server_port_number:** Espress Manager 接続のポート番号

例：mainTitle、xTitle、yTitle、zTitle の各パラメータはチャートのメインタイトルと軸タイトルを指定し、以下のテンプレートで定義されたタイトルを上書きします：

```
<PARAM name="mainTitle" value="This is the main Title">
<PARAM name="xTitle" value="x axis title">
<PARAM name="yTitle" value="y axis title">
<PARAM name="zTitle" value="z axis title">
```

例：RefreshInterval パラメータは以下のように設定します：

```
<PARAM name="RefreshInterval" value="60">
```

上記の例では、アプレットはデータベースまたは（EspressManager を利用して）データファイルからデータを読み込み、60 秒ごとに自動的にチャートを再描画します。これはデータが頻繁に変更されるデータベースへアクセスする場合に便利な方法です。

10.2 チャートビューワのデータソースの指定

10.2.1 データベースから読み込まれたデータ

以下のサンプルは、データベースから抽出されたデータによって描画されるチャートをチャートビューワで表示するための HTML コードです。

```
<applet code = "quadbase.chartviewer.Viewer.class" width=640
height=480>
<PARAM name="sourceDB" value="jdbc:odbc:DataSource ,
                                sun.jdbc.odbc.JdbcOdbcDriver,
                                username ,
                                password ,
                                select * from products">
<PARAM name="dataMap" value="0 1 -1 3">
<PARAM name="chartType" value="3D Column">
</applet>
```

引数 dataMap は、入力されたデータから異なるカラムを使用してチャートを描画する方法を指定します。分散チャートの場合、これらは（series, x-value, y-value, z-value）となります。HLCO または High-Low チャートの場合、これらは（series, category, high, low, open, close）となります。他のすべてのチャートの場合、引数は

（series, category, sumBy, and value）となります。より詳細な情報は、チャート API リファレンスのカラムマッピングの章を参照してください。引数 `chartType` は以下のいずれかのタイプのチャートを表示するよう指定します：

- 2D カラム
- 2D バー
- 2D スタックバー
- 2D スタックカラム
- 2D エリア
- 2D スタックエリア
- 2D ライン
- 2D パイ
- 2D 分散
- 2D High-Low
- 2D HLCO
- 2D 100%カラム
- 3D サーフェス
- 2D バブル
- 2D ボックス
- 2D レーダー
- 2D ダイアル
- 2D ガント
- 2D ポーラ
- 3D カラム
- 3D バー
- 3D スタックバー
- 3D スタックカラム
- 3D エリア
- 3D スタックエリア
- 3D ライン
- 3D パイ
- 3D 分散
- 3D High-Low
- 3D HLCO
- 3D 100%カラム
- 2D オーバーレイ

10.2.2 データファイルから読み込まれたデータ

以下の HTML コードは、データファイルから読み込まれたデータを使用してチャートを描画します。

```
<applet code = "quadbase.chartviewer.Viewer.class" width=640
height=480>
<PARAM name="sourceFile" value="http://.../test.dat">
<PARAM name="dataMap" value="-1 0 -1 1">
<PARAM name="chartType" value="3D Pie">
</applet>
```

10.2.3 引数から読み込まれたデータ

チャートビューワはデータファイルやデータベースからではなく、HTML ファイル自体から直接データを読むことも可能です。

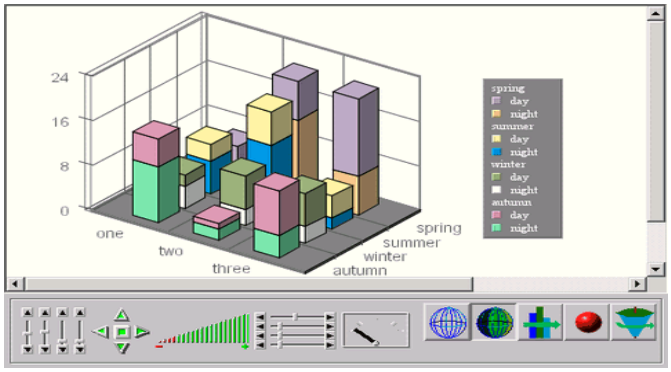
```
<applet code = "quadbase.chartviewer.Viewer.class" width=640
height=480>
<PARAM name="sourceData" value="int string int |
                                        value name, vol |
                                        10, 'John', 20 |
                                        3, 'Mary', 30 |
                                        8, 'Kevin' 3 |
                                        9, 'James', 22">
```

```
<PARAM name="dataMap" value="-1 1 -1 2">
<PARAM name="chartType" value="3D Bar">
</applet>
```

sourceData パラメータの形式はデータファイルの形式と同じですが、各ラインの最後に垂直線"|"が加えられます。

10.3 チャートビューワの使用

チャートビューワを使用すると、マウスによって簡単にチャートの表示方法を操作できます。3D チャートの場合、チャートビューワの画面はドローイングパネルとナビゲーションパネルの 2 つのセクションで構成されます。2D チャートの場合はドローイングパネルのみが表示されます。チャートはドローイングパネル内の描画エリアに表示されます。



チャートビューワの表示例

以下にチャートビューワで実行できるチャートの操作をリストアップします。ただしこれは操作の順番や重要度を示すものではありません：

- ナビゲーションパネルで実行可能な操作：
  - 光源位置の変更
  - チャートの回転
  - チャートの移動
  - チャートのズームイン/ズームアウト
  - x、y、z 面でのチャートのスケーリング
  - バー/ライン/ポイント/パイの厚さ比の調整
  - チャートアニメーションの開始とスピードの操作
  - ワイヤーフレームモードとソリッドモードの切り換え
  - チャートのすべての縁に黒いアウトラインを描画
  - グローシェーディングの実行
- 左ボタンのドラッグにより、チャート上でメインタイトル、x/y/z レーベル、凡例を移動

- 右ボタンのドラッグにより、描画エリア上でチャートのサイズを変更
- Alt キーを押しながら右ボタンをドラッグすることにより、描画エリア上でキャンバスのサイズを変更
- ドローイングパネルで左ボタンをダブルクリックし、ナビゲーションパネルの表示/非表示を切り換え
- データポイントを個別にクエリ
- データポイントを左クリックし、そのデータポイントに関連付けられたデータを表示
- データポイントを右クリックし、そのデータポイントに関連付けられたハイパーリンクを表示
- 左ボタンをダブルクリックし、データポイントに関連付けられたハイパーリンクにジャンプ。（この操作を行うと新規のブラウザウィンドウが開きます。ブラウザの「戻る」ボタンで前のチャートに戻ることはできません）
- 右ボタンをダブルクリックし、一つ前のチャートに戻る
- Alt-Z によりズームインパラメータを指定
- Ctrl-R により手動でデータをリフレッシュ
- Ctrl キーを押しながら左クリックすることにより、ズームインの範囲を選択。Ctrl キーを押しながら右クリックすることにより、ズームアウトの範囲を選択。

10.4 軸目盛

オプションによりチャートビューワおよびチャート API で軸目盛を表示できます。目盛を表示するには、チャートビューワではアプレットタグでパラメータ{axisRuler}を設定します。チャート API で目盛を表示する方法については、API ドキュメントの quadbase.util.IAxisRuler を参照してください。軸目盛はスクロール時に参照ポイントとして役立つほか、チャートの移動によってチャート自体の軸が見えなくなった時に便利です。この機能は 2D チャートでのみ使用できます。

10.5 Parameter Server

Parameter Server は、Espress チャートビューワが TCP/IP を使用して指定されたリクエストを参照し、データの動的なアップデートを可能にします。シンタックスは以下の通りです。

```
<applet code = "quadbase.chartviewer.Viewer.class" width=640
height=480>
<PARAM name="ParameterServer" value="machine:portno">
</applet>
```

セキュリティ保護のため、machine は通常はウェブサーバと同じになります。そのため machine は未入力のままで構いません（例 value=":portno"）。Parameter Server の詳細については Appendix A を参照してください。

10.6 ポップアップメニュー

チャートの作成時や編集時にポップアップメニューのオプションを選択した場合、ポップアップメニューでチャートの修正が可能です。チャートを右クリックするとメニューが開きますので、左クリックでオプションを選択します。チャートの表示に使用可能なオプションはダイナミックデータドリルダウン、チャートタイプの変更、軸のズーミングおよび時間シリーズデータのズーミングです。

10.6.1 チャートの次元とデータの変更

ポップアップメニューから"2D/3D"オプションまたは"Type"オプションを使用することによって、チャートの次元およびチャートタイプを変更できます。

2D/3D オプションでは以下の選択が可能です：

- **2D**: 2D チャートに変更
- **3D**: 3D チャートに変更

Type オプションでは以下の選択が可能です：

- **Column**: カラムチャートに変更
- **Bar**: バーチャートに変更
- **Line**: ラインチャートに変更
- **Pie**: パイチャートに変更
- **Area**: エリアチャートに変更
- **Overlay**: オーバーレイチャートに変更
- **Box**: ボックスチャートに変更
- **Dial**: ダイアルチャートに変更

チャートが 3D の場合、オーバーレイ、ボックス、およびダイアルチャートのオプションは表示されません。

10.6.2 軸のズーミング

ビューポイントを超えるような範囲を持つ大きいチャートの場合（チャートが大きすぎてウィンドウ内で全体を表示することができない場合など）、オプションとして軸のズーミングが可能です。これによってすべての軸の移動やズームイン、ズームアウトが可能になります。

このオプションは 2D チャートでのみ使用できます。ただしバブル、ダイアル、パイ、レーダー、分散、スタックエリア、ポーラー、垂直ボックスの各チャートの場合、この機能は使用できません。

このオプションを使用するには、ポップアプメニューから"Enable Zoom"を選択します。x 軸のズームインと y 軸のズームインのどちらか、あるいは両方を選択できます。

ズームインするには、ズームしたいエリアを左クリックしてドラッグします。y 軸のズーミングのみが選択されている場合は、x 軸の長さや位置を考慮する必要はありません。

スクロールはズーミングされた軸にのみ行うことができます。軸のズーミングが有効な場合、その軸上にスクロールバーが表示されます。このスクロールバーを左クリックしてドラッグすることにより、アプレットのビューポイントを移動できます。

コントロールキーを押しながら左クリックすると、アプレットは前のズーム状態に戻ります。メモリに保存されるのは一段階のズーミングのみなので、戻ることができるのは一段階前のズーミングのみです。デフォルトの x および y スケールに戻るにはキーボードの"Home"キーを押します。

10.6.3 ズーミング

ズーミングが有効なチャートを表示している場合、ポップアップメニューでズーミングのオプションを使用できます。ズーミングオプションを使用すると、カテゴリエレメントをユーザ定義の間隔で分類して、各グループ内のポイントを集計することができます。日付/時間単位のズーミングの詳細については、セクション 7.8.2 を参照してください。ズーミングを有効にすることによって、下限や上限の設定（最初のデータポイントから最後のデータポイントに達する場合は無効化も可能）、時間スケールの設定、x 軸をリニアとするか否かの設定など、様々なズームオプションが使用できます。

10.6.4 ダイナミックデータドリルダウン

ポップアップメニューから Drill Down を選択することにより、次のドリルダウンを設定できます。このオプションはチャートのダイナミックデータドリルダウンが有効になっている場合のみ使用できます。ポップアップメニューから Drill Down オプションを選択すると、次のドリルダウンチャートの現在の設定を表示できます。Category が None である場合は設定がまだ行われていません。チャートの生成に使われているデータが未使用のカラムを含む場合、Category にカラムを選択できます。Category が選択されたら、次レベルのチャートの Type を設定します（Series および SumBy の設定も同時に行います）。設定が終わったら、データポイントを左クリックによって一段下のレベルへ、右クリックによって一段上のレベルへそれぞれ移動することができます。

10.6.5 Query Parameter

パラメータ化されたチャートを表示する場合、ポップアップメニューから Query Parameter オプションを選択することによってパラメータに異なるバリューを入力できます。このオプションを選択すると、選択されたパラメータに基づく新しいデータでチャートが再描画されます。

10.7Swing バージョン

EspressChart/lib ディレクトリで EspressViewer.jar の代わりに Swing EspressViewer.jar を参照することにより、チャートビューワの JFC/Swing バージョンの使用が可能になります。

11 EspressChart API の概要

11.1 イントロダクションと初期設定

EspressChart では、EspressChart デザイナ内でチャートテンプレートを設計および作成することに加えて、使いやすい API（アプリケーションプログラミングインターフェース）が提供されるため、ユーザは専用のアプリケーション（およびアプレット）内で 2D および 3D チャートを作成し、カスタマイズすることができます。API は 100% Pure Java で記述されているため、ほとんど、あるいはまったく変更することなく、どのプラットフォームでも実行できます。また、チャートは、API を使用して完全にカスタマイズできます。チャートにチャートを追加する場合も、わずか 4 行ほどのコードを使用するだけで済みます。

チャートの作成には、クラス QbChart を使用します。このコンポーネントにはさらに補助クラスのセットが組み合わされており、これらは quadbase.ChartAPI および quadbase.util の 2 つのパッケージに含まれています。以下の解説では、API の内容とそれらの使用方法について説明します。API ドキュメント全体は..../EspressChart/help/apidocs ディレクトリにあります。

API を使用するには、CLASSPATH に EspressAPI.jar と ExportLib.jar（EspressChart/lib ディレクトリ内にあります）を追加します。データの出力や読み込みに XML を使用している場合、同ディレクトリにある parser.jar と jaxp.jar を CLASSPATH に追加します。PNG にエクスポートする場合、同ディレクトリにある ExportLib.jar を CLASSPATH に追加します。チャートを SVG または Flash にエクスポートする場合、同ディレクトリにある SVGExport.jar または FlashExport.jar を CLASSPATH に追加します。また、CLASSPATH には qblicense.jar を追加する必要があります。アプリケーションを Windows または Solaris で使用している場合、ご使用のプラットフォームに合わせて以下の環境変数を追加してください。

```
((Windows 用)) set PATH=%PATH%;<path to EspressChart root
directory>\lib

(Solaris 用) export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:<path to
EspressChart root directory>/lib
```

11.2 チャート API の推奨される使い方

EspressChart は、チャートをプログラムで最初から作成するために使用できる API を提供します。ただし、通常、これは大量のコードで構成されます。したがって、デザイナを使用して、まずチャートテンプレート（.TPL ファイ

ル）を作成することをお奨めします。チャートテンプレートは、QbChart コンストラクタ内で受け渡すことができるため、このルック＆フィールは、新しく作成されるチャートオブジェクトにも適用できます。したがって、チャート作成時、大半のルック＆フィールはが設定されることになります。この後、プロパティを変更/追加/削除するコードを作成します。この方法を使用すると、コーディングの時間を短縮でき、パフォーマンスも向上します。

EspressChart のチャート API を使用する場合、EspressManager に接続するかどうかを選択できます。デザイナを使用する際には EspressManager に接続する必要がありますが、ExpressChart のチャート API を利用するアプリケーションを実行するときには、接続する必要はありません。通常、EspressManager に接続しないことをお奨めします。接続しないことによって、アーキテクチャ内の別の階層を回避できるからです。詳細については、次のセクションを参照してください。

本書に記載する例およびコードはすべて、上記の 2 つの勧告（すなわちテンプレートをベースにしたアプローチを使用することと EspressManager に接続しないこと）に従っています。特に明記しない限り、すべてのコード例は、テンプレート（<EspressChart をインストールしたディレクトリ>/help/manual/code/templates ディレクトリ内にあります）を使用し、EspressManager に接続しません。

さらに、EspressChart のチャート API を使用するアプレットがある場合、ブラウザには、少なくとも 1.4 JVM プラグインが必要であることにも注意してください。

11.3EspressManager とのインタラクション

EspressChart は一般的に EspressManager と組み合わせて使用されます。ファイルの読み書き、データベースへのアクセス、集計などの一部の高度な機能を実行するために必要なデータの事前処理などを行うには、チャートコンポーネントを EspressManager に接続します。ただし、チャートコンポーネントは単独モードで使用することもでき、その場合は、EspressManager を使用することなく直接ファイル I/O やデータベースアクセスを行います。

どちらの方法を使用してもそれぞれのメリットがあります。チャートがアプレット内に含まれている場合、セキュリティ上の制約によって直接ファイルの入出力を行うことはできません。クライアント上に JDBC ドライバがない場合はデータベースへのアクセスも困難になります。このような場合には、EspressManager によってこれらのサービスがチャートに供給されます。その一方で、チャートがアプリケーション内で使用されている場合はこのような制約が発生することはなく、直接アクセスが可能に

なることによってパフォーマンスが向上します。EspressManager が既知の場所で実行されている状態でなくともアプリケーションは実行することができます。

例えば Machine Client 上でアプレットまたはアプリケーションが実行されており、Machine DbMachine 上のデータベースからのデータを要求したとします。しかし Machine DbMachine がファイアーウォール内にあったり、セキュリティ上の制約によって Machine Client から Machine DbMachine への直接のアクセスが許可されない場合があります。EspressManager は Machine Server 上で作動し、アプレット/アプリケーションは Machine Server 上の EspressManager に接続することができます。このため、JDBC/ODBC を使用した Machine Server から Machine DbMachine への接続によってデータを得ることができるようになり、データが Machine Client に運ばれてチャートの生成が可能になります。こうした方法はデータのセキュリティを確保するためにクライアントマシンからの接続を不可能にし、すべての接続がサーバマシーン（EspressManager を実行しているマシーン）を通じて行われるようにしたい場合に便利です。また、この方法はすべてのクライアントが EspressChart を通じてデータにアクセスした際のログを保存するために利用することもできます（ログファイルは EspressManager を起動した時に作成されます。ログファイルは espressmanager.log という名前になります）。しかし、この方法ではコードが EspressManager を通じてデータへの接続を行う（レイヤーが追加される）ため、パフォーマンスが若干犠牲になるというデメリットもあります。

デフォルトではチャートコンポーネントは EspressManager があることを要求します。このモードを変更するには、アプレット/アプリケーションの開始時（QbChart オブジェクトが作成される前）に QbChart クラススタティックメソッドを使用します：QbChart objects are created):

```
static public void setEspressManagerUsed(boolean b)
```

アプリケーションおよびアプレットはともに、EspressManager にアクセスするかどうかにかに関わらず実行することができます。通信は、http プロトコルを使用して行われます。サーバの場所は、API コード内で受け渡される IP アドレスとポート番号によって決定されます。EspressManager の接続方法については、以下の手順を参照してください。

11.4 EspressManager への接続

11.4.1 EspressManager をアプリケーションとして実行する

EspressManager は基本的にはアプリケーションとして実行されます。アプリケーションとして実行されている EspressManager に接続するためにチャート API を使用する場合、API メソッドを使用して、EspressManager が置かれている IP アドレスまたはマシン名、および EspressManager がリスンするポート番号を指定します。
接続情報を設定するには、以下の 2 つの API メソッドを使用します。

```
static void setServerAddress(java.lang.String address);
static void setServerPortNumber(int port);
```

例えば以下のコードのラインを使用すると、someMachine 上で実行され somePortNumber を使用する EspressManager に接続します。

```
QbChart.setServerAddress("someMachine");
QbChart.setServerPortNumber(somePortNumber);
```

EspressManager への接続情報を指定しないと、コードは、ローカルマシン上で実行され、デフォルトのポート番号(22071)をリスンしている EspressManager に接続しようとすることに注意してください。

これらのメソッドは、QbChart および QbChartDesigner 内にあることに注意してください。

11.4.2 EspressManager をサーブレットとして実行する

EspressManager はサーブレットとして実行することも可能です。チャート API を使用してサーブレトとして実行されている EspressManager に接続するには、以下のメソッドを使用します。

```
public static void useServlet(boolean b);
public static void setServletRunner(String comm_url);
public static void setServletContext(String context);
```

例えば以下のコードのラインを使用すると、http://someMachine:somePortNumber/EspressChart/servlet で実行されている EspressManager に接続します。

```
QbChart.useServlet(true);
QbChart.setServletRunner("http://someMachine:somePortNumber");
QbChart.setServletContext("EspressChart/servlet");
```

これらのメソッドは、QbChart および QbChartDesigner 内にあることに注意してください。

11.5API の使用

以下のセクションでは、API の利用方法について説明します。以下の API の例およびコードも、上記の勧告（テンプレートをベースとしたアプローチを使用すること、および EspressManager に接続しないこと）に従って作成されています。

ほかに明記されていない限り、すべての例は、<EspressChart をインストールしたディレクトリ>/help/examples/DataSources/database ディレクトリ内にある Woodview HSQL データベースを使用します。例を実行するには、データベース HSQL JDBC ドライバ（hsqldb.jar）をクラスパスに追加する必要があります。ドライバは、<EspressChart をインストールしたディレクトリ>/help/examples/DataSources/database ディレクトリ内にあります。

さらに、本書では、すべての API 例はコアコードを示します。ただし、完全な例を入手して実行することもできます。どのコードスニペットには、完全な例を指示するリンクがあり、これらの例は、（CLASSPATH に EspressAPI.jar および qblicense.jar が含まれているとすれば）*help/manual/code/bin* ディレクトリから実行しなければなりません。

API メソッドの詳細については、API ドキュメンテーションを参照してください。

11.5.1 チャートのロード

チャートは、CHT または TPL フォーマット（ともにプロバイダ固有のフォーマット）を使用してファイルに保存することができます。TPL ファイルは、実データを除くすべてのチャート情報を保存するのに対して、CHT ファイルは、実データも保存します。これらのフォーマットは、チャートオブジェクトを再構築するために使用できます。データは、TPL ファイルを開くたびに自動的にオリジナルデータソースから再ロードされます。CHT ファイルを開くと、チャートを作成するために使用されたデータが表示されます（ただし、データソースからデータを取り出すためのオプションもあります）。

重要なのは、デフォルトでは、TPL ファイルにデータは含まれないという点です。これには、チャートテンプレート情報（すなわち、チャートのルック&フィール）とともに、指定されたデータソースが入ります。このため、TPL ファイルをロードすると、チャートのデータは、データソースに問い合わせることによって取得されます。一方、CHT ファイルは、開かれたときにデータソースに問い合わせません。これは、チャートを作成するために使用されたデータを表示して開始します。これらのフォーマットはともに、デザイナまたは QbChart クラス内で提供される export()メソッドを使用することによって取得できます。

以下の例は、アプレットまたはアプリケーションとして実行することができ、CHT ファイルを読み取って、チャートを再構築します。

```
Component doOpeningTemplate(Applet parent) {
      //       Do       not       use       EspressManager
      QbChart.setEspressManagerUsed(false);

      // Open the template
      QbChart chart = new QbChart (parent, // container
                  "../templates/API_10_5_1_OpeningChartTemplate.ch
      t"); // template

      // Show chart in Viewer


      return chart;
}
```

完全なソースコード
チャートテンプレート
エクスポートされた結果

この例のコンストラクタは、以下のとおりです。

```
public QbChart(Applet applet, String file);
```

ここで、applet はアプレットコンテナで、file は CHT/TPL ファイル名です。

注意: ファイル名は、URL 文字列として指定するか、または相対/絶対パスを使用して指定することができます。パス名は、EspressManager の現在のディレクトリに対して、または EspressManager を使用しない場合は現在のアプリケーションに対して相対的な名前とみなされます。

11.5.1.1 パラメータ化されたチャート

チャートには、何らかの形式でデータを制限するためのパラメータを含むことができます。一般に、データソースがデータを制限するには、クエリ（または IN）パラメータが使用されます。

クエリパラメータには、単一値および複数値パラメータ型のどちらでも使用することができます。

以下のコンストラクタを使用して、パラメータ化されたテンプレートを開くと、ダイアログボックスが表示され、パラメータの値を問い合わせてきます。このダイアログボックスは、テンプレートを開いたときにデザイナに表示されるダイアログボックスと同一です。

```
QbChart(Applet parent, String templateName);
```

11.5.1.1.1 オブジェクトアレイ

パラメータ化されたチャートは、値の入力を要求するダイアログボックスを表示しないで開くこともできます。アレイ内の各要素は、そのパラメータの値を表します。

アレイの順序は、パラメータがデザイナ内で作成された順序と一致しなければなりません。正しい結果を生成するには、値のデータ型もパラメータのデータ型と一致しなければなりません。

クエリパラメータには、複数値パラメータ型を使用することもできます。複数値パラメータの場合、ベクトルがオブジェクトアレイに渡されます。このベクトルには、複数値パラメータのすべての値が含まれます。

以下の例は、アプレットまたはアプリケーションとして実行することができ、チャートテンプレートを開いたときに、パラメータ値を渡します。

```
Component doObjectArray(Applet parent) {
      //       Do       not       use       EspressManager
      QbChart.setEspressManagerUsed(false);

      // Object array for Query Parameters
      Vector vec = new Vector();
      vec.addElement("CA");
      vec.addElement("NY");

      // Object array for Query Parameters
      Object queryParams[] = new Object[3];
      queryParams[0] = vec;
      queryParams[1] = new Date(101, 0, 4);
      queryParams[2] = new Date(103, 01, 12);

      // Open the template
      QbChart chart = new QbChart (parent, // container
                      "../templates/API_10_5_1_1_1_Object
      Array.cht", // template
                      queryParams); // Query Parameters

      // Show chart in Viewer
      return chart;

}
```

完全なソースコード
チャートテンプレート
エクスポートされた結果

11.5.2 チャートテンプレートの適用

QbChart オブジェクトを最初から作成する場合（付録 10.Bを参照）、チャートは、デフォルトの属性を使用して作成されます。ただし、チャートの構成中にチャートテンプレート（.tpl）を使用して、ユーザ定義の属性を指定することができます。ほとんどの属性（データソースおよびチャートタイプを除く）は、テンプレートから抽出されて、QbChart オブジェクトに適用されます。通常、テンプレート名は、QbChart コンストラクタ内で最後の引数として表示されます。

QbChart クラス内で applyTemplateFile（文字列 fileName）を使用してテンプレート名を指定することもできます。

以下の例は、アプレットまたはアプリケーションとして実行することができ、QbChart オブジェクトにテンプレートを適用します。

```
Component doApplyingTemplate(Applet parent) {
      // Do not use EspressManager
      QbChart.setEspressManagerUsed(false);

      String templateLocation = "..";

      // Apply the template
      QbChart chart = new QbChart (parent, // container
                      QbChart.VIEW2D, // chart dimension
                      QbChart.BAR, // chart type
                      data, // data
                      columnMapping, // column mapping
                      templateLocation); // template

      // Show chart in Viewer
      return chart;

}
```

完全なソースコード
チャートテンプレート
エクスポートされた結果

上記のコードは手引きとして掲載されているもので、完全ではないことに注意してください。ただし、リンク先には、実行可能な完全なアプリケーションコードが掲載されています。

さらに、テンプレートからカラムマッピングを行って、作成する QbChart オブジェクトにそれを適用させることもできます。このためには、CollInfo オブジェクトの代わりに”null”（二重引用符は不要）を渡します。

11.5.3 データソースの変更

デザイナ内でチャートテンプレートを作成し、API を使用して、それらのテンプレートを開くことができます。作成された QbChart オブジェクトは、テンプレートと同一のデータソースを使用し、データを取り出そうとします。ただし、テンプレートにすべてのルック＆フィールが入っている場合、デー

タソースは不正なものである可能性があります。以下では、チャート全体を作成し直すことなくデータソースを切り替える方法について説明します。

最善の結果を生成するには、カラム数と各カラムのデータ型が 2 つのデータソース（すなわち、デザイナ内でテンプレートを作成するために使用されたデータソースと新しいデータソース）間で一致しなければなりません。

データソースを切り替えると、QbChart オブジェクトが新しいデータを取り出すように強制しなければなりません。このためには、QbChart クラス内でリフレッシュメソッドを呼び出します。

11.5.3.1 データベースからのデータ

指示するデータソースをデータベースに切り替えるのは簡単です。必要なのは、データベース接続情報と使用されるクエリを指定して、それを QbChart オブジェクトに渡すことだけです。データベース接続情報（およびクエリ）は、DBInfo オブジェクト内で指定することができます（DBInfo オブジェクトの作成の詳細については、付録 10.A.1を参照してください）。

以下の例は、アプレットまたはアプリケーションとして実行することができ、QbChart オブジェクトのデータソースをデータベースに切り替えます。

```
Component doChartSwitchToDatabase(Applet parent) {
      // Do not use EspressManager
      QbChart.setEspressManagerUsed(false);

      // Open the template
      QbChart chart = new QbChart (parent, // container
                      "../templates/API_10_5_3_1_SwitchTo
      Database.cht"); // template

      // New database connection information
      DBInfo newDatabaseInfo = new DBInfo(.....);

      try {
            // Switch data source
            chart.gethInputData().setDatabaseInfo(newData
            baseInfo);

            // Refresh the chart
            chart.refresh();

      } catch (Exception ex)
      {
            ex.printStackTrace();
```

```
            }

            // Show chart in Viewer
            return chart;

      }
```

完全なソースコード
チャートテンプレート
エクスポートされた結果

上記のコードは手引きとして掲載されているもので、完全ではないことに注意してください。ただし、リンク先には、実行可能な完全なアプリケーションコードが掲載されています。

11.5.3.1.1 JNDI

データソースを JNDI データソースに変更することもできます。このためには、DBInfo オブジェクトで JNDI 接続情報を指定した後、それを QbChart オブジェクトに渡します。

以下の例は、アプレットまたはアプリケーションとして実行することができ、QbChart オブジェクトのデータソースを JNDI データベースに切り替えます。

```
      Component doChartSwitchToDatabaseJNDI(Applet parent) {
            // Do not use EspressManager
            QbChart.setEspressManagerUsed(false);

            // Open the template
            QbChart chart = new QbChart (parent, // container
                        "../templates/API_10_5_3_1_1_ChartS
            witchToDatabaseJNDI.cht"); // template

            // New database connection information
            DBInfo newDatabaseInfo = new DBInfo(.....);

            try {
                  // Switch data source
                  chart.gethInputData().setDatabaseInfo(newData
                  baseInfo);

                  // Refresh the chart
                  chart.refresh();

            } catch (Exception ex)
            {
```

```
            ex.printStackTrace();
        }

        // Show chart in Viewer
        return chart;

    }
```

完全なソースコード
チャートテンプレート
エクスポートされた結果

上記のコードは手引きとして掲載されているもので、完全ではないことに注意してください。ただし、リンク先には、実行可能な完全なアプリケーションコードが掲載されています。アプリケーションコードを実行する場合、Tomcat 環境内で Woodview データベースを JNDI データソースとして設定し、接続情報が一致するようにアプリケーションコードを変更する必要があることに注意してください。

11.5.3.2 データファイル（txt/dat/xml）からのデータ

テキストファイルが Quadbase ガイドラインに従っていれば、データソースをテキストファイルに切り替えることができます（詳細については、付録 10.A.2を参照してください）。

以下の例は、アプレットまたはアプリケーションとして実行することができ、QbChart オブジェクトのデータソースをテキストファイルに切り替えます。

```
Component doChartSwitchToDataFile(Applet parent) {
        // Do not use EspressManager
        QbChart.setEspressManagerUsed(false);

        // Open the template
        QbChart chart = new QbChart (parent, // container
                        "../templates/API_10_5_3_2_ChartSwi
        tchToDataFile.cht"); // template

        try {

              // Switch data source
              chart.gethInputData().setDataFile(....);
```

```
                // Refresh the chart
                chart.refresh();

        } catch (Exception ex)
        {
                ex.printStackTrace();
        }

        // Show chart in Viewer
        return chart;

}
```

完全なソースコード
チャートテンプレート
エクスポートされた結果

上記のコードは手引きとして掲載されているもので、完全ではないことに注意してください。ただし、リンク先には、実行可能な完全なアプリケーションコードが掲載されています。

11.5.3.3 XML データソースからのデータ

データを伴う.dtd または.xml スキーマがある場合、データソースをカスタム XML データに切り替えることができます。XML データ情報を指定して（詳細については、付録 10.A.3を参照してください）、QbChart オブジェクトに渡します。

以下の例は、アプレットまたはアプリケーションとして実行することができ、QbChart オブジェクトのデータソースを XML データに切り替えます。

```
Component doSwitchToXMLData(Applet parent) {
        // Do not use EspressManager
        QbChart.setEspressManagerUsed(false);
        // Open the template

        QbChart chart = new QbChart (parent, // container
                        "../templates/API_10_5_3_3_ChartSwi
        tchToXMLData.cht"); // template
```

```
        // XML data source information
        XMLFileQueryInfo newData = new
        XMLFileQueryInfo(...);

        try {
              // Switch data source
              chart.gethInputData().setXMLFileQueryInfo(new
              Data);
              // Refresh the chart
              chart.refresh();

        } catch (Exception ex)
        {
              ex.printStackTrace();
        }

        // Show Chart in Viewer
        return chart;

}
```

完全なソースコード
チャートテンプレート
エクスポートされた結果

上記のコードは手引きとして掲載されているもので、完全ではないことに注意してください。ただし、リンク先には、実行可能な完全なアプリケーションコードが掲載されています。

11.5.3.4 カスタムインプリメンテーションからのデータ

正規のデータソースに加えて、専用のカスタムデータを渡すこともできます。カスタムデータを QbChart オブジェクトに渡すには、IDataSource インタフェースを使用します（詳細については、付録 10.A.5を参照してください）。

以下の例は、アプレットまたはアプリケーションとして実行することができ、QbChart オブジェクトのデータソースをカスタムインプリメンテーションに切り替えます。

```
Component doSwitchToCustomData(Applet parent) {
```

```
        // Do not use EspressManager
        QbChart.setEspressManagerUsed(false);

        // Open the template
        QbChart chart = new QbChart (parent, // container
                        "../templates/API_10_5_3_4_ChartSwi
        tchToCustomData.cht"); // template

        try {

              // Switch data source
              chart.gethInputData().setClassFile(..);

              // Refresh the chart
              chart.refresh();

        } catch (Exception ex)
        {

              ex.printStackTrace();

        }

        // Show chart in Viewer
        return chart;

}
```

カスタムインプリメンテーションソースコード

完全なソースコード
チャートテンプレート
エクスポートされた結果

上記のコードは手引きとして掲載されているもので、完全ではないことに注意してください。ただし、リンク先には、実行可能な完全なアプリケーションコードが掲載されています。

11.5.3.5 メモリ内でのアレイとしてのデータ引き渡し

アレイを使用してデータを渡すこともできます。通常、アレイデータは、メモリ内に格納され、QbChart オブジェクトに渡されます（詳細については、付録 10.A.4を参照してください）。

以下の例は、アプレットまたはアプリケーションとして実行することができ、QbChart オブジェクトのデータソースをメモリ内のアレイに切り替えます。

```
Component doSwitchToArrayData(Applet parent) {
      // Do not use EspressManager
      QbChart.setEspressManagerUsed(false);

      // Open the template
      QbChart chart = new QbChart (parent, // container
                        "../templates/API_10_5_3_5_ChartSwi
      tchToArrayData.cht"); // template

      // Create array data
      DbData newData = new DbData(...);

      try {
            // Switch data source
            chart.gethInputData().setData(newData);

            // Refresh the chart
            chart.refresh();

      } catch (Exception ex)
      {
            ex.printStackTrace();
      }

      // Show chart in Viewer
      return chart;

}
```

完全なソースコード
チャートテンプレート
エクスポートされた結果

上記のコードは手引きとして掲載されているもので、完全ではないことに注意してください。ただし、リンク先には、実行可能な完全なアプリケーションコードが掲載されています。

11.5.4 チャート属性の修正

EspressChart には数百に及ぶチャートプロパティがあり、これによってチャートの様々な要素の微調整が可能です。プロパティはいくつかのグループにカテゴリ分類さ

れ、インターフェースとして提供されますので、これらの調整は簡単に行えます。まずアプリケーションが gethXXX メソッドを使用してインターフェースのグループに対するハンドルを取得し、さらにそのインターフェース上のメソッドを呼び出すことによってチャートのプロパティを直接操作します。ほとんどのインターフェースは quadbase.util および quadbase.ChartAPI パッケージに含まれています。

11.5.4.1 カラー、フォントなどの調整

チャートはインスタンス、キャンバス、軸、凡例、チャートデータなど、様々なエレメントによって構成されています。これら各エレメントは、それぞれに対応するインターフェースを取得し、メソッドを呼び出してプロパティを変更することによって修正できます。

例えば以下のコードは、チャートの背景色を白に設定します。

```
chart.gethCanvas().setBackgroundColor(Color.white);    //
chart being an object of type QbChart
```

また以下のコードは、X 軸の色を黒に変更します。

```
chart.gethXAxis().setColor(Color.black);               //
chart being an object of type QbChart
```

チャートのデータエレメントのカラーは、シリーズまたはカテゴリ（シリーズが定義されていない場合）に対して設定することもできます。例えばシリーズ内に 5 つのエレメントがあるカラムチャートの場合、以下のコードによってシリーズ内のカラムのカラーをイエロー、オレンジ、レッド、グリーン、ブルーに設定できます。

```
Color dataColors[] = {Color.yellow, Color.orange, Color.red,
Color.green, Color.blue};
chart.gethDataPoints().setColors(dataColors);        //
chart being an object of type QbChart
```

定義されるカラーの数は、シリーズ（またはシリーズが定義されない場合、カテゴリ）内の独立したエレメントの数と同じである必要があります。

同様にフォントのスタイルも、適切なインタフェースを呼び出してメソッドを使用することで設定できます。以下のコードは凡例に使用されるテキストのフォントを Arial ボールド、サイズ 15 に設定します。

```
Font legendFont = new Font("Arial", Font.BOLD, 15);
chart.gethLegend().setFont(legendFont);             // chart
being an object of type QbChart
```

以下のコードは、Y 軸に使用されるラベルのフォントと色を設定します。

```
Font YAxisFont = new Font("Helvetica", Font.ITALIC, 15);
chart.gethYAxis().gethLabel().setFont(YAxisFont);    //
```

```
chart being an object of type QbChart
chart.gethYAxis().gethLabel().setColor(Color.blue);  //
chart being an object of type QbCha
```

他のプロパティも同様に、インターフェース内のメソッドを使用することによって設定を行います。

11.5.4.2 事前定義パターンの設定

クラス QbPattern は java.awt.Color のサブクラスであるため、パターンを明示的に設定するために新しく紹介するメソッドはありません。基本的に、まず QbPattern オブジェクトを作成し、それを Color オブジェクトと同様に使用します。ここからは、EspressChart がそのオブジェクトを内部で処理します。軸、キャンバスなどのほかのコンポーネントの場合、パターン機能はデータポイントにのみ適用されます。EspressChart は QbPattern プロパティを無視し、Color とまったく同様に使用します。

以下の例は、アプレットまたはアプリケーションとして実行することができ、データポイントのパターンの設定方法を示します。

```
Color[] dataColors = chart.gethDataPoints().getColors();
QbPattern dataPattern = new QbPattern(colors[2],
QbChart.PATTERN_THICK_FORWARD_DIAGONAL);
dataColors[2] = dataPattern;
chart.gethDataPoints().setColors(dataColors);
```

完全なソースコード
チャートテンプレート
エクスポートされた結果

全部で 30 種類の事前定義パターンを利用することができます。パターン ID は 0 から 29 までです。ユーザがこの範囲を超える整数を渡すと、ID=0 とみなされます。これは、真っ黒です。パターン ID は、qadbase.ChartAPI. IMiscConstants クラスと quadbase.common.swing.color.PattenImage クラス内で定義されます。QbChart では IMiscConstants インターフェースがインプリメントされているため、ユーザはこれらの IDQbChart から直接取得することができます。チャートデザイナーのパターンパレットで各パターンがどのように表示されるかを確認することができます。

*

11.5.4.3 カスタマイズ化パターンの設定

さらに、独自のパターン（テキスチャ）イメージを設定することもできます。この機能は API を経由してのみ利用することができ、変更は.cht または tpl ファイルに保存されません。この機能は主として、実行時にチャートを変更する必要があるユーザ向けです。

以下の例は、アプレットまたはアプリケーションとして実行することができ、データポイントのカスタムパターンの設定方法を示します。

```
Color[]    dataColors    =    chart.gethDataPoints().getColors();
QbPattern    dataPattern    =    new    QbPattern(dataColors[2]);
File  customImageFile  =  new  File("../data/Quadbase_Logo.png");
BufferedImage    customImage    =    ImageIO.read(customImageFile);
dataPattern.setPatternImage(customImage);
dataColors[2]                    =                    dataPattern;
chart.gethDataPoints().setColors(dataColors);
```

完全なソースコード
チャートテンプレート
エクスポートされた結果

上記のコードでは、パターン ID を指定しないで QbPattern オブジェクトを作成します。新しく作成されるオブジェクトは、java.awt.Color オブジェクトと同等です。開発者は、手続き内で"NullPointerException"プロンプトを回避するためにイメージを指定する責任があります。次に、ファイルから既存のイメージをデータポイント 2 のパターンとして渡します。

11.5.4.4 サイズの調整

チャートの作成にあたっては、以下の 2 つのサイズ設定が使用できます。

- **相対サイズ:** サイズをチャートキャンバスに対する割合で設定します。例えばチャート描画の高さを.7、幅を.8 に設定したとします。キャンバスが 300 x 300 ピクセルであった場合、描画されるチャートの高さは 210 ピクセル (.7 x 300) 、幅は 240 ピクセル (.8 x 300)になります。キャンバスサイズが 500 x 600 ピクセルに広げられた場合、描画されるチャートの高さは 420 ピクセル (.7 x 600)、幅は 400 ピクセル(.8 x 500)になります。相対サイズではキャンバスの高さと幅によってチャートのサイズが決定されます。
- **絶対サイズ:** キャンバスのサイズがピクセルで指定されます。

サイズパラメータが使用されているすべてのチャートエレメントは、キャンバスに対する相対サイズで定義されます。そのため、チャートエレメントの相対サイズは浮動小数点を使用して表されます。例えば以下のコードでは、チャートの高さは.85、幅は.65 に設定されます。

```
// chart being an object of type QbChart
chart.gethChartPlot().setRelativeHeight(.85f);
chart.gethChartPlot().setRelativeWidth(.65f);
```

以下のコードはチャートのキャンバスサイズを設定します。

```
// chart being an object of type QbChart
Dimension canvasSize = new Dimension(500, 450);
chart.gethCanvas().setSize(canvasSize);
```

他のエレメントのサイズも、同様にインターフェース内のメソッドの使用によって設定できます。

11.5.4.5 日付/時刻ズームチャートの変更

デザイナ内で作成された日付/時刻ズームテンプレートのプロパティ（目盛、開始または終了時間など）を変更することができます。このためには、（IZoomInfo インタフェースを使用して）ズームプロパティのハンドルを取得した後、そこで各種のメソッドを使用してプレゼンテーションを変更します。

以下の例は、アプレットまたはアプリケーションとして実行することができ、日付/時刻ズームチャートの変更方法を示します。

```
// Modify starting and ending period and scale
IZoomInfo zoomInfo = chart.gethZoomInfo();

// Begin Date
Calendar beginDate = new GregorianCalendar(2001, 0, 1);

// End Date
Calendar endDate = new GregorianCalendar(2003, 11, 31);

zoomInfo.setLowerBound(beginDate.getTime());
zoomInfo.setUpperBound(endDate.getTime());

zoomInfo.setScale(6, IZoomInfo.MONTH);

chart.refresh();
```

完全なソースコード
チャートテンプレート
エクスポートされた結果

上記のコードでは、チャートは、QbChart 型オブジェクトです。チャートのズームプロパティを変更した後、変更したプロパティを有効にするには、チャートをリフレッシュする必要があることに注意してください。

11.5.5 チャートのエクスポート

デフォルトでは、チャートに含まれるチャートは、JPEG としてエクスポートされますが、JPEG、GIF および PNG フォーマットとしてもエクスポートすることができます。チャートのフォーマットは変更することができます。ChartChartObject オブジェクトを作成する際に、イメージタイプがメソッド setImageType(int)を使用することを指定します。

EspressChart API は GIF、JPEG、PNG、BMP、PDF 等の様々なフォーマットでチャートをエクスポートできます。さらに、チャートを独自形式である.cht または.tpl フォーマットでエクスポートすることもできます。.cht はすべてのチャート情報を保存するフォーマットで、EspressChart では静止画像ファイルとして扱われます。ただし.cht ファイルは通常の静止画像ファイルとは異なり、データソースからファイル内のデータをリフレッシュしたり、ズームやドリルダウンなどの追加機能を持つことができます。.tpl ファイルは.cht ファイルと同じですが、実際のデータは保存しません。.tpl ファイルでは、チャートが読み込まれる時にオリジナルのデータソースから自動的にデータがリロードされます。.cht ファイルと.tpl ファイルは、Espress チャートビューワまたは EspressChart API で表示できます。スタンドアロンチャートを使用する場合、QbChart オブジェクトを作成して、そのクラス内で各種のエクスポートメソッドを使用することによって、フォーマットを指定することができます。

異なるオプションを使用して、スタンドアロンチャートを各種のフォーマットでエクスポートするために使用できる様々なメソッドがあります。最も単純なのは以下のメソッドです。

```
public export(int format, String filename)
```

上記のメソッドで、format は、いずれかのフォーマットコンスタントを表し、filename は出力するファイル名（拡張子付き、またはなし）を表します。

以下にチャートコンポーネントのエクスポートに使用するコンスタントのリストを示します。

- Bitmap: QbChart.BMP
- GIF: QbChart.GIF
- JPEG: QbChart.JPEG
- PNG: QbChart.PNG
- PDF: QbChart.PDF
- Flash: QbChart.FLASH
- Scalable Vector Graphics : QbChart.SVG

- Text データ: QbChart.TXTFORMAT
- XML データ: QbChart.XMLFORMAT
- Windows Meta File : QbChart.WMF

選択したフォーマットによっては、追加オプションが使用できます。例えばチャートオブジェクトを jpeg でエクスポートする場合、画質の選択（0～99 の範囲）を行えます。また PNG でエクスポートする場合、使用する圧縮の種類を選択できます。オプションは以下のメソッドで指定します。

```
public export(int format, String filename, int option)
```

以下のコードは、アプレットまたはアプリケーションとして実行することができ、チャートの構築とエクスポートの方法を示します。

```
// Open the template
QbChart chart = new QbChart(parent,
"../templates/API_10_5_5_ExportChart.cht");

try {
      chart.export(QbChart.PNG,
      "../export/API_10_5_5_ExportChart");
} catch (Exception ex) {
      ex.printStackTrace();
}
```

完全なソースコード
チャートテンプレート
エクスポートされた結果

チャートをテキストファイルとしてエクスポートする場合、デフォルトではタブ区切りが行われます。しかし以下のメソッドにより、他の区切り文字（","、";" および " "）の設定も可能です。

```
// where chart is an Object of type QbChart
chart.export(QbChart.TXTFORMAT, "TextData", QbChart.COMMA);
```

チャートオブジェクトを表示せずにエクスポートする場合、以下の静的メソッドを true に設定するとパフォーマンスがわずかに向上します。

```
QbChart.setForExportOnly(boolean);
```

このメソッドを QbChart オブジェクトを構築する前に呼び出すと、パフォーマンスが向上します。

以下のメソッドにより、QbChart オブジェクトをバイトアレイにエクスポートできます。

```
public byte[] exportChartToByteArray();
```

これによって、.cht と同様のファイルをバイトアレイの形式でエクスポートできます。これは QbChart オブジェクトをシリアル化する場合に便利です。

11.5.5.1 レコードファイルのエクスポート

レコードファイルのエクスポートを使用すると、大量のデータをより効率的に扱えます。レコードファイルをエクスポートするには、メモリに記憶するレコードの数と temp ディレクトリを指定します。データ内のレコードの数が指定した数を超えると、データはディスク上で指定された temp ディレクトリに保存されます。

この機能を使用するには、以下の条件が必要です。

- EspressManager が使用されていない
- XML ソースからのデータを使用していない
- データが保存されている

レコードファイルのエクスポートを行うには、QbChart コンストラクタを呼び出す前に以下のコードを追加する必要があります。

```
QbChart.setEspressManagerUsed(false);
QbChart.setTempDirectory(<specify the temp directory to store data.
Default is ./temp>);
QbChart.setMaxCharForRecordFile(<specify the maximum character
length. Default is 40>);
QbChart.setMaxRecordInMemory(<specify the maximum number of rows to
be kept in memory. Default is -1 i.e., store all records in memory>);
QbChart.setFileRecordBufferSize(<specify the number of rows to be
retrieved at one time from disk. Default is 10,000>);
```

11.5.5.2 チャートのストリーム

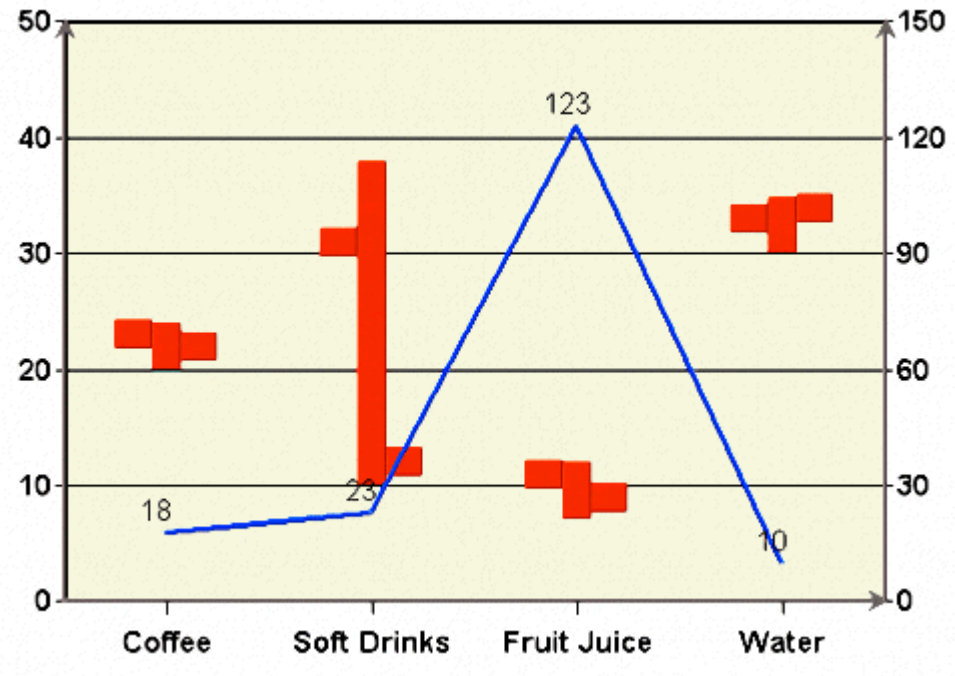
チャートは、ローカルドライバにエクスポートするだけでなく、バイトストリームとしてエクスポートして、Web ブラウザに直接ストリームすることができます。通常、サーバ側のコードは、イメージをバイナリフォーマットでのみエクスポートすることに注意してください。エクスポートされたイメージのラッパー（すなわち、DHTML/HTML コンテンツ）を作成するのはユーザの責任です。

以下の例は、チャートを PNG にエクスポートして、それをブラウザにストリームする例です。この例を実行するには、ソースコードを作成してコンパイルした後、サーブレットランナーのサーブレットディレクトリ内にクラスファイルをデプロイメントする必要があります。chartTemplate 変数を絶対パス、またはご使用のアプリケーションサーバのワーキングディレクトリに対する相対パスに変更してください。

```
public void doGet(HttpServletRequest req,
HttpServletResponse resp) throws ServletException,
IOException {
      // Do not use EspressManager
      QbChart.setEspressManagerUsed(false);

      String chartTemplate =
      "../templates/API_10_5_5_2_StreamingChart.cht";

      // Open template
      QbChart chart = new QbChart((Applet)null,
      chartTemplate);

      // Export the chart to PNG and stream the bytes
      ByteArrayOutputStream chartBytes = new
      ByteArrayOutputStream();

      try {

            chart.export(QbChart.PNG, chartBytes);

      } catch (Exception ex)
      {

            ex.printStackTrace();

      }

      resp.setContentType("image/x-png");
      resp.setContentLength(chartBytes.size());

      OutputStream toClient = resp.getOutputStream();
      chartBytes.writeTo(toClient);
      toClient.flush();
      toClient.close();

}
```

完全なソースコード
チャートテンプレート

この例を実行するには、必ず、コードがアクセスする位置にチャートテンプレートをコピーしてください。（例で示すように）相対パスを使用する場合、現在のディレクトリは、ご使用のアプリケーションサーバのワーキングディレクトリであることに注意してください。たとえば、Tomcat のワーキングディレクトリは<Tomcat>/bin あるため、コードを実行するには、ディレクトリ<Tomcat>/templates/を作成して、そこにチャートテンプレートをコピーする必要があります。生成されるチャートを以下に示します。



チャートのストリーム

11.5.6 チャート API からのチャートデザイナーの呼び出し

チャートデザイナーをアプリケーションまたはアプレット内のチャート API から呼び出すこともできます。コードによって、パラメータを渡して指定したチャートファイル、指定したデータソース、または他のパラメータでチャートデザイナーを開くことができます。

チャート API からチャートデザイナーを呼び出すには以下の点を確認してください。

- EspressManager が実行されていること。
- EspressAPI.jar および qblicense.jar が CLASSPATH に追加されていること。
- 関連の API 呼び出しを使用して、EspressManager に対する接続情報が指定されていること（これは、EspressManager が別のマシン上にある場合、または EspressManager が 22071 以外のポート番号で起動されている場合、あるいはその両方の場合に特に重要です）。
- イメージと backgroundimages ディレクトリがワーキングディレクトリにコピーされているか、またはコード内の適切なコンストラクタで（これらのディレクトリの）位置が渡されていること。

EspressManager の-RequireLogin および-QbDesignerPassword フラグによっては、EspressManager に接続するために userid および password を引き渡す必要があります。EspressManager に-RequireLogin フラグが設定されている場合、setVisible()またはいずれかの getDesigner メソッドを呼び出す前に以下のコードを追加する必要があります。

```
public login(String userName, String password);
// Method found in QbChartDesigner class
```

EspressManager に-QbDesignerPassword が設定されている場合、setVisible()またはいずれかの getDesigner メソッドを呼び出す前に以下のコードを追加する必要があります。

```
public login(String password);
// Method found in QbChartDesigner class
```

以下にチャート API からチャートデザイナーを呼び出すための各方法を解説します。

11.5.6.1 チャートテンプレートを指定する

チャートテンプレートファイルを指定してチャートデザイナーを開くことができます。こうすることによって、エンドユーザはチャートデザイナー GUI 内で専用のカスタムチャートを作成し、完成したチャートを確認できます。

以下のコンストラクタを使用します :

```
QbChartDesigner(Object parent, String chtFile, String[]
imagesPath);
```

以下は、.cht ファイルを指定してチャートデザイナー を呼び出す例です :

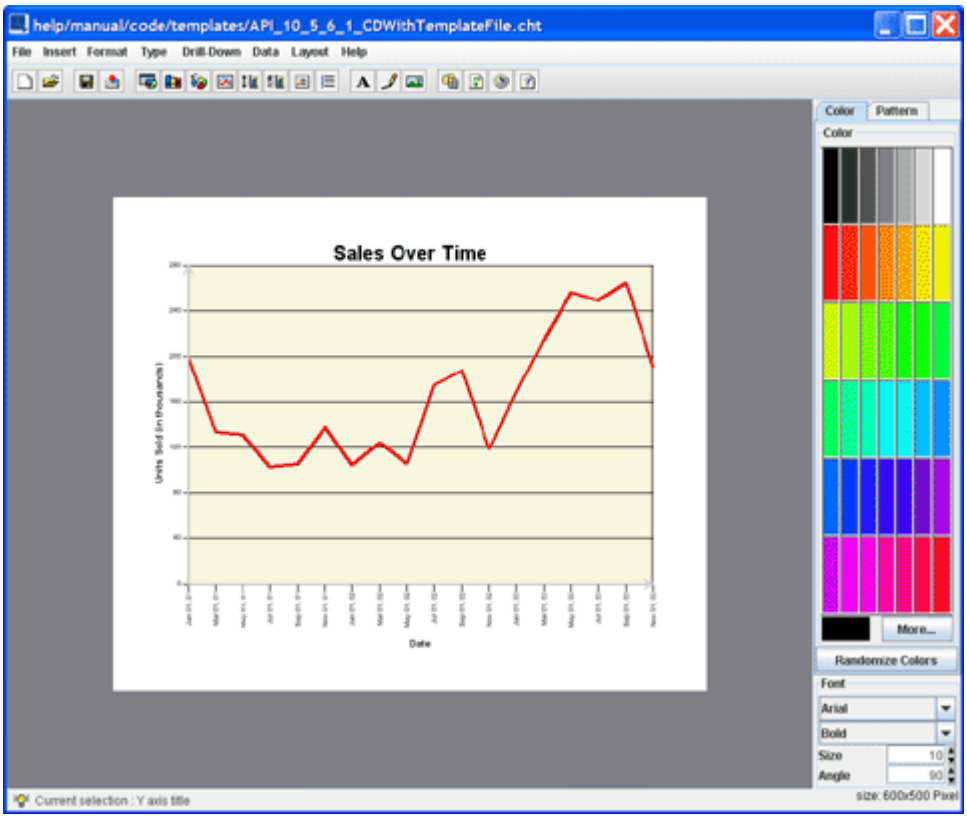
```
public void doChartDesignerApplet(Object parent) throws
Exception {
      // Connect to EspressManager
      QbChartDesigner.setServerAddress("127.0.0.1");
      QbChartDesigner.setServerPortNumber(22071);

      String[] imagesPath = {"../../../images",
      "../../../backgroundImages"};

      // Create a new QbChartDesigner instance
      designer = new QbReportDesigner(parent,
            "help/manual/code/templates/API_10_5_6_1_CDW
      ithTemplateFile.cht", imagesPath);
```

```
            // Start Designer
            designer.setVisible(true);

        }
```

[完全なソースコード]
[テンプレートファイル]
[アプレットとしてシステム実行する（HTTP プロトコルを使用しなければならない）]

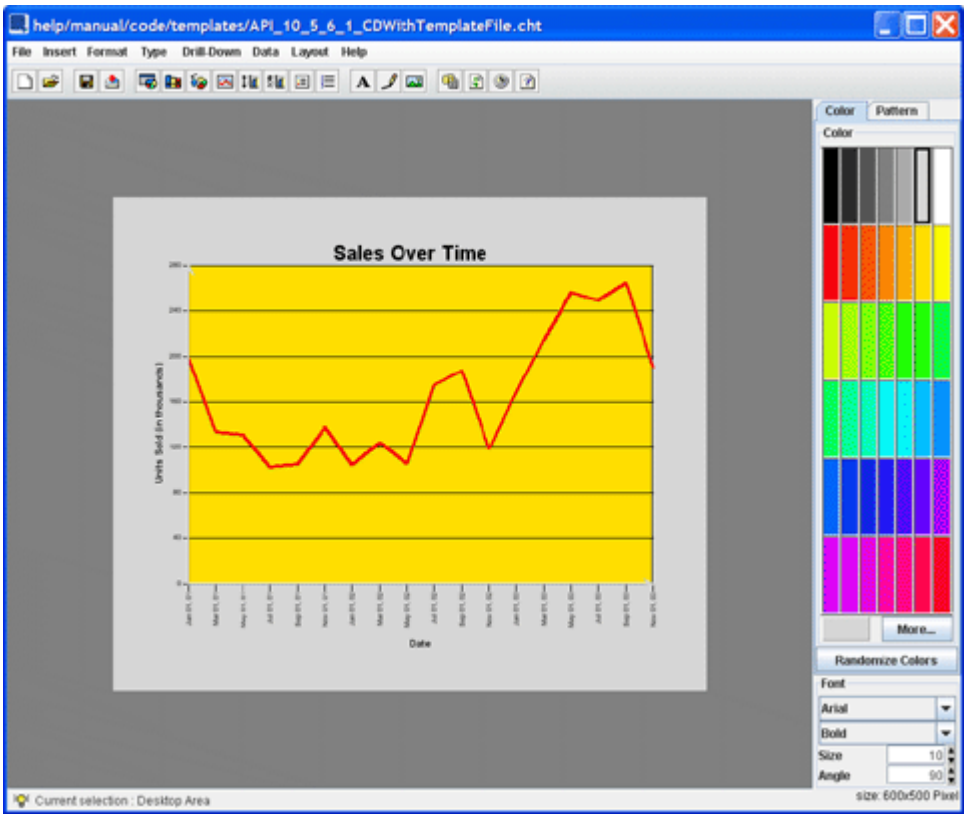
上記のコードはアプリケーションとアプレットのどちらとしても使用することができます。

ユーザがこのコードを実行すると、チャートデザイナーは、指定されたチャートテンプレートで起動されます。



テンプレートが指定されたチャートデザイナー

この後、ユーザは、チャートをカスタマイズして、結果を保存することができます。



エンドユーザによるカスタマイズ

11.5.6.2 データレジストリを指定する

チャートデザイナーを開いて、（データレジストリ用の.xml ファイルを指定した）データソースマネージャから開始させることができます。こうすることによって、エンドユーザはチャートデザイナー GUI 内で専用のカスタムチャートを作成し、完成したチャートを確認することができます。

以下のコンストラクタを使用します：

```
QbChartDesigner(Object parent, String nameOfXMLFile, boolean
newChart, String[] imagesPath);
```

以下は、.xml ファイルを指定して チャートデザイナー を呼び出す例です：

```
public void doChartDesignerWithDataRegApplet(Object
parent) throws Exception {
```

```
            // Connect to EspressManager
            QbChartDesigner.setServerAddress("127.0.0.1");
            QbChartDesigner.setServerPortNumber(22071);

            String[] imagesPath = {"../../../images",
            "../../../backgroundImages"};

            // QbChartDesigner (Object, name of XML file, start
            from Data Registry?, images directory)
            designer = new QbChartDesigner(parent,
            "DataRegistry/sample.xml", true, imagesPath);

            designer.setVisible(true);
      }
```

完全なソースコード
アプレットとしてシステム実行する（HTTP プロトコルを使用しなければならない）

上記のコードはアプリケーションとアプレットのどちらとしても使用することができます。

11.5.6.3 DBInfo オブジェクト（データベース情報）を指定する

チャートデザイナーを開いて、（DBInfo オブジェクトを指定した）"Select Chart Type"ウィザードウィンドウから開始することができます。こうすることによって、エンドユーザはチャートデザイナー GUI 内で専用のカスタムチャートを作成し、完成したチャートを確認することができます。

以下のコンストラクタを使用します：

```
QbChartDesigner(Object parent, DBInfo databaseInformation,
QueryInParamSet inset boolean newChart, String[]
imagesPath);
```

以下は、DBInfo オブジェクトを指定して レポートデザイナー を呼び出す例です：

```
public QbChartDesigner designer;
String url =
"jdbc:hsqldb:help/examples/DataSources/database/woodview"
;
String driver = "org.hsqldb.jdbcDriver";
String username = "sa";
```

```
String password = "";
String query = "SELECT * FROM Order_Details";

public void doChartDesignerWithDBInfoApplet(Object
parent) throws Exception {
      // Connect to EspressManager
      QbChartDesigner.setServerAddress("127.0.0.1");
      QbChartDesigner.setServerPortNumber(22071);

      String[] imagesPath = {"../../../images",
      "../../../backgroundImages"};

      DBInfo dbInfo = new DBInfo(url, driver, username,
      password, query);

      // QbChartDesigner (Object, Database Information,
      Parameter information, start from Select Chart
      wizard?, Images paths)
      designer = new QbChartDesigner(parent, dbInfo, null,
      true, imagesPath);

      designer.setVisible(true);
}
```

完全なソースコード
アプレットとしてシステム実行する（HTTP プロトコルを使用しなければならない）

上記のコードはアプリケーションとアプレットのどちらとしても使用することができます。

11.5.6.4 DBInfo オブジェクトを指定してクエリビルダを開く

DBInfo オブジェクトを指定してクエリビルダを開くことができます。こうすることによって、エンドユーザはクエリビルダ GUI 内で専用のカスタム SQL クエリを作成し、保存することができます。

以下に、DBInfo オブジェクトを指定してクエリビルダを呼び出す例を示します：

```
public void doQueryBuilderApplet(Object parent, String
sqlName) {
      // Connect to EspressManager
      QbChartDesigner.setServerAddress("127.0.0.1");
      QbChartDesigner.setServerPortNumber(22071);
```

```
            queryMain = new QbQueryBuilder(parent, this);
            if (sqlName != null) {
                  // sqlName .qry file name (without extension)
                  // System.out.println("OPEN QUERY - " +
                  sqlName);
                  queryMain.openQuery(sqlName, false, null,
                  null, driver, url, username, password);
            } else {
                  // System.out.println("NEW QUERY ");
                  queryMain.newQuery("Setup Query", false, null,
                  null, driver, url, username, password);
            }
            frame = queryMain.getBuilder();
            frame.setVisible(true);
            try { queryMain.showTablesWindow();
            } catch (Exception ex) {};

      }

      public void back() {};

      public void cancel() {};

      public void next() {
            queryMain.getBuilder().setVisible(false);
            DBInfo dbInfo = new DBInfo(url, driver, username,
            password, queryMain.getSQL());
            designer = new QbChartDesigner(applet, dbInfo,
            queryMain.getInSet(), true, imagesPath);
            designer.setVisible(true);
      }
```

完全なソースコード
アプレットとしてシステム実行する（HTTP プロトコルを使用しなければならない）

上記のコードはアプリケーションとアプレットのどちらとしても使用することができます。

クエリビルダを、データベース接続（スキーマ）内で DBInfo オブジェクトではなく Java オブジェクトとして引き渡すことによって開くこともできます。以下に、スキーマ情報を引き渡すことによってクエリビルダを呼び出す例を示します：

```
      public API_3_5_8_4_QBWithDBInfo2() {
```

```
        // Connect to EspressManager
        QbChartDesigner.setServerAddress("127.0.0.1");
        QbChartDesigner.setServerPortNumber(22071);

        try {
              Class.forName(driver);
              conn = DriverManager.getConnection(url,
              username, password);
              metaData = conn.getMetaData();
        } catch (Exception ex) { ex.printStackTrace(); }

        queryMain = new QbQueryBuilder(null, this);
        queryMain.newQuery("Setup Query", this);
        queryMain.setVisible(true);

        try { queryMain.showTablesWindow();
        } catch (Exception ex) {};

  }

  public void back() {};
  public void cancel() {};

  public void next() {
        queryMain.getBuilder().setVisible(false);
        DBInfo dbInfo = new DBInfo(url, driver, username,
        password, queryMain.getSQL());
        designer = new QbChartDesigner(applet, dbInfo,
        queryMain.getInSet(), true, imagesPath);
        designer.setVisible(true);
  }

  public String getDatabaseProductName() throws Exception {
        return metaData.getDatabaseProductName();
  }

  public ResultSet getTables(String catalog, String
  schemPattern, String tableNamePattern,
  String[] types) throws Exception {
        return metaData.getTables(catalog, schemPattern,
        tableNamePattern, types);
  }

  public String getNumericFunctions() throws Exception {
        return metaData.getNumericFunctions();
```

```
}

public String getStringFunctions() throws Exception {
      return metaData.getStringFunctions();
}

public String getTimeDateFunctions() throws Exception {
      return metaData.getTimeDateFunctions();
}

public String getSystemFunctions() throws Exception {
      return metaData.getSystemFunctions();
}

public ResultSet executeQuery(String sql) throws
Exception {
      Statement stmt = conn.createStatement();
      return stmt.executeQuery(sql);
}
```

完全なソースコード
アプレットとしてシステム実行する（HTTP プロトコルを使用しなければならない）

上記のコードはアプリケーションとアプレットのどちらとしても使用することができます。

11.5.6.5 ディレクトリを事前設定してチャートデザイナーを開く

チャートをロードおよび保存するディレクトリを設定することもできます。この機能によって、ユーザは指定されたディレクトリより上のレベルに移動できなくなるため、ユーザが表示できるファイルを制御することができます。

以下に、ブラウズできるルートディレクトリを指定することによってチャートデザイナーをカスタマイズする方法を示した例を示します。

```
public void doChartDesignerApplet(Object parent) {
      // Connect to EspressManager
      QQbChartDesigner.setServerAddress("127.0.0.1");
      QbChartDesigner.setServerPortNumber(22071);
```

```
            String[] imagesPath = {"../../../images",
            "../../../backgroundImages"};

            designer = new QbChartDesigner(parent,
                  "help/manual/code/templates/API_10_5_6_5_CDWi
            thPreSetDirectories.cht", imagesPath);
            designer.setRootDirectoryForBrowse(".");

            designer.setVisible(true);
      }
```

完全なソースコード
アプレットとしてシステム実行する（HTTP プロトコルを使用しなければならない）

上記のコードはアプリケーションとアプレットのどちらとしても使用することができます。

上記の方法に加えて、以下のメソッドを使用して、チャートデザイナーが使用するデフォルトディレクトリを入手することもできます。

```
public BrowseDirectories getBrowseDirectories();
```

この後、BrowseDirectories 内でメソッドを使用して、異なるブラウズダイアログのディレクトリのデフォルトの位置を入手することができます。

11.5.6.6 メニューバーおよびツールバーを変更してチャートデザイナーを開く

メニューに項目を追加したり、ツールバーから項目を削除することもできます。以下に、その例を示します。

```
public void doCustomizeDesignerMenuApplet(Object parent)
{
      // Connect to EspressManager
      QbChartDesigner.setServerAddress("127.0.0.1");
      QbChartDesigner.setServerPortNumber(22071);

      String[] imagesPath = {"../../../images",
      "../../../backgroundImages"};

      designer = new QbChartDesigner(parent,
```

```
            "help/manual/code/templates/API_10_5_6_6_CDWithModi
            fiedBars.cht", imagesPath);

            // Add a new menu item
            JMenuBar menuBar = designer.getChartMenuBar();
            JMenu fileMenu = menuBar.getMenu(0);
            newItem = new JMenuItem("Hello World");
            newItem.addActionListener(this);
            fileMenu.insert(newItem, 7);

            // Remove toolbar buttons
            JToolBar designBar = designer.getChartToolBar();
            designBar.remove(5); // Remove Separator
            designBar.remove(4); // Remove Export
            // Do not prompt to save the chart if unsaved on
            exiting Designer
            designer.setSaveOnExitEnabled(false);

            designer.setVisible(true);
      }

      public void actionPerformed(ActionEvent e) {
            /*** save report **********/
            designer.save("API_10_5_6_6_CDWithModifiedBars_Temp
            .cht");

            /**** create new testing frame *********/
            JPanel contentPane = new JPanel();
            contentPane.setLayout(new BorderLayout());
            contentPane.add(new JLabel("Hello World!"),
            "Center");
            JFrame frame = new JFrame();
            frame.setContentPane(contentPane);
            frame.pack();
            frame.setVisible(true);
      }
```

完全なソースコード
テンプレートファイル
アプレットとしてシステム実行する（HTTP プロトコルを使用しなければならない）

上記のコードはアプリケーションとアプレットのどちらとしても使用することができます。

11.6API の追加機能

EspressChart API はチャートデザイナーのすべての機能を再現できます。また、EspressChart API は以下のような追加機能も備えています。以下では、これらの追加機能について説明します。一部の機能は、スタンドアロンチャートにのみ使用できることに注意してください。

掲載するコードスニペットでは、チャートは QbChart 型オブジェクトであることに注意してください。

11.6.1 表示方法

以下に示す機能は、チャートとそのコンポーネントの表示方法に関するもので、それぞれ、何らかの形でチャートの表示の仕方を変更します。これらの変更は、静的イメージにエクスポートされるときに実行されます。

11.6.1.1 キャンバス/描画

以下では、チャートプロットまたはキャンバス、あるいはその両方の表示方法を変更するために使用できる API 機能について説明します。以下の説明は、チャートプロットまたはキャンバス、あるいはその両方に対する一般的な変更に関するものであることに注意してください。各コンポーネント固有の変更については、個別のセクションで説明しています。

11.6.1.1.1 描画するデータがない場合のメッセージの変更

描画するデータがない、またはデータが足りない場合に表示されるメッセージを変更できます。このためには、INoDataToPlotMessage のハンドルを取得し、新しいメッセージを指定します。

```
INoDataToPlotMessage noData = chart.gethNoDataToPlotMessage();
noData.setMessage("Not enough data to plot chart");
```

詳細はオンライン API ドキュメント（quadbase.util. INoDataToPlotMessage）を参照してください。

11.6.1.1.2 チャートをキャンバス内にフィット

チャートがキャンバス内に適切に収まるよう、サイズを変更できます。チャートに付随するテキストやラベルも正しくサイズ変更されます。チャートをキャンバス内に収

めるには、ICanvas インタフェース内で setFitOnCanvas(boolean b)メソッドを使用します。

```
ICanvas canvas = chart.gethCanvas();
canvas.setFitOnCanvas(true);
```

詳細はオンライン API ドキュメント（quadbase.util. ICanvas）を参照してください。

11.6.1.1.3 チャートを不可視にする

チャートを不可視にし、表形式の描画データのみを表示するよう設定できます。チャートを不可視にするには、ICanvas インタフェースのハンドルを取得して、setChartVisible(boolean b)メソッドを使用します。

```
ICanvas canvas = chart.gethCanvas();
canvas.setChartVisible(false);
```

詳細はオンライン API ドキュメント（quadbase.util. ICanvas）を参照してください。

11.6.1.1.4 デフォルト以外のレンダリング手法の適用

アンチエイリアスなど、デフォルト以外のレンダリング手法を適用できます。これによってユーザが独自に指定した手法でチャートを描画できます。

API でこの機能を使用するには、setRenderingHint を呼び出して hint key と hint value パラメータを引き渡します。

```
chart.setRenderingHint(java.awt.RenderingHints.KEY_ANTIAL
IASING, java.awt.RenderingHints.VALUE_ANTIALIAS_ON);
```

詳細はオンライン API ドキュメント（quadbase. ChartAPI.QbChart）を参照してください。

この機能は、スタンドアロンチャートにのみ使用できます。

11.6.1.1.5 チャートの描画位置

チャートのキャンバスに対する x および y 位置を-0.5 下げて配置できます。これによって、チャートはキャンバスの端ぴったりに配置されます。

API でこの機能を使用するには、IPlot のハンドルを取得して setPosition メソッドを使用します。

```
IPlot chartPlot = chart.gethChartPlot();
chartPlot.setPosition(new Position(-0.5, -0.5));
```

詳細はオンライン API ドキュメント（quadbase.util.IPlot）を参照してください。

11.6.1.1.6 一つの座標上で複数のチャートを描画

主チャートの軸を使用して、複数のチャートをオーバーラップ表示できます。主チャートのキャンバスが使用されることに注意してください。主チャート上にオーバーラップ表示されるチャートはすべて、テキストおよびキャンバス情報が削除されます。

API でこの機能を使用するには、QbChart 内で setAddOnChart メソッドを使用します。

以下のコードは、アプレットまたはアプリケーションとして実行することができ、1 つの座標で 3 つのチャートをオーバーラップ表示する方法を示しています。

```
// Open the template
QbChart primaryChart = new
QbChart(parent,                                                    //
container
                                          "../templates/API_10_6
_1_1_6_AddOnChart1.cht"); // template
// Open other two templates
QbChart chart2 = new QbChart(parent,
"../templates/API_10_6_1_1_6_AddOnChart2.cht");
QbChart chart3 = new QbChart(parent,
"../templates/API_10_6_1_1_6_AddOnChart3.cht");

primaryChart.setAddOnChart(new QbChart[] {chart2,
chart3});
```

完全なソースコード
最初のテンプレート
2 番目のテンプレート
3 番目のテンプレート
エクスポートされた結果

詳細はオンライン API ドキュメント（quadbase. ChartAPI.QbChart）を参照してください。

11.6.1.2 ヒントボックス

以下では、チャートヒントボックスの表示方法を変更するために使用できる API 機能について説明します。このような変更としては、ヒントボックスの内容と見え方の変更があります。

11.6.1.2.1 ヒントボックスの変更

EspressChart では、ヒントボックスを変更することができます。すなわち、ヒントボックスを表示する際、その内容を指示することができます。このためには、IHingBoxInfo を実装するクラスを作成し、IHint 内でメソッド setHintBoxInfo を使用してそのクラスを割り当てます。

以下のコードは、アプレットまたはアプリケーションとして実行することができ、ヒントボックス情報の変更方法を示しています。

```
// Open the template
QbChart chart = new
QbChart(parent,
// container
                              "../templates/API_10_6_1_2_1_
ModifyHintBox.cht"); // template

IHintBoxInfo hintBoxInfo = new Hint();
chart.gethDataPoints().gethHint().setHintBoxInfo(hintBoxI
nfo);

...

class Hint implements IHintBoxInfo {

      public Vector getHint(PickData pickData) {
            Vector vec = new Vector();

            if (pickData.series != null)

                  vec.addElement("(" +
                  pickData.seriesName+ ") " +
                  pickData.s_series);

            if (pickData.category != null)

                  vec.addElement("(" +
                  pickData.categoryName + ") " +
                  pickData.s_category);

            if (pickData.s_value != null)

                  vec.addElement("(" + pickData.valueName
                  + ") " + pickData.s_value +
                  (pickData.value > 7 ? ":Good" :
                  ":Restock"));
```

```
                return vec;

            }

        }
```

完全なソースコード
チャートテンプレート
エクスポートされた結果

詳細はオンライン API ドキュメント（quadbase.util. IHintBoxInfo）を参照してください。

この機能は、スタンドアロンチャートにのみ使用できます。

11.6.1.2.2 データおよびハイパーリンクヒントボックスのオフセット

EspressChart では、データおよびハイパーリンクヒントボックスのオフセットを設定し、チャートや他のオブジェクトと重ならないように配置できます。

API でこの機能を使用するには、チャートオブジェクトまたは IHyperLinkSet から IHint のハンドルを取得し、setOffset メソッドを使用します。

```
IHint dataHints = chart.gethDataPoints().gethHint();
dataHints.setoffset(new Dimension (30, 20));
```

詳細はオンライン API ドキュメント（quadbase.util.IHint）を参照してください。

11.6.1.2.3 ヒントボックスの境界色

EspressChart では、API を使用して、ヒントボックスの境界色を設定できます。このためには、IHint のハンドルを取得して setBorderColor メソッドを使用します。

```
IHint chartHintBox = chart.gethHint();
chartHintBox.setBorderColor(Color.red);
```

詳細はオンライン API ドキュメント（quadbase.util.Ihint）を参照してください。

11.6.1.2.4 イメージマップのヒントボックスのカスタマイズ

EspressChart では、チャートをマップファイルにエクスポートする際のヒントボックスのテキストをカスタマイズすることができます。マップファイルが

DHTML/HTML ファイルに含まれている場合、カスタマイズされるヒントボックスは、マウスがチャートのデータポイント上に移動されたときに表示されます。カスタマイズされたイメージマップヒントボックスを作成するには、ICustomizedImageDataMapHintBox を実装するクラスを作成し、QbChat 内で setImageMapDataHintBoxHandle メソッドにそのクラスを割り当てます。

以下のコードは、アプレットまたはアプリケーションとして実行することができ、イメージマップヒントボックス情報のカスタマイズ方法を示しています。

```
// Open the template
QbChart chart = new
QbChart(parent,
// container
                              "../templates/API_10_6_1_2_4_
CustomizeImageMapHintBox.cht"); // template

chart.setImageMapDataHintBoxHandle(new
customizeImageMap());

try {
      chart.export(QbChart.PNG,
      "../export/API_10_6_1_2_4_CustomizeImageMapHintBox",
      "../export/API_10_6_1_2_4_CustomizeImageMapHintBox",
      0, 0, true);
} catch (Exception ex) {
      ex.printStackTrace();
}

...

class customizeImageMap implements
ICustomizeImageMapDataHintBox {
      public String customizeHintBox(String str) {
            return str.substring(6, 11) + " " +
            str.substring(str.length()-3);
      }

}
```

完全なソースコード
チャートテンプレート
エクスポートされたマップ
エクスポートされた結果

詳細については、オンラインの API ドキュメンテーション（quadbase.util.ICustomizeImageDataHintBox）を参照してください。

11.6.1.3 凡例/注釈

以下では、チャートの凡例または注釈、あるいはその両方の表示方法を変更するために使用できる API 機能について説明します。このような変更としては、チャートの凡例または注釈の内容と見え方の変更があります。

11.6.1.3.1 シンボル付き注釈

EspressChart では、注釈にシンボルを含めることにより、シンボルとストリングをチャート内に配置できます。この機能を使用すると、EspressChart が作成するデフォルトの凡例の代わりにユーザ独自の凡例を作成できます。

これを使用してシンボルおよびテキストを追加するために使用できる IAnnotation 用の新しいコンストラクタが作成されました。

以下のコードは、アプレットまたはアプリケーションとして実行することができ、シンボルと注釈を組み合わせる方法を示しています。

```
// Open the template
QbChart chart = new
QbChart(parent,
       // container
                              "../templates/API_10_6_1_3_1_
AnnotationWithSymbol.cht"); // template

// Create custom legend
IAnnotationSet set = chart.gethAnnotations();
String[] text = {"New Legend", "Hello World", "ABC", "I
got It"};
int[] shape = {QbChart.PLUS, QbChart.NOSYMBOL,
QbChart.SQUARE, QbChart.DASH};
Color[] color = {Color.red, Color.black, Color.blue,
Color.white};
IAnnotation anno = set.newAnnotation(text, shape, color);
Point_2D newPosition = new Point_2D(.7f, .7f);
anno.setRelativePosition(newPosition);
set.addAnnotation(anno);
```

完全なソースコード
チャートテンプレート
エクスポートされた結果

詳細はオンライン API ドキュメント（quadbase.util. IannotationSet および quadbase.util.IAnnotation）を参照してください。

11.6.1.3.2 凡例と注釈テキストの基準位置の設定

凡例と注釈テキストの基準位置を、左上端またはデフォルト位置（左下端）に設定できます。基準位置を左上端に変更するには、ICanvas 内で setReferenceAtTop(boolean b)メソッドを使用します。

```
ICanvas canvas = chart.gethCanvas();
canvas.setReferenceAtTop(true);
```

詳細はオンライン API ドキュメント（quadbase.util. ICanvas）を参照してください。

11.6.1.4 その他

以下では、チャートのさまざまなコンポーネントの表示方法を変更するために使用できる機能について説明します。このような変更としては、チャートおよびその要素の内容と見え方の変更があります。

11.6.1.4.1 ティッカーラベルの置き換え

ティッカーラベルをユーザ定義のストリングに置き換えられます。これによってユーザが独自のティッカーバリューを設定できます。

ティッカーラベルを置き換えるには、IAxis 内で setTickerLabels メソッドを使用します。

```
IAxis hXAxis=chart.gethXAxis();
hAxis.setTickerLabels(new String[] {new String("a")),"), new
String("b"),new String("c")});
```

詳細はオンライン API ドキュメント（quadbase.util. IAxis）を参照してください。

11.6.1.4.2 データトップラベルのカスタマイズ

EspressChart では、プライマリおよびセカンダリの両方のチャートのデータトップラベルをカスタマイズできます。このためには、IDataLabelInfo を実装するクラスを作成し、IDataPointSet 内で setDataLavelInfo メソッドを使用して、そのクラスを渡します。

以下のコードは、アプレットまたはアプリケーションとして実行することができ、データトップラベルをカスタマイズする方法を示しています。

```
// Open the template
QbChart chart = new
QbChart(parent,
        // container
                                "../templates/API_10_6_1_4_2_
CustomizeDataTopLabel.cht"); // template
chart.gethDataPoints().setDataLabelInfo(new
customizeDataTopLabel());

...

class customizeDataTopLabel implements IDataLabelInfo {
      public String getDataLabel(PickData pickData,
      String originalDataLabel) {
            return (pickData.toString() + " (" +
            originalDataLabel + ")")
      }

}
```

完全なソースコード
チャートテンプレート
エクスポートされた結果

データトップラベルのカスタマイズは、データトップラベルが表示される場合のみ反映されます。

詳細はオンライン API ドキュメント（quadbase.util. IDataLabelInfo）を参照してください。

11.6.1.4.3 値軸を日付／時間／タイムスタンプとして表示

EspressChart では、チャートに値軸がある場合、値軸を数字で表示する代わりに日付または時間として表示できます。この場合でも、値軸のデータは数値でなければならないことに注意してください。

API でこの機能を使用するには、IAxis のハンドルを取得して setDisplay LabelAsDate メソッドを使用します。

```
IAxis chartYAxis = chart.gethXAxis();
chartXAxis.setDisplayLabelAsDate(true);
```

詳細はオンライン API ドキュメント（quadbase.util.IAxis）を参照してください。

11.6.1.4.4 文字列のレンダリングの選択

水平または垂直に描画される文字列は、アンチエイリアスなしの方がきれいに表示されます。しかし他の角度で描画される文字列は、アンチエイリアスありの方がきれいに表示されます。EspressChart では、0 度または 90 度のストリングのみをアンチエイリアスなしで描画し、他の角度のテキストはアンチエイリアスありの状態で描画するよう設定できます。

API でこの機能を使用するには、IDataPointSet のハンドルを取得して setDisableJava2DForStraightText メソッドを使用します。

```
IDataPointSet dataPoints = chart.gethDataPoints();
dataPoints.setDisableJava2DForStraightText(true);
```

詳細はオンライン API ドキュメント（quadbase.util. IDataPointSet）を参照してください。

11.6.1.4.5 データポイントを水平／垂直／トレンドラインの上に描画

データポイントを水平／垂直／トレンドラインの上に描画できます。これによってデータポイントはラインの下に隠れることなく、ラインの上に乗るようにして表示されます。

API でこの機能を使用するには、ILinePropertySet のハンドルを取得して setDataDrawnOnTop メソッドを使用します。

```
ILinePropertySet lineProperties = chart.gethLineProperties();
lineProperties.setDataDrawnOnTop(true);
```

詳細はオンライン API ドキュメント（quadbase.util.Iline PropertySet）を参照してください。

11.6.2 データ

以下に示す機能は、チャートとそのコンポーネントのデータに利用できる機能です。以下では、表示されるデータ（通常、その表示方法）を変更するか、またはチャート内のデータを取得するための機能について説明します。これらの変更は、静的イメージにエクスポートされるときに実行されます。

11.6.2.1 座標の取得

データポイントの情報を、指定された位置に基づいて取得できます。指定された位置にデータポイントが存在する場合、そのデータポイントのカテゴリ、バリュー、シリーズ、総計が返されます。指定された位置にデータポイントがない場合は null が返されます。

pickData を取得するには、IHint の getPickData(width, height)メソッドを使用します。

```
IDataPointSet dataPoints=chart.gethDataPoints();
IHint hint=dataPoints.gethHint();
PickData pickdata=hint.getPickData(200, 300)
// pixel at width=200 and height=300
```

詳細はオンライン API ドキュメント（quadbase.util. IHint）を参照してください。

11.6.2.2 軸スケールでのデータリミットの設定

手動設定の軸が選択された場合、軸の最大値を超えたデータポイントを切り捨てるよう設定できます。このようなデータリミットを設定するには、IDataPointSet 内で setLimitAtAxisScale(boolean b)メソッドを使用します。

```
IDataPointSet dataPoints = chart.gethDataPoints();
dataPoints.setLimitAtAxisScale(true);
```

詳細はオンライン API ドキュメント（quadbase.util. IDataPointSet）を参照してください。

11.6.2.3 Null データを 0 に設定

null データを 0 データとして表示できます。これによって、例えば null データのデータポイントを 0 として表示できます。null データを 0 に設定するには、quadbase.util.IDataPointSet から以下の API の呼び出しを使用します。

```
IDataPointSet dataPoints = chart.gethDataPoints();
dataPoints.setNullDataAsZero(true);
```

詳細はオンライン API ドキュメント（quadbase.util. IDataPointSet）を参照してください。

11.6.2.4 トレンドラインの追加オプション

正規曲線トレンドラインの平均値、最小値、最大値、確率、逆正規の取得が行えます。また、標準偏差トレンドラインを描画できます。これらを実行するには ITrendLine を呼び出し、適切なメソッドを使用します。

API でこの機能を使用するには、ITrendLine を呼出して適切なメソッドを使用します。

正規曲線の標準偏差トレンドラインは、チャートがヒストグラムチャートで setLinearScale と setRounded が true である場合のみ描画できます。

以下のコードは、アプレットまたはアプリケーションとして実行することができ、チャート内の正規曲線のトレンドラインから情報を入手する方法を示しています。

```
ITrendLine normalCurve =
(ITrendLine)chart.gethDataLines().elements().nextElement(
);
System.out.println("Mean = " + normalCurve.getMean());
System.out.println("St. Dev = " +
normalCurve.getStandardDev());
System.out.println("Min = " + normalCurve.getMin());
System.out.println("Max = " + normalCurve.getMax());
System.out.println("Dev of 1.0, Prob = " +
normalCurve.getProbability(1.0));
System.out.println("Back Calculated Dev = " +
normalCurve.getInverseNorm(normalCurve.getProbability(1.0
)));
```

完全なソースコード
チャートテンプレート
エクスポートされた結果

詳細はオンライン API ドキュメント（quadbase.util. ITrendLine）を参照してください。

11.6.3 チャートの種類別

以下に示す機能は、チャートの種類に応じて利用できる API 追加機能です。以下では、各機能について説明します。これらの変更も、静的イメージにエクスポートされるときに実行されます。

11.6.3.1 カラム/バーチャート

以下では、カラム/バーチャートの表示方法を変更するために使用できる API 機能について説明します。このような変更としては、チャートの内容と見え方の変更があります。

11.6.3.1.1 カラーセパレータ

カラーセパレータを使用すると、定義されたカテゴリバリューに基づいてカラムを異なる色で表示できます。カラーセパレータを使用するには、IDataPointSet 内で setColorSeparators メソッドを使用します。

以下のコードは、アプリケーションまたはアプレットとして実行することができ、カラーセパレータの設定方法を示しています。

```
IDataPointSet          hDataPoints=chart.gethDataPoints();
hDataPoints.setColorSeparators(new    Color[]{Color.green,
Color.red,                                    Color.blue},
                                  new          Integer[]{new
Integer((int)Math.rint(9)),                              new
Integer((int)Math.rint(14))},
                                  QbChart.ASCENDING);
```

完全なソースコード
チャートテンプレート
エクスポートされた結果

詳細はオンライン API ドキュメント（quadbase.util. IDataPointSet）を参照してください。

11.6.3.1.2 影の非表示

カラムおよびバーチャートで表示される影を非表示にできます。この機能を使用するには、IDataPointSet のハンドルを取得して setShadowOnPoint メソッドを使用します。

```
IDataPointSet dataPoints = chart.gethDataPoints();
dataPoints.setShadowOnPoint(false);
```

詳細はオンライン API ドキュメント（quadbase.util. IDataPointSet）を参照してください。

11.6.3.2 パイチャート

以下では、パイチャートの表示方法を変更するために使用できる API 機能について説明します。このような変更としては、チャートの内容と見え方の変更があります。

11.6.3.2.1 パイスライスの描画方向

パイスライスの描画方向を時計方向または反時計方向に設定できます。パイチャートの描画方向を設定するには、IDataPointSet 内で reverseOrder メソッドを使用します。

```
IDataPointSet dataPoints = chart.gethDataPoints();
dataPoints.reverseOrder(QbChart.CATEGORY);
```

詳細はオンライン API ドキュメント（quadbase.util. IDataPointSet）を参照してください。

11.6.3.2.2 0%または100%スライスの境界線

パイチャートにおいて、パイ全体の 0%または 100%を占めるスライスの境界線を削除できます。API でこの機能を使用するには、IPiePropertySet のハンドルを取得して setRadialBorderDrawnForZero メソッドを使用します。

```
IPiePropertySet pieProperties = chart.gethPieProperties();
pieProperties.setRadialBorderForZero(true);
```

詳細はオンライン API ドキュメント（quadbase.util. IPiePropertySet）を参照してください。

11.6.3.2.3 凡例内のカテゴリ文字列とパーセント値文字列間のセパレータのカスタマイズ

パイチャートの凡例内のカテゴリ文字列とパーセント値文字列の間にユーザ定義セパレータを配置できます。この機能を使用するには、IPiePropertySet のハンドルを取得して、setSepSymbol メソッドを使用します。

```
IPiePropertySet pieProperties = chart.gethPieProperties();
pieProperties.setSepSymbol(",");
```

詳細はオンライン API ドキュメント（quadbase.util. IPiePropertySet）を参照してください。

11.6.3.2.4 パイの境界色のカスタマイズ

EspressChart では、パイチャートのパイの境界色を指定できます。このためには、IPiePropertySet 内で setBorderColor メソッドを使用します。

```
IPiePropertySet pieProperties = chart.gethPieProperties();
pieProperties.setBorderColor(Color.red);
```

詳細はオンライン API ドキュメント（quadbase.util.Ipie PropertySet）を参照してください。

11.6.3.3 ラインチャート

以下では、ラインチャートの表示方法を変更するために使用できる API 機能について説明します。このような変更としては、チャートの内容と見え方の変更があります。

11.6.3.3.1 ラインエリア

データラインと任意の水平ラインとの間にラインエリアを作成し、データの変化をより明確に示すことができます。例えば 25 の位置にある水平ラインと、データラインを作成するとします。この水平ラインとデータラインとで囲まれたエリアは、水平ラインより上のエリアと下のエリアを異なるカラーで塗りつぶせます。この機能はシリーズを伴わない 2D ラインチャートでのみ使用できます。また、水平ラインの上と下のカラーを設定する必要があります。

この機能を API で使用するには、ILinePropertySet のハンドルを取得して setAreaVisible および setAreaColors メソッドを使用します。

以下のコードは、アプレットまたはアプリケーションとして実行することができ、ラインエリアの使用方法を示しています。

```
ILinePropertySet lineProperties =
chart.gethLineProperties();
lineProperties.setAreaVisible(true);
lineProperties.setAreaColors(Color.green, Color.yellow)
```

完全なソースコード
チャートテンプレート
エクスポートされた結果

詳細はオンライン API ドキュメント（quadbase.util. ILinePropertySet）を参照してください。

11.6.3.4 分散チャート

以下では、分散チャートの表示方法を変更するために使用できる API 機能について説明します。このような変更としては、チャートの内容と見え方の変更があります。

11.6.3.4.1 トップラベルでのシリーズ表示

分散チャートでは、トップラベルでシリーズを表示できます。トップラベルでシリーズを表示するには、IDataPointSet の showSeriesInTopLabel メソッドを使用します。

```
IDataPointSet dataPoints=chart.gethDataPoints();
dataPoints.showSeriesInTopLabel(true);
```

詳細はオンライン API ドキュメント（quadbase.util. IDataPointSet）を参照してください。

11.6.3.4.2 描画順序

分散チャートにおいて、ラインの接続をデータセットの順序に合わせて描画できます。例えばデータのポイントが(0,2)、(3, 4)、(1, 2)、(2, 5)である場合、デフォルトのラインの接続は(0,2)から(1, 2)、(2, 5)、(3 ,4)という順序で行われます。この順序をデータの順序に合わせることが可能です。

API でこの機能を使用するには、IDataPointSet のハンドルを取得して setConnectLinesInOriginalOrder メソッドを使用します。

```
IDataPointSet dataPoints = chart.gethDataPoints();
dataPoints.setConnectLinesInOriginalOrder(true);
```

詳細はオンライン API ドキュメント（quadbase.util. IDataPointSet）を参照してください。

11.6.3.5 オーバーレイチャート

以下では、オーバーレイチャートの表示方法を変更するために使用できる API 機能について説明します。このような変更としては、チャートの内容と見え方の変更があります。

11.6.3.5.1 複数の軸タイトル

EspressCart では、オーバーレイチャート内の軸ごとに異なるタイトルを設定することができます。このためには、個々の軸のハンドルを取得し、gethTitle().setValue メソッドを使用します。個々の軸のハンドルを取得するには、レイヤを指定します。

```
IAxis axis0 = chart.gethYAxis();
axis0.gethTitle().setValue("XYZ");
IAxis axis1 = chart.gethAxis(1); // Get axis of Layer 1
axis1.gethTitle().setValue("ABC");
IAxis axis2 = chart.gethAxis(2); // Get axis of Layer 2
axis2.gethTitle().setValue("DEF");
```

タイトルを設定する前に、バックグラウンドでチャートを描画する必要があることに注意してください。

詳細については、オンラインの API ドキュメンテーション（quadbase.util.IAxis）を参照してください。

11.6.3.6 ダイアルチャート

以下では、ダイアルチャートの表示方法を変更するために使用できる API 機能について説明します。このような変更としては、チャートの内容と見え方の変更があります。

11.6.3.6.1 制御領域のスケールラベル

EspressChart では、制御領域の開始および終了スケールをラベルとして表示することができます。このためには、制御領域のハンドルを取得し、setShowLabel メソッドを使用します。

```
ControlRange cr1 = chart.gethControlRanges().elementAt(0);
cr1.setShowLabel(true);
```

詳細については、オンラインの API ドキュメンテーション（quadbase.util.ControlRange）を参照してください。

11.6.3.7 HLCO チャート

以下では、HLCO チャートの表示方法を変更するために使用できる API 機能について説明します。このような変更としては、チャートの内容と見え方の変更があります。

11.6.3.7.1 キャンドルスティックのカラーの変更

API の追加機能を使用して、HLCO チャートのキャンドルスティックのカラーを変更できます。カラーを変更するには IDataPointSet のハンドルを取得し、setCandleStickColors メソッドを使用します。

```
IDataPointSet dataPoints = chart.gethDataPoints();
// Up color is green, Down color is red
dataPoints.setCandleStickColors(Color.green, Color.red);
```

詳細はオンライン API ドキュメント（quadbase.util. IDataPointSet）を参照してください。

11.6.3.7.2 キャンドルスティックの上端および下端幅の変更

API の追加機能を使用して、HLCO チャートのキャンドルスティックの上端および下端の幅を変更できます。幅の変更は IDataPointSet を取得し、setCandleStickWidth メソッドを使用して行います。引き渡される数はキャンドル幅に対する上端および下端幅の比率です。

```
IDataPointSet dataPoints = chart.gethDataPoints();
// Set width to 0.5
dataPoints.setCandleStickWidth((float)0.5);
```

詳細はオンライン API ドキュメント（quadbase.util. IDataPointSet）を参照してください。

11.6.3.8 サーフェイスチャート

以下では、サーフェイスチャートの表示方法を変更するために使用できる API 機能について説明します。このような変更としては、チャートの内容と見え方の変更があります。

11.6.3.8.1 ヒートマップ

EspressChart では、ユーザが等高線図のようなサーフェイスチャートを描画することができます。基本的に、サーフェスチャートはセクションごとに分かれた状態で描画されます。各セクションは指定されたしきい値に従って異なるカラーで表示されます。

以下のコードは、アプレットまたはアプリケーションとして実行することができ、ヒートマップを使用したサーフェイスチャートの作成方法を示しています。

```
double [] heatMapValues = {3, 6};
Color [] heatMapColors = { Color.green, Color.yellow,
Color.red};
ColorSpectrum heatMapColorSpectrum = new
ColorSpectrum(heatMapColors, heatMapValues);
```

```
I3DPropertySet set = chart.geth3DProperties();
set.setColorSpectrum(heatMapColorSpectrum);
```

上記のコードはグリーン、イエロー、レッドの 3 色の各セクションを持つ 3D サーフェスチャートを作成します。カラーを決定するしきい値は 3 と 6 です。

完全なソースコード
チャートテンプレート
エクスポートされた結果

詳細はオンライン API ドキュメント（quadbase.util. ColorSpectrum および quadbase.util.I3DPropertySet）を参照してください。

11.6.4 パフォーマンス

以下に示す機能は、チャートの作成およびエクスポートのパフォーマンスを向上させるためのものです。以下では、チャートを作成するために必要なメモリリソースおよび時間を削減するためにチャートに利用できる各追加機能について説明します。

11.6.4.1 BufferedImage または Frame の使用によるエクスポート時のパフォーマンスの向上

チャートのエクスポート時に java.awt.image.BufferedImage または java.awt.Frame を使用して、パフォーマンスを向上できます。デフォルトでは、チャートオブジェクトの作成には java.awt.Frame が使用されます。java.awt.Frame はデータポイントの数が多い場合によりパフォーマンス向上の効果があります。java.awt.image.BufferedImage はチャートが 3D の場合によりパフォーマンス向上の効果があります。このためには、QbChart 内で setBufferedImageUsed メソッドを使用します。

```
chart.setBufferedImageUsed(true);
```

詳細はオンライン API ドキュメント（quadbase.ChartAPI.QbChart）を参照してください。

この機能は、スタンドアロンチャートにのみ使用することができます。

11.6.4.2 チャートコンポーネントの順番の指定

チャートが生成される順番の選択や、API のみを使用してチャートを生成する際のエレメントの追加を行えます。例えばまず背景を描画し、次に円を描画してその円の中心にチャートの生成を実行できます。このためには、IChartGraphics インタフェースを実装するクラスを作成した後、QbChart 内で setChartGraphics メソッドにそのクラスを割り当てます。要するに、IChartGraphics インタフェースを使用すると、チャートを描画する前または後にグラフィックス情報を追加または変更することができます。

以下のコードは、アプレットまたはアプリケーションとして実行することができ、指定した順序でチャートおよびグラフィックスを生成する方法を示しています。

```
// Open the template
QbChart chart = new
QbChart(parent,
// container
                              "../templates/API_10_6_4_2_Ch
artGeneration.cht"); // template

chart.setChartGraphics(new chartGenerationGraphics());;

...

class chartGenerationGraphics implements IChartGraphics {

      public void initializeGraphics(Graphics g, int w,
      int h) {
            g.setColor(Color.red);
            g.fillOval(50, 50, 400, 400);

      }

      public void finalizeGraphics(Graphics g, int w, int
      h) {

            g.setColor(Color.white);
            g.fillOval(125, 225, 50, 50);
            g.setColor(Color.orange);
            g.drawString("HELLO WORLD", 150, 250);

      }

}
```

完全なソースコード
チャートテンプレート
エクスポートされた結果

詳細はオンライン API ドキュメント（quadbase.util. IChartGraphics）を参照してください。

この機能は、スタンドアロンチャートにのみ使用することができます。

11.6.5 ビューワ

以下に示す機能は、java アプリケーションまたは java アプレット内でチャートを表示する際のビューワに関するものです。以下では、チャートビューワに利用できる追加機能について説明します。

11.6.5.1 コールバックメカニズム

EspressChart はより高いレベルのイベントを操作するためのコールバックメカニズムを備えています。チャート内のデータオブジェクトが選択されると、アクションイベントが生成されます。イベントアーギュメントは PickData インスタンスを含んでおり、これによって選択されたデータポイントのシリーズ、カテゴリ、バリューなどの情報が供給されます。

以下のコードは、アプレットまたはアプリケーションとして実行することができ、イベントをキャプチャする方法を示します。

```
static TextField textField;

// Open the template
QbChart chart = new
QbChart(parent,                                    //
container
                              "../templates/API_10_6_5_1_Ca
llBack.cht"); // template

chart.setActionListener(new callBackActionListener());;

...

class callBackActionListener implements ActionListener {

      public void actionPerformed(ActionEvent e) {
            Object         arg         =         ((QbChart)
            e.getSource()).getArgument();

            String click;
```

```
                switch (e.getModifiers()) {

                case QbChart.LEFT_SINGLECLICK:
                      click    =    "Left    single    click";
                      break;
                case QbChart.LEFT_DOUBLECLICK:
                      click    =    "Left    double    click";
                      break;
                case QbChart.RIGHT_SINGLECLICK:
                      click    =    "Right    single    click";
                      break;
                case QbChart.RIGHT_DOUBLECLICK:
                      click    =    "Right    double    click";
                      break;
                default: // shall not happen
                      click = "Error !";
                }

                if (arg instanceof PickData)
                      textField.setText(((PickData)
                      arg).toString() + " " + click);
                else
                      textField.setText((String) arg + " " +
                      click);
          }
     }
```

[完全なソースコード]
[チャートテンプレート]
[エクスポートされた結果]

この機能は、スタンドアロンチャートにのみ使用することができます。

11.6.5.2 ツールヒントの表示／非表示

ナビゲーションパネルのツールヒントの表示または非表示を設定できます。このためには、I3DControlPnael 内で setToolTipEnabled(boolean b)メソッドを使用します。このオプションは、3D チャート専用であることに注意してください。

```
I3DControlPanel controlPanel = chart.geth3DControlPanel();
controlPanel.setToolTipEnabled(true);
```

詳細はオンライン API ドキュメント（quadbase.util. I3DControlPanel）を参照してください。

11.6.5.3 キャンバスエリア

キャンバスを表示するビューパネルを選択し、より多くのイベントプロパティ（ユーザ定義のイベントプロパティなど）を追加できます。このためには、ICanvas のハンドルを取得し、getCanvas Area()メソッドを使用してコンポーネントを返します。

```
ICanvas chartCanvas = chart.gethCanvas();
Component chartCanvasComponent = chartCanvas.getCanvasArea();
```

詳細はオンライン API ドキュメント（quadbase.util. ICanvas）を参照してください。

11.7 チャートビューワのオプションの変更

場合によっては、ユーザがチャートビューワを使用してチャートを表示する際に実行できることと実行できないことを設定しなければならないことがあります。

ユーザがチャートビューワを使用してチャートを表示する際、チャート上で右クリックすると、メニューがポップアップされます。ユーザはこのメニューを使用して、チャートタイプの変更やチャート次元の変更などのチャートオプションを選択することができます。さらに、ユーザは、チャートのエクスポートや静的イメージのタイプを選択することもできます。デフォルトのポップアップメニューは、利用可能なオプションをすべてリストしますが、チャートビューワのポップアップメニューで使用できるオプションを制御する API メソッドがあります。

これらの API 呼び出しは、IPopupMenu 内で使用することができます。

```
IPopupMenu popupMenu = chart.gethPopupMenu();
popupMenu.setDimMenuEnabled(boolean b);
popupMenu.setPopupMenuEnabled(boolean b);
popupMenu.setTypeMenuEnabled(boolean b);
```

詳細はオンライン API ドキュメント（quadbase.util.IPopupMenu）を参照してください。

11.8 Javadoc

API 全体の Javadoc が EspressChart と共に提供されています。API は標準および Mini API ライブラリの双方をカバーし、<EspressChart がインストールされているディレクトリ>/help/apidocs ディレクトリに配置されています。

11.9 Swing バージョン

標準 API の 1.1 JFC/Swing バージョンも使用できます。詳細は quadbase.ChartAPI.swing および quadbase.miniapi.swing パッケージを参照してください。

11.10 概要

EspressChart API は、様々なビジネスに応用できるシンプルかつ強力なチャート作成ライブラリです。チャートデザイナーを使用すれば、アプレットまたはアプリケーションに 1 ラインのコードを追加するだけでプログラミングが可能です。すべてのチャート属性は、EspressChart デザイナーで作成するテンプレートファイルで設定できます。EspressChart API の動作確認済みプラットフォームは Windows 95、Windows NT/2000、Solaris、 Linux、AIX、HP、ブラウザは Netscape's Communicator（4.06 以上）、 Microsoft's Internet Explorer（4.x 以上）、Sun's Appletviewer（1.2 以上）です。

12 サーブレットと JavaServer ページ

12.1 サーブレット

12.1.1 イントロダクション

Java サーブレットは、Sun Microsystems™ のウェブサイト(http://java.sun.com) で以下のように紹介されています。「Java サーブレットは、ウェブサーバ機能の拡張や既存のビジネスシステムへのアクセスにシンプルで一貫したメカニズムをもたらす、ウェブ開発者のためのツールです。サーブレットはサーバサイドで実行されるアプレットのように機能します。Java サーブレットによって、様々なウェブアプリケーションが実現できます。

サーブレットはインタラクティブなウェブアプリケーションの構築に幅広く利用されています。サーブレットコンテナは Apache ウェブサーバ、iPlanet ウェブサーバ（旧 Netscape エンタープライズサーバ）、Microsoft IIS などで使用できます。サーブレットコンテナはウェブ上で作動するアプリケーションサーバに統合することも可能で、BEA WebLogic アプリケーションサーバ、IBM WebSphere、Netscape アプリケーションサーバなどで使用できます。」

以下のセクションでは EspressChart のサンプルサーブレットを設定する方法を解説します（サンプルサーブレットは EspressChart/help/examples/servlet/DatabaseJPEG にあります）。以下の各セクションはサンプルのデータベースサーブレットについて説明していますが、ユーザ独自のサーブレットや他のサンプルサーブレットの設定および実行のためのガイドとしても利用できます。

データベースサーブレットに加えて、他のサーブレットのサンプルも用意されています。これらの中にはアプレットまたは静止画像としてチャートの単純な表示を行うものから、チャートのストリーミングを直接ブラウザに送信してドリルダウンチャートを静止画像として表示するものもあります。これらのサンプルの設定と実行を行うには、以下のセクションを参照してください。

12.1.2 設定

- EspressManager が実行されていることを確認してください。
- 使用するサーブレットサーバ（実行環境）に合わせて、以下のガイドラインを実行します。これが完了したら、ブラウザで ExportChart2.html（EspressChart/ help/examples/servlet/DataFile ディレクトリにあります）を開きます。

以下の説明は Windows プラットフォームを例に記載されていますが、ファイル名やパスを Unix/Linux 標準に合わせて変更することにより、Unix/Linux プラットフォームでも同様に使用できます。

以下の例は簡略化のためにシングルスレッドモデルを使用していますが、EspressChart API はマルチスレッド環境でも使用できます。

以下、EspressChart/help/examples/servlet ディレクトリにある DataFile サーブレットを例に解説します。

12.1.3 実行環境

12.1.3.1 Apache Tomcat

1. Tomcat バージョンをチェックする

   インストールされている Tomcat サーバが 5.5.20 または 6.0.10 であることを確認してください（おそらく、本書は古いバージョンにも適用可能です）。

   Windows オペレーティングシステムの場合、以下の 2 つのバージョンのインストール用ファイルがあります。

   - zip ファイル – サーバの起動と停止には、startup.bat および shutdown.bat ファイルを使用します。これらは、アーカイブの抽出後、bin ディレクトリ内にあります。

   - exe ファイル – Windows サービスをインストールし、”Apache Tomcat Monitor”アプリケーションを使用します。

2. 必ず EspressManager を実行する

   <EspressChart のインストールディレクトリ> 内にある”EspressManager.bat”を実行して、EspressManager を実行してください。

3. 仮想ディレクトリをマップする

   例を正しく動作させるには、EspressChart インストールディレクトリを Apache Tomcat のウェブルートの下の仮想ディレクトリとしてマップする必要があります。

   エディタを使用して、xml タグ<Host>と</Host>の間に以下を追加し、変更を保存して、（”<Tomcat のインストールディレクトリ> \conf\"ディレクトリ内にある）"server.xml"ファイルを開きます。

```
<Context   path="/EspressChart"   docBase="<EspressChart
install dir path>" debug="0" privileged="true"/>
```

たとえば、以下のようになります。

```
<Context  path="/EspressChart"  docBase="c:\EspressChart"
debug="0" privileged="true"/>
```

注意: 仮想ディレクトリのマッピングを回避するために、EspressChart を”<Tomcat のインストールディレクトリ> \webapps\ROOT”ディレクトリにインストールすることもできます。

4. クラスパスを設定する

   クラスパスに”EspressAPI.jar”および”servlet-api.jar”ファイルを含めなければなりません。

   EspressAPI.jar ファイルは"EspressChart\lib"ディレクトリ内にあり、servlet-api.jar は"<Tomcat のインストールディレクトリ>\common\lib\"内にあります。

   以下に、クラスパスの設定例を示します（コマンドラインコンソールを使用します）。

   ```
   set classpath=%classpath%;<Tomcat 5.5 のインストールディレクト            リ            >\common\lib\servlet-api.jar;
   C:\EspressChart\lib\EspressAPI.jar (Tomcat 5.5 サーババージョンを使用する場合)
   ```

   ```
   set classpath=%classpath%;<Tomcat 6.0 のインストールディレクト                リ                >\lib\servlet-api.jar;
   C:\EspressChart\lib\EspressAPI.jar (Tomcat 6.0 サーババージョンを使用する場合)
   ```

   注意: クラスパスは必ずしも設定する必要はありませんが、これは”.java”ファイルのコンパイルを簡素化します（そこにクラスパスを追加する必要ありません）。

   注意: MyComputer（マイコンピュータ） -> Administrate（管理） -> Details（詳細）タブウィンドウを使用して Classpath 変数を設定することができます。

5. 例ファイル（ExportServlet2.java）をコンパイルする

コンパイルコマンド:
（本書のステップ4 でclasspath 環境変数を設定しなかった場合）

```
javac -classpath "<Tomcat 5.5 のインストールディレクトリ
>\common\lib\servlet-api.jar;
C:\EspressChart\lib\EspressAPI.jar"  ExportServlet2.java
```

(Tomcat 5.5 サーババージョンの場合)

javac -classpath "<Tomcat 6.0 のインストールディレクトリ>\lib\servlet-api.jar; C:\EspressChart\lib\EspressAPI.jar" ExportServlet2.java *(Tomcat 6.0 サーババージョンの場合)*

または単に、
（classpath が適切に設定された場合）

```
javac ExportServlet2.java
```

6. クラスファイルを Tomcat を配置しているディレクトリにコピーする

   "ExportServlet2.class"ファイルを"<Tomcat のインストールディレクトリ>\webapps\ROOT\WEB-INF\classes"にコピーします。

   注意: デフォルトでは、"<Tomcat のインストールディレクトリ>\webapps\ROOT\WEB-INF\classes"ディレクトリが存在する必要はないため、これを作成しなければなりません。

7. サーブレットを実行するために必要なライブラリを lib ディレクトリに追加する

   この例を実行するには、必要なライブラリを Web アプリケーションの lib ディレクトリに追加する必要があります。"EspressChart\lib"ディレクトリ内にある EspressAPI.jar ファイルを"<Tomcat のインストールディレクトリ>\webapps\ROOT\WEB-INF\lib"ディレクトリにコピーします。

   注意: デフォルトでは、"<Tomcat のインストールディレクトリ>\webapps\ROOT\WEB-INF\lib"ディレクトリが存在する必要はないため、これを作成しなければなりません。

8. サーブレットアプリケーションを web.xml ファイルに登録する

   サーブレットアプリケーションを実行するには、アプリケーションの web.xml ファイルを変更しておく必要があります。このファイルは、"<Tomcat のインストールディレクトリ>\webapps\ROOT\WEB-INF\"ディレクトリ内にあります。

ExportServlet2 を登録するには、xml タグ<web-app>と</web-app>の間に以下のコードを追加します。

```
<servlet>
   <servlet-name>ExportServlet2</servlet-name>
   <servlet-class>ExportServlet2</servlet-class>
</servlet>

<servlet-mapping>
   <servlet-name>ExportServlet2</servlet-name>
  <urlpattern>/servlet/ExportServlet2</url-pattern>
</servlet-mapping>
```

注意: webpp ディレクトリ内にあるすべてのサーブレットを自動的に呼び出すこともできます。この場合、サーブレットアプリケーションを登録する必要はありません。ただし、これはテスト用です。本番環境で invoker サーブレットを使用することはお奨めしません。またサポートもされていません。詳細については、Tomcat サーバのドキュメンテーションを参照してください。

9. Tomcat サーバを起動する

   zip インストールファイルを使用した場合、（<Tomcat のインストールディレクトリ>\bin ディレクトリ内にある）starup.bat（Linux の場合は startup.sh）ファイルを実行します。

   注意: Tomcat サーバがすでに実行されている場合、この例を実行するには、再起動（シャットダウンして再度起動）する必要があります。

10. 例を実行する

    最後のステップは、例の実行です。例を実行するには、単に、ブラウザ内で（*EspressChart\help\examples\servlet\DataFile"* ディレクトリ内にある）**ExportChart2.html** ファイルを実行します。データファイルに対するパスを設定するか、EspressChart をインストールするときに付属していたデフォルトのデータファイルを使用します。”Get Chart Image（チャートイメージの取得）”ボタンをクリックすると、ExportServlet2 が呼び出されて、チャートイメージがエクスポートされ、ページにも表示されます。

    注意: 必ず、EspressManager を実行してから例を実行してください。

    注意: 設計および提供されたサーブレットは、サーブレットランナーとクライアントブラウザの両方が同一のマシン上にある場合のみ動作します。クライアントを別のマシン上に配置できるようにするには、”ExportChart2.html”ファイル内の 1 行を以下のように変更します。

変更前:

```
<form  action="http://localhost:8080/servlet/ExportServlet2"
method="post">
```

変更後:

```
<form
action="http://<machine_name>:8080/servlet/ExportServlet2"
method="post">
```

12.1.3.2 JRun

- html ファイル内のマシン名とポート番号を、JRun サーバを実行しているマシンの名前に変更します。
- `<FORM ACTION=http://<machine name>:8100/servlet/Export Servlet method=POST>`JRun Default Server Administrator にログインします。
- Java Settings をクリックし、次に Classpath をクリックします。
- 個別のラインで EspressAPI.jar と ExportLib.jar へのパスを追加し、 Update をクリックします。
- JRun Default Server を再起動します。
- サーブレットをコンパイルし、 <jrun_installation_directory>/servers/default/default-app/WEB-INF/classes にコピーします。
- 修正した ExportChart2.html をブラウザで開き、パラメータを入力します。ボタンをクリックするとチャートが取得できます。

12.1.3.3 ColdFusion サーバ

- ExportChart2.html の名前を ExportChart2.cfm に変更し、cfm ファイルのマシン名とポート番号を JRun サーバを実行しているマシンの名前に変更します。

  ```
  <FORM ACTION=http://<machine name>:8100/servlet/Export
  Servlet  method=POST>
  ```

- JRun Default Server Administrator にログインします。
- Java Settings をクリックし、Classpath を入力します。
- 個別のラインで EspressAPI.jar と ExportLib.jar へのパスを追加し、 Update をクリックします。
- JRun Default Server を再起動します。
- サーブレットをコンパイルし、 <jrun_installation_directory>/servers/default/default-app/WEB-INF/classes にコピーします。
- 修正した ExportChart2.cfm をブラウザで開き、パラメータを入力します。ボタンをクリックするとチャートが取得できます。

12.1.3.4 WebLogic 6.0

NT プラットフォームにおいて EspressChart を WebLogic 6.0 で実行するには、EspressChart を以下のディレクトリにインストールします。
bea\wlserver6.0\config\examples\applications\ examplesWebApp

以下を入力し、EspressChart にアクセスします。
http://<web address or machine name>:7001/examplesWebApp/EspressChart/index.html
始めに必ず EspressManager を起動し、チャートデザイナーへアクセスしてください。

サーブレットのサンプルを実行するには、まず wlserver6.0\config\ディレクトリに入ります。以下の指示に従い、“setExamplesEnv.cmd”と“startExamplesServer.cmd”を変更します。

- “@rem Set user-defined variables“セクションに“set ES_CHART=YOUR_WEBROOT\espresschart”を追加します。WebLogic 6.0 を c:\bea 下にインストールした場合、YOUR_WEBROOT は c:\bea\wlserver6.0\config\examples\ applications と同じになります。変数 ES_CHART は、マシン上の EspressChart Home ディレクトリへのパスを含みます。ご使用のマシンに合わせて値を変更してください。また、変数 WL_HOME および JAVA_HOME も適切なパスに変更してください。
- 同じファイルの“set CLASSPATH”フィールドに c:\bea;.; %ES_CHART%\lib\EspressAPI.jar; %ES_CHART%\lib\ ExportLib.jar; を追加します。

次に以下の手順を実行します（ファイルは espresschart\help\ examples\servlet\Weblogic ディレクトリにあります）：

- ExportChart2.html を wlserver6.0\config\examples\applications\ examplesWebApp ディレクトリに置きます。
- コード“archon:7001”のマシン名を ExportChart2.html フィアルの“yourMachine Name:7001”に変更します。
- ExportServlet2.java フィアルを wlserver6.0\samples\examples\servlets ディレクトリに置きます。
- examplesWebApp ディレクトリ下の wlserver6.0\config\examples\applications\examplesWebApp\WEB-INF\ディレクトリにある“web.xml”ファイルに、以下のコードを入力します。<servlet>コードはファイルの <servlet>セクションに、<servlet-mapping>コードはファイルの<servlet-mapping>にそれぞれ入力します。

```
<servlet>
   <servlet-name>ExportServlet2</servlet-name>
   <servlet-class>ExportServlet2</servlet-class>
      <init-param>
         <param-name>dataFileName</param-name>
         <param-value>help/examples/data/sample.dat</param-
         value>
      </init-param>
      <init-param>
```

```
            <param-name>category</param-name>
            <param-value>0</param-value>
        </init-param>
        <init-param>
            <param-name>series</param-name>
            <param-value>-1</param-value>
        </init-param>
        <init-param>
            <param-name>sumby</param-name>
            <param-value>-1</param-value>
        </init-param>
        <init-param>
            <param-name>value</param-name>
            <param-value>3</param-value>
        </init-param>
        <init-param>
            <param-name>chartType</param-name>
            <param-value>Column</param-value>
        </init-param>
        <init-param>
            <param-name>fileName</param-name>
            <param-value>c:\temp</param-value>
        </init-param>
    </servlet>
<servlet-mapping>
    <servlet-name>ExportServlet2</servlet-name>
    <url-pattern>/ExportServlet2/*</url-pattern>
</servlet-mapping>
```

- コマンドプロンプトウィンドウで wlserver6.0\config\examples ディレクトリに移動し、“setExamplesEnv”を実行します。
- 次に wlserver6.0\samples\examples\servlets ディレクトリへ移動し、以下のコマンドラインを使用して ExportServlet2.java をコンパイルします。

```
javac -d %EX_WEBAPP_CLASSES% ExportServlet2.java
```

- 同じコマンドプロンプトウィンドウでサンプルディレクトリの wlserver6.0\config\に戻り、“startExamples Server”を入力して Enter キーを押します。これによって WebLogic サーバが起動します。
- ウェブブラウザを開いて http://yourMachineName:7001/examplesWebApp/ExportChart2.htmlへ移動し、サーブレットのサンプルを表示します。

注意：問題が起こった場合は、入力ミスがないか確認してください。

12.1.3.5 WebSphere 3.5

- WebSphere Administrator's Console を開き、ノード（通常はマシン名） -> Default Servlet Engine -> Default App 下の Advanced タブをクリックします。Classpath に個別のラインで EspressAPI.jar と ExportLib.jar を追加します。終了したら Apply をクリックします。
- コンパイルしたクラスファイル(ExportServlet2.class)を<WebSphere インストールディレクトリ>/AppServer/servlets ディレクトリに移動します。

- FORM ACTION タグを以下のように変更します：

```
<FORM ACTION=http://<machine name>/servlet/ExportServlet2
method=POST>
```

- ノード下の Default Server を起動または再起動します。

12.1.4 サーブレットの実行

上記のようにサーブレットを設定したら、ブラウザで ExportChart2.html（EspressChart/help/examples/servlet/DataFile ディレクトリ）を開きます。パラメータを選択し、Get Chart Image をクリックします。

場合によっては、最新の結果を表示するためにメモリキャッシュおよびディスクキャッシュを毎回クリアする必要があることに注意してください。

12.2 JavaServer Pages (JSP)

12.2.1 イントロダクション

JavaServer Pages (JSP)を使用すると、ダイナミックコンテンツを HTML から分離することができます。JSP を使用する場合、HTML は通常の方法で記述されますが、ダイナミックコンテンツは HTML 内の特別なタグの中に置かれます。このタグは通常 <%" and end with ">%で始まります。

JSP は通常の HTML ファイルと同様に配置でき、ファイルの外見も HTML とほぼ同じです。しかし通常の HTML が実行されると画面表示が行われるのに対し、JSP が実行された場合はページがサーブレットにコンバートされます。

基本的に JSP は Servlet テクノロジーの拡張ですが、ユーザインターフェースをコンテンツの生成から分離できるという大きな利点があります。そのため、デザイナーはダイナミックコンテンツに手を加えることなくページ全体のレイアウトを変更することができます。

EspressChart には JSP のサンプルがいくつか含まれています。以下に JSWDK で JSP を実行するための例を紹介します。ここで使用するサンプルは datafile サーブレットを修正したもので、EspressChart/help/examples/ jsp/Chart に置かれています。以下の例は他の JSP サンプル実行のガイドとして利用できるほか、ユーザ独自の JSP コードの実行のためのガイドとしても利用できます。

12.2.2 実行環境

12.2.2.1 Apache Tomcat

- 以下の例のように、使用するマシンのクラスパスに EspressAPI.jar、ExportLib.jar、servlet.jar in the classpath を置きます。

```
set
classpath=c:\netscape\suitespot\docs\espresschart\lib\ Espre
```

```
ssAPI.jar;
c:\netscape\suitespot\docs\espresschart\lib\ExportLib.jar;
c:\tomcat\common\lib\servlet.jar;.
```

- webapps\examples\jsp\の下に *chart* サブディレクトリを作成します。Tomcat が c:\にインストールされている場合、次のようなディレクトリ構成になります。 c:\tomcat\ webapps\examples\jsp\chart
- ExportChart.jsp、ExportChartResponse.jsp、error.jsp を examples/jsp/chart ディレクトリにコピーします。
- tomcat\webapps ディレクトリ下にもう一つの *chart ディレクトリを作成します*。ディレクトリ構成は c:\tomcat\webapps\chart となります。次に getFileLocation()の URL "http://yourMachine Name/EspressChart"を正しい URL に変更します。例えばマシン名が Mach1 で、EspressChart をウェブサーバディレクトリにインストールした場合、getFileLocation() の URL は "http://Mach1/EspressChart"になります。
- CreateChart.java をコンパイルします。これによって、同じディレクトリに CreateChart.class が置かれます。
- Tomcat サーバを起動し、EspressManager を起動します。
- ウェブブラウザで http://yourMachineName:8080/examples/jsp/chart/ExportChart.jsp（yourMachineName はご使用のマシン名に置き換えてください）に移動します。
- 必要なオプションを入力し、チャートを取得します。

12.2.2.2 WebSphere

- EspressAPI と ExportLib.jar をノード（通常はマシン名）下の Dependent Classpath に追加します。
- ツールバーの Wizards ボタン（最後のボタン）をクリックし、Create a Web Application を選択します。
    - ウェブアプリケーション名に Quadbase と入力します。Enable File Servlet オプションの**チェックを外し**、Server Servlet のクラス名を**チェックして** Next をクリックします。
    - ノードを Default Servlet Engine として選択し（Default Servlet Engine が表示されるまでツリーを開いてください）、Next をクリックします。
    - Web Application Web Path を/Quadbase に変更し、Next をクリックします。
    - Classpath に個別のラインで EspressAPI.jar と ExportLib.jar を追加し、Finish を選択します。
- Default Server を再起動します。
- ご使用のウェブサーバのドキュメントルートの下に Quadbase というディレクトリを作成し、ここに.jsp ファイルと java ファイルを移動します。
- CreateChart.java を以下のように変更します。

- パッケージを削除またはコメントアウトします。
- エクスポートコマンドを以下に置き換えます。

```
chart.export(QbChart.JPEG, "<Absolute path to location of
web server document root>/EspressChart/temp", 500, 400);
```

- .jsp ファイルの chart.CreateChart を CreateChart に置き換えます。
- ご使用のウェブサーバのドキュメントルートの下に EspressChart というディレクトリを作成します。
- ブラウザで http://<machine name>/Quadbase/Export Chart.jsp を開きます。パラメータを入力してボタンをクリックし、チャートを取得します。

12.3 チャートのファイル保存とブラウザへのチャート送信

ブラウザにチャートを返すには、サーブレットまたは JSP を使用する方法があります。チャートはインタラクティブな形式でアプレットに表示することも、あるいは jpeg や png などの静止画像として表示することもできます。回線のトラフィック量とバンド幅によって、アプレットと静止画のどちらが適切かを判断してください。

チャートをブラウザに返すには、チャートをファイル保存する方法と、ブラウザに直接送信する方法とがあります。前者の方法ではチャートはサーバサイドに保存され、クライアントのブラウザにファイルの場所が送信されることによってチャートが表示されます。後者の方法では、チャートそのものがクライアントのブラウザにストリングまたは出力ストリームとして送信されます。

12.3.1 チャートのファイル保存

表示したチャートを再利用する場合やコピーが必要な場合、サーブレットおよび JSP はチャートをファイル保存することができます。チャートはアプレットで表示されているか静止画像として表示されているかに関わらず、サーバサイドのファイルに保存されます。しかし、このタイプのサーブレットや JSP の使用には注意が必要です。複数のクライアントがページを閲覧している場合、あるクライアントが別のクライアントのチャートを見ることも可能になります。またチャートが静止画とのき、ブラウザは前のチャートをキャッシュすることがありますので、新しいチャートを表示するにはリフレッシュまたはリロードを行う必要が発生します。さらに、ファイルはサーバサイドに保存されるため、クライアントが閲覧する前にチャートを上書きしてしまわないよう注意が必要です。またサーバスペースから定期的に不要なチャートを取り除くことも必要です。

いずれの場合でも、チャートは任意のフォーマットでエクスポートできます。

```
QbChart chart = new QbChart (...........);
....
....
....
```

```
int format = QbChart.CHT; // Here set to an interactive chart. For
// a static chart change to QbChart.JPEG
chart.export(format, "chart", 500, 500);
```

チャートの場所の指定を含む HTML ファイルがブラウザに返され、ブラウザにチャートが表示されます。

```
<applet code="quadbase.chartviewer.Viewer.class" width=650
height=650
archive="http://<machineName>/EspressChart/lib/MiniViewer.jar">
<PARAM name="filename" value="chart.cht">
</applet>
```

この例ではチャートビューワを使用しています。上記の場合、EspressManager が必ず実行されていなければならないことに注意してください。

チャートが静止画像の場合、単純な<img src>タグでチャートの場所を指定するだけでチャートを表示できます。

この方法によって、チャートを返すページをサーブレット／JSP から分離し、サーブレット／JSP と関わりなく事前にページをデザインしておくことができます。データベースまたはハッシュテーブルを使用することによって、チャート生成の記録や再表示、および定期的なチャートの再生成を行うこともできます。

詳細は *help/examples/servlets/DatabaseJPEG* ディレクトリおよび *help/examples/jsp/Chart* ディレクトリのサンプルファイルを参照してください。

12.3.2 チャートを直接ブラウザへ送信

サーブレットおよび JSP は、チャートを直接ブラウザへ送信することができます。この場合、チャートはサーバサイドにファイルとして保存される必要がないため、サーバのディスクスペースを占有せず、ファイルのメンテナンスも必要ありません。しかしこの場合、ブラウザでチャートを表示しない限りチャートのコピーは作成されませんので、再びチャートが必要となった時はチャートを再生成する必要があります。

チャートのブラウザへの送信はファイル保存とは異なり、チャートが出力ストリーム（静止画像の場合）またはストリング（インタラクティブチャートの場合）として送信されます。

静止画像の場合、サーブレットサイドで出力ストリームが作成され、チャートが出力ストリームにエクスポートされます。

```
public class OutputStreamServlet extends HttpServlet {
    public void doGet(HttpServletRequest request, HttpServlet
    Response response) throws ServletException, IOException {
        // first, set the "content type" header of the response
        response.setContentType("image/html");
        // where response is the response to the servlet
        OutputStream toClient = res.getOutputStream();
        QbChart chart = new QbChart (.....);
```

```
            ....
            ....
            ....
            chart.export(QbChart.JPEG, toClient, 500, 500);
        }
    }
```

HTML 上では、<img src>タグにサーブレットを含むことによってチャートの表示ができます。

```
<HTML><BODY>
<img src="http://machine_name:8080/servlet/OutputStreamServlet>
</BODY></HTML>
```

マップファイルを出力ストリームとして出力し、生成された HTML ファイルに埋め込むこともできます。マップファイルを出力ストリームとして生成するには、以下の QbChart クラスの方法を使用します。

```
export(int format, OutpuStream image, OutputStream mapFile, String
fileName, int w, int h, boolean generateMapFile, int option);
```

インタラクティブチャートはストリングとしてエクスポートされます。

```
QbChart chart = new QbChart (.....);
String buffer = chart.exportChartToString();
....
....
....
```

HTML 上では、チャートビューワを使用してインタラクティブチャートを表示できます。以下のように、HTML はサーブレットコード内に埋め込まれます。

```
toClient.println("<applet
code=\"quadbase.chartviewer.Viewer\ "width=600 height=600" +
"archive=\"http://<machineName>/ EspressEnterprise/EspressChart/lib
/EspressViewer.jar\">");
toClient.println("<PARAM name=\"ChartData\" value=\"" +
chart.exportChartToString() + "\">");
toClient.println("</applet>");
```

詳細は help/examples/servlets/StreamingJPEG および help/examples/servlets/StreamingChart ディレクトリのサンプルファイルを参照してください。

JSP 上では、チャートを作成して String オブジェクトとして返すための Java Bean オブジェクト(myStreamChart)を作成する必要があります。JSP ページは以下のように変更してください。

```
<jsp:useBean id="myStreamChart" scope="page" class="streamingChart.
CreateChart" />
<jsp:setProperty name="myStreamChart" property="*"/>
<applet code="quadbase.chartviewer.Viewer" width=600 height=600
archive="http://<machineName>/EspressChart/lib/EspressViewer.jar">
```

```
<PARAM name="ChartData" value="<%= myStreamChart.export()%>">
</applet>
```

myStreamChart java bean に以下のコードを入れます。

```
public string export() throws Exception {

QbChart chart = new QbChart(....);

return chart.exportChartToString();

}
```

詳細は help/examples/jsp/streamingChart にあるファイルを参照してください。

静止画像をブラウザに直接送信する場合は、JSP とサーブレットを組み合わせて使用します。

13 ディプロイメント

13.1 概要

EspressChart は、複数のコンポーネント（チャートデザイナー、チャートビューワ、EspressManager、チャート API など）で構成されています。チャートデザイナーを使用して、GUI（グラフィカル･ユーザ･インタフェース）環境でチャートを作成します。 チャートビューワは、（.cht または.tpl フォーマットで保存された）チャートを表示するために使用するアプレットです。プログラムを使用してチャートを作成するには、チャート API を使用します。スケジューラは、チャートのエクスポートをスケジューリングするのに使用します。最後に、EspressManager はユーザアドミニストレータとして作動し、データとデータバッファリングを処理します。

コードを実行するには、クラスパスまたは HTML ページに（ほかの jar とともに）qblicense.jar がアーカイブとして含まれていなければならないことに注意してください。

チャートデザイナーは EspressManager とともに使用される必要がありますが、チャートビューワとチャート API を EspressManager に接続しないで稼動させるためのオプションが提供されています。

EspressManager を接続するには、EspressManager に接続する方法に関する情報を指定する必要があります。EspressManager がアプリケーションとして実行されている場合、API メソッドを使用して、EspressManager が位置づけられている IP アドレス/マシン名と EspressManager がリスンしているポート番号を指定することができます。

以下の 2 つの API メソッドは、接続情報を設定します。

```
static void setServerAddress(java.lang.String address);
static void setServerPortNumber(int port);
```

たとえば、以下のコード行は、*someMachine* で実行され、*somePortNumber* をリスンしている EspressManager に接続します。

```
Qbchart.setServerAddress("someMachine");
Qbchart.setServerPortNumber(somePortNumber);
```

EspressManager への接続情報を指定しないと、コードは、ローカルマシン上で実行され、デフォルトのポート番号(22071)をリスンしている EspressManager に接続しようとすることに注意してください。

EspressManager がサーブレットとして実行されている場合、以下のメソッドを使用することができます。

```
public static void useServlet(boolean b);
public static void setServletRunner(String comm_url);
public static void setServletContext(String context);
```

たとえば、以下のコード行は、*http://someMachine:somePortNumber/EspressChart/servlet* で実行されている EspressManager に接続します。

これらのメソッドは、QbChart および QbChartDesigner 内にあることに注意してください。

チャートビューワ（アプレット内）の場合、以下のパラメータで受け渡すことによって EspressManager への接続情報を設定することができます。

server_address (server address), *server_port_number* (port number)

上記のパラメータは、EspressManager がアプリケーションとして実行されている場合に設定されます。EspressManager がサーブレットとして実行されている場合、以下のパラメータを使用します。

comm_protocol (servlet), *comm_url* (machine name/IP address and port number), *servlet_context* (servlet context)

以下のセクションではよくある様々なディプロイメントとそれぞれの結果について説明します。

13.2EspressManager でのディプロイメント

このセクションでは EspressManager と一緒に動くチャートコンポーネントのディプロイメントを説明していきます。相互作用に関しての詳細は“EspressChart API とは”の章の“EspressManager を使って“のセクションを参照してください。

相対 URL 参照を使用すると、データソースやチャートファイル（例えば：*help\examples\data\sample.dat*）は、EspressManager が実行されているディレクトリに対して相対的に参照されることに注意してください。

13.2.1 チャートデザイナー

先にも述べたとおり、チャートデザイナーは EspressManager と一緒に使用される必要があります。チャートデザイナーは API を使って呼び出すこともできます。それには CLASSPATH に ChartDesigner.jar を追加し、.class ファイルのワーキングディレクトリ下に images（EspressChart/images）をコピーし、backgroundImages (EspressChart/ backgroundImages)ディレクトリを配置してください。

API からチャートデザイナーを呼び出す際にディレクトリをワーキングディレクトリに対して相対的にコピーする代わりに、images/および backgroundImages/ディレクトリの位置を設定するオプションを使用することもできます。単に、パラメータ **imagesPath** を指定した QbchartDesigner コンストラクタを使用するだけです。QbchartDesigner コンストラクタまたは imagesPath パラメータの詳細については、EspressChart Java API ドキュメンテーションを参照してください。

アプリケーションとして実行している EspressManager に接続する場合、QbchartDesigner で以下の 2 つの API メソッドを使用して、接続情報を設定します。

```
static void setServerAddress(java.lang.String address);
static void setServerPortNumber(int port);
```

サーブレットとして実行している EspressManager に接続している場合、QbchartDesigner で以下の 2 つの API メソッドを使用して、接続情報を設定します。

```
static void useServlet(boolean b);
static void setServletRunner(String comm_url);
static void setServletContext(String context);
```

QbChartDesigner コンストラクタを呼び出す前に、EspressManager への接続情報が指定されていなければならないことに注意してください。

13.2.2 チャートビューワ

チャートビューワのデプロイメントは、チャート Viewer のデプロイメントと同様です。チャートビューワをアップレットまたはアプリケーション環境で使用すると、（デザイナーまたは API によって作成される）*.cht* または *.tpl* ファイルに保存されたチャートを表示することができます。

チャートビューワをアプレット環境にデプロイするには, EspressViewer.jar がアーカイブとして HTML ファイルに含まれている必要があります。

チャートビューワを使用するには、正確なアプレットコードで HTML ページを構成します（詳細については、 チャートビューワ の章を参照してください）。チャートの位置とデータは、EspressManager が起動されたディレクトリに対して相対的に参照されることに注意してください。

アプリケーションとして実行されている EspressManager に接続するには、以下のパラメータをチャート Viewer アプレットに渡します。

```
server_address (server address), server_port_number (port number)
```

サーブレットとして実行している EspressManager に接続しているには、チャート Viewer アプレットに次のパラメータを渡します。

```
comm_protocol (servlet), comm_url (machine name/IP address and port
number), servlet_context (servlet context)
```

13.2.3 チャート API

チャート API はアプレット環境またはアプリケーションで使用されます。チャート API はサーブレットまたは jsp 環境でサーバーサイドチャートを生成するのに使用されます。

チャート API をディプロイするには、EspressAPI.jar と ExportLib.jar が、CLASSPATH 内（アプリケーションとサーブレット/jsp 環境の場合）にあるか、または、HTML ファイル（アプレットの場合）に含まれていなければなりません。

アプリケーションとして実行されている EspressManager に接続するには、QbChart 内で以下のメソッドを使用して、接続情報を設定します。

```
static void setServerAddress(java.lang.String address);
static void setServerPortNumber(int port);
```

サーブレットとして実行されている EspressManager に接続しているには、QbChart 内で以下のメソッドを使用して、接続情報を設定します。

```
static void useServlet(boolean b);
static void setServletRunner(String comm_url);
static void setServletContext(String context);
```

QbChart コンストラクタを呼び出す前に、EspressManager への接続情報が指定されてなければならないことに注意してください。

13.3EspressManager に接続しない場合のデプロイメント

このセクションでは EspressManager とは独立して動作するチャートコンポーネントのデプロイメントについて説明していきます。EspressManager とのインタラクショ

ンの詳細については、「EspressChart API の概要」 の章の「EspressManager とのインタラクション」のセクションを参照してください。

EspressManager とは独立してチャートを生成すると、EspressManager とチャートまたはチャートコンポートネント間のインタラクションが省かれるため、パフォーマンスが多少向上します。

注意しなければならない重要な点は、データソースやチャートソース（例 *help\examples\data\sample.dat* ）は、HTML ファイル（アプレットの場合）またはクラスファイル（アプリケーションの場合）が実行されるワーキングディレクトリに対して相対的に参照されることです。サーブレットまたは jsp 環境では、通常、ワーキングディレクトリはクラスがあるディレクトリとは異なります。ワーキングディレクトリを見つけるには、サーブレットまたは jsp で次のコードを使います。

```
System.out.println("The working directory is " +
System.getProperty("user.dir") + ". ");
```

上記のコードを挿入して、サーブレットまたは jsp を実行することによって、ワーキングディレクトリが確認され、コードで指示されたとおりに、データファイルとチャートテンプレートをワーキングディレクトリに対して相対的な位置に移動することができます。

13.3.1 チャートビューワ

チャートビューワをディプロイするには、EspressViewer.jar がアーカイブとして HTML ファイルに含まれていなければなりません。また、使用する機能に応じて、必要があれば別の jar を追加します。

アプレットコードに以下のパラメータを追加すると、チャートビューワを EspressManager とは独立して使用することができます。

```
<param name="EspressManagerUsed" value="false">
```

チャートの位置またはデータは、HTML ファイルがあるディレクトリに対して相対的に参照されることに注意してください。例えば、cht ファイルで.指定されたデータソースが help\examples\data\sample.dat 内にあり、HTML ファイルが D:\EspressChart\TestApplet にある場合、チャートを正しく表示するには、help\examples\data\sample.dat が D:\EspressChart\TestApplet 内に存在しなければなりません(すなわち、D:\EspressChart\TestApplet\help\examples\data\sample.dat が存在しなければなりません)。

13.3.2 チャート API

チャート API をディプロイするには、ChartAPI.jar と ExportLib.jar が CLASSPATH（アプリケーションとサーブレット/jsp 環境）、あるいは HTML ファイル（アプレットの場合）に含まれていなければなりません。

チャート API は以下のコードを加えるこにより EspressManager に接続することなく使用できます。

```
QbChart.setEspressManagerUsed(false);
```

このメソッドは、いかなる QbChart インストラクションよりも**優先してコールされなくてはなりません**。

チャートビューワと同様、チャートロケーションやデータの相対参照はクラスファイルが実行されるワーキングディレクトリに対して相対位置となります。例えば、指定されたデータソースが help\examples\data\sample.dat で、クラスファイルが D:\EspressChart\TestApplication という場所にある場合、チャートが表示されるようにするには、help\examples\data\sample.dat が D:\EspressChart\TestApplication 内に存在しなければなりません。（つまり、D:\EspressChart\TestApplicationhelp\examples\data\sample.dat が存在している必要があります。）

13.4Windows 以外の環境でのデプロイメント

EspressChart はチャート作成のためのピュア Java ツール です。そのものとしては、カラー、フォント、その他の AWT 情報を生成するグラフィカルなライブラリはもっていません。そのため、Java はその情報を与えるシステム（そのシステムでチャートが作成されます。）のライブラリに依存します。従って、AWT 情報やグラフィクスカード（スタティックフォーマットにエキスポートするため）を供給できることが必要とされます。

Windows 環境では、既にそれがあるので、そのような環境でも特に何かを設定しなければならないということはありません。GUI インタフェースが既に実行され、グラフィックフカードも既に存在しています。

> Windows 以外でも Unix や Linux のような環境ではまた違ってきます。X を持つこと、またはそのようなシステムでいくつかの X が実行され、ディスプレイを X が実行されているマシーンポイントする必要があります。（コーンシェルでコマンド export DISPLAY=192.168.0.16:0.0 を実行するなど。）最も効果的なパフォーマンスを得るには、Quadbase はマシーン上で X を実行す

ることを推奨します。（またはその DISPLAY を X が実行されている他のマシーンへポイントように設定します。）しかし、もしそれができない場合はさらに２つの方法があります。

13.4.1 Xvfb (X Virtual Frame Buffer)

Xvfb はディスプレイハードウェアのない、物理的に入力デバイスのないマシーンで実行できる X サーバです。バーチャルメモリを使ってダムターミナルのフレームバッファを再現します。

このサーバの主な使用はサーバテストをするのが目的です。また、Xvfb は処理できるカラーとフォントの数に制限があります。

Xvfb はダウンロード（システムによって使用できるバージョンが違います。）され、実行できます。 まず、Xvfb を実行し、DISPLAY 変数をセットします。そして、アプリケーション（またはサーブレットエンジン）に渡されます

13.4.2 JVM 1.4 in headless mode

EspressChart を使用してアプリケーションを稼動させるために JVM1.4 ランタイムパラメータを使用することが出来ます。この JVM1.4 ランタイム パラメータは java.awt.headless でグラフィック情報の X 接続をすることなく Java での true 結果に対して設定することが出来ます。

例えば、jpg のチャートを生成する為の chartExport と呼ぶアプリケーションを稼動させるには X への接続が必然となります。しかし下記のコマンドを使用することで、X への接続が無くっても同じアプリケーションを稼動させることが出来ます。

```
java -Djava.awt.headless=true exportChart
```

13.5Platform Specific Issues

13.5.1 AS/400

13.5.1.1 Installation notes for AS/400

- Run installer (EspressChart installation program) and unpack to E:\home\ quadbase (where E:\ is the drive mapped to the AS/400 on the RAWT box)
- In AS/400 http server, add PASS /echart/* to /home/quadbase/EspressChart/* (creates a virtual directory to EspressChart so that EspressChart can be accessed by http://<machine_name>/echart)
- Restart http server
- CRTJVAPGM on server jar file (EspressManager.jar) (This converts the jar file to native code)

- Configure on RAWT, start on PC (For more info. please go to http://www.as400.ibm.com/java . Click on Library and then on AS/400 Info Center: Manuals/Redbooks (which takes you to http://publib.boulder.ibm.com/html/as400/infocenter.html) to find out how to configure RAWT. Change CLASSPATH to include RAWTGUI.zip
- Start EspressManager
    - CHGCURDIR to /home/quadbase/EspressChart
    - Set up properties for RAWT
    - Make sure -monitor:OFF (Monitor option is off)

13.5.1.2 Running WebSphere Servlets under AS/400

- Create a Qtmhhttp directory under /home (if it doesn't exist)
- Add a SystemDefault.properties file which contains

    RmtAwtServer=<IP address of RAWT box>

    os400.class.path.rawt=1 (This depends on how the RAWT is configured and what version of AS/400 is being used. Please substitute accordingly)

13.5.2 Linux/Unix

13.5.2.1 *12.5.2.1. X 下で動作する場合*

Linux/Unix の環境で、EspressChart を統合しているアプリケーションを X で動作する場合、下記の手順を行なうことになります：

環境変数の DISPLAY を、X クライアント名と表示番号に設定します。

13.5.2.2 *12.5.2.2. Headless モードで動作する場合*

Linux/Unix の環境で、EspressChart を統合しているアプリケーションをヘッドレス(headless)モードで動作する場合、下記の手順を行なうことになります：

- Java を実行する前に、Java のシステムプロパティを設定します。

    java.awt.headless=true

- QbChart コンストラクターを呼び出す前に、下記のコードを含む必要です。

    QbChart.setForExportOnly(true);

Appendix A: Parameter Server

A.1 Writing Parameter Server

チャートビューワ supports push technology by means of interacting with a parameter server. A programmer can write a parameter server using the class quadbase.chartviewer.ParamServer to supply updated data continuously for Chart Viewer to plot the chart.

On the client side, the HTML syntax for the user is either

```
<applet code = "quadbase.chartviewer.Viewer.class" width=640
height=480>
<PARAM name="filename" value="example.cht">
<PARAM name="ParameterServer" value="machine:portno">
</applet>
```

or

```
<applet code = "quadbase.chartviewer.Viewer.class" width=640
height=480>
<PARAM name="filename" value="example.tpl">
<PARAM name="ParameterServer" value=":portno">
</applet>
```

Once the browser has loaded Chart Viewer class and the chart data from the web server, Chart Viewer attempts to connect to the parameter server as specified. In the first case Chart Viewer will try to connect to the specific machine and port number identified in the parameter. In the second case a default machine is used: the web server machine. For security reasons untrusted applets are not allowed to open a connection to other machines. The parameter server can interact with Chart Viewer in order to update and manipulate the records held by Chart Viewer.

On the server side, a server program should be written to listen to the port number. This server can be written using the class `quadbase.chartviewer.Paramserver`.

Upon opening a connection, the server will supply data to Chart Viewer.

The constructor for the server is:

```
public ParamServer(DataInputStream in, DataOutputStream out)
```

Where `in` and `out` are the socket input and output used to communicate with the server.

`ParamServer` has three basic methods to manipulate the records in Chart Viewer:

```
int addRecord(Object record[]);
int deleteRecord(int recordNo);
int updateRecord(Object record[], int recordNo);
```

After a sequence of record updates a final call

```
void repaint();
```

will send a request to Chart Viewer to repaint the chart using the new data.

The following Java program provides an example of using the parameter server class to generate new charts. チャート API for `ParamServer` is provided in the online API documentation. In this example, one field in the twelfth record is simply updated with an integer chosen at random. In a real application, a programmer would have to write the code which enabled the parameter server to interact with the data source.

```
import java.io.*;
import java.net.*;
import java.util.*;
import quadbase.chartviewer.ParamServer;

// Sample Program to update the data in Chart Viewer using
// Parameter Server
public class pserver extends Thread
{
       protected int port = 1997;
       // port no. for Chart Viewer to connect
       protected ServerSocket listen_socket;
       public static void fail(Exception e, String msg)
       {
          System.err.println(msg + ":" + e);
          System.exit(1);
       }
       public pserver()
       {
          try
          {
             listen_socket = new ServerSocket(port);
          }
          catch (IOException e)
          {
             fail(e, "Exception creating server socket");
          }
          this.start();
       }
       public void run()
       {
              try
              {
                 while(true)
                 {
                    // create a thread for each connection to
                    // handle the request

                    Socket client_socket = listen_socket.accept();
                    Connection c = new Connection(client_socket);
                 }
```

```
            }
            catch (IOException e)
            {
                fail(e, "Exception while listening for connections");
            }
    }
    public static void main(String[] args)
    {
                new pserver();
    }
}
class Connection extends Thread
{
    protected Socket client;
    protected DataInputStream in;
    protected DataOutputStream out;

    public Connection(Socket client_socket)
    {
           client = client_socket;
           try
           { // get the input and output stream
              in = new DataInputStream(client.getInputStream());
              out = new DataOutputStream(client.getOutputStream());
           }
           catch (IOException e)
           {
              try
              {
                     client.close();
              }
              catch (IOException e2)
              {}
              return;
           }
           this.start();
}
public void run()
{
    ParamServer paramserver;
    Random random = new Random(System.currentTimeMillis());
    try
    {
       paramserver = new ParamServer(in, out);
       System.out.println("Total no of record = " +
       paramserver.getRecordNo());
       // get record 12 from Chart Viewer
       Object rec[] = paramserver.getRecord(12);
       for (int i=0; i < 100; i++)
       {
          // update the third field of record 12
          rec[3] = new Integer(random.nextInt() % 30);
          paramserver.updateRecord(rec, 12);
          // tell Chart Viewer to repaint
```

```
                //after every tenth record is updated
                if (i % 10 == 0)
                    paramserver.repaint();
            }
        }
        catch (IOException e) {}
        finally
        {
            try
            {
                    client.close();
            }
            catch (IOException e2) {}
            }
        }
    }
```

Chart Viewer would have read an HTML file of the form:

```
<html>
<title>Sample Chart</title>
<applet code = "quadbase.chartviewer.Viewer.class" width=600
height=450>
<PARAM name="filename" value="col2d.cht">
<PARAM name="ParameterServer" value=":1997">
</applet>
</html>
```

Here the Java class that Chart Viewer reads is the name of a chart file col2d.cht and the next line specifies the machine and port to which Chart Viewer should open a connection. EspressManager will then provide information to Chart Viewer at regular intervals such as updated records so that Chart Viewer can refresh the chart it displays.

13.5.3 A.1.1 Variables

UPDATE_RECORD
public final static int UPDATE_RECORD

INSERT_RECORD
public final static int INSERT_RECORD

DELETE_RECORD
public final static int DELETE_RECORD

GET_RECORD
public final static int GET_RECORD

GET_RECORDNO
public final static int GET_RECORDNO

GET_RECORD_DATATYPE
public final static int GET_RECORD_DATATYPE

REPAINT

public final static int REPAINT

OK
public final static int OK

ERROR
public final static int ERROR

13.5.4 A.1.2 Constructor

```
public ParamServer(DataInputStream in, DataOutputStream out) throws
IOException
```

Constructor for Parameter Server

13.5.5 A.1.3 Methods

updateRecord

```
public int updateRecord(String rec[], int recordNo) throws IOException
```

Update a record
Parameters:
rec - the record passed in as a string array, see class quadbase.chartAPI. DbData for the format of the string array.
recordNo - the record number to be updated
Returns:
OK if success, returns ERROR if no such record number or record type mismatch.

updateRecord

```
public int updateRecord(Object rec[], int recordNo) throws IOException
```

Update a record
Parameters:
rec - the record passed in as an object array
recordNo - the record number to be updated
Returns:
OK if success, return ERROR if no such record number or record type mismatch.

addRecord

```
public int addRecord(String rec[]) throws IOException
```

Add a record
Parameters:
rec - the record passed in as string array
Returns:
OK if success, returns ERROR if record type mismatch.

addRecord

```
public int addRecord(Object rec[]) throws IOException
```

Add a record
Parameters:
rec - the record passed in as an object array
Returns:
OK if success, returns ERROR if record type mismatch.

deleteRecord

```
public int deleteRecord(int recordNo)throws IOException
```

Delete a record

Parameters:

recordNo - the record number to be deleted

Returns:

OK if success, returns ERROR if record type mismatch.

repaint

```
public void repaint() throws IOException
```

Inform Chart Viewer to repaint the chart

checkRecord

```
public boolean checkRecord(Object rec[])
```

Auxiliary function to check whether the record type given matches the record type used in Chart Viewer

Parameters:

rec - the input record to be checked

Returns:

true if record match, otherwise return false

getRecordNo

```
public int getRecordNo() throws IOException
```

Get the number of records used

Returns:

the number of record

getDataType

```
public int[] getDataType() throws IOException
```

Get the data type of the record used

Returns:

array of data types as defined in jdbc/Types.class, the size of the array is equal to the number of field in record

getRecordSize

```
public int getRecordSize()
```

Get the number of fields in record

Returns:

number of fields in record

getRecord

```
public Object[] getRecord(int recordNo) throws IOException
```

Get the record

Parameters:

recordNo - the record number to be retrieved

Returns:

an object array for the record

convertRecord

```
public Object[] convertRecord(String rec[])
```

Convert a record in string array format to Object array format

Returns:

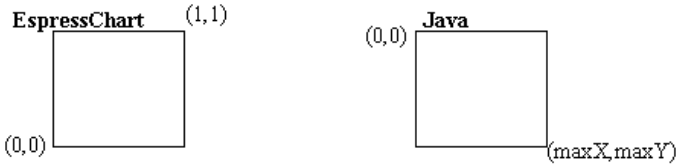
object array for the record, null if data type mismatch

Appendix B: EspressChart XML Chart File Parameters

To view the list of parameters that can be used within the XML chart file, please go to the online document which is located at EspressChart/help/manual/index.html and click on Appendix B. Also note that any one of those parameter values can be modified to change the look and feel of the chart.

Appendix C: Customizing Chart Layout

C.1 Introduction



Component Layout differences between EspressChart and Java

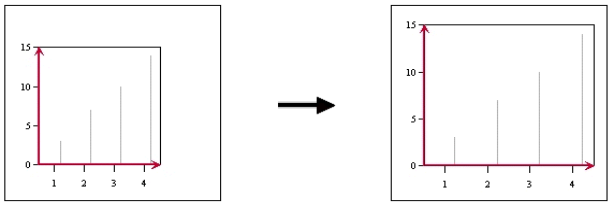
EspressChart describes its component layout in relative coordinates. On the other hand, Java uses layout format in terms of pixels. Because of backward-compatibility issues, the EspressChart layout format will not be changed. The issues pertaining to relative & absolute position is less relevant in Chart Designer since there is a graphical user's interface. However, it is helpful to know how the layout format is in Chart API: the origin (0, 0) is at the lower left-hand corner of the canvas, and the maximum (1, 1) is at the upper right-hand corner.

In the Java layout format, the origin is at the upper left-hand corner, and the maximum pixel (maximumX, maximumY) is at the lower right-hand corner.

In the IAxis Class, the methods setMaxScale, setMinScale, and setScaleStep refer to the range and interval of the axis. Therefore, these methods would not affect the apparent length of the axis, rather, they specify the range of the axis.

The following sections discuss some examples of customization techniques relating to layout of chart components, re-sizing and data point labeling.

C.2 Changing the Chart Plot Area



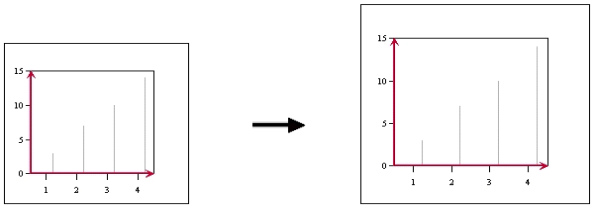
Enlarging the Chart Plot Area

The default relative size of the Chart Plot area is 0.6 in the x dimension and 0.6 in the y dimension. If you want to adjust these ratios using the EspressAPI, you can use the setRelativeHeight () and setRelativeWidth () methods.

```
// chart plot's default relative size is 0.6, 0.6
IPlot chartPlot = chart.gethChartPlot();
chartPlot.setRelativeHeight( (float) 0.75);
chartPlot.setRelativeWidth( (float) 0.8);
```

To adjust the chart plot in the Chart Designer, simply click and hold down the right mouse button when the mouse cursor is on the chart. Drag the mouse to the right to increase the chart plot area and to the left to decrease the chartplot area. Depending on the direction that you move the mouse (either vertically, horizontally or diagonally), the chart plot changes in one or both dimensions.

C.3 Changing the Canvas Area



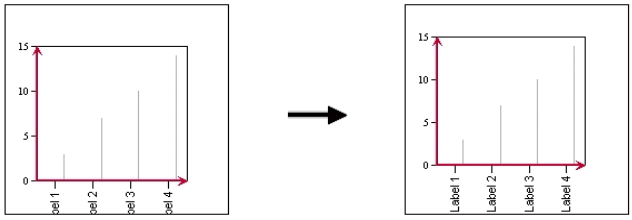
Enlarging the Canvas Area

This example shows you how to adjust the size of the canvas while keeping the Chart Plot size constant in relation to the canvas.

```
// default canvas size is 473 pixels by 427 pixels
int width = 500; // in pixels
int height = 550; // in pixels
chart.gethCanvas().setSize(new Dimension(width, height));
```

To change the canvas area in the Chart Designer, go to Format -> Canvas and change the dimensions there. Note that the canvas is always equal to or greater than the Chart Designer window size. If you want a smaller Canvas area, you must reduce the dimensions of the Chart Designer window first.

C.4 Fit Chart Elements

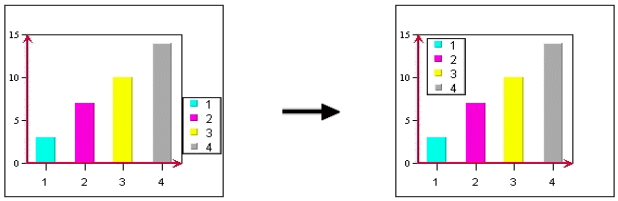


Fitting Chart elements onto Canvas

If you notice that there are chart elements (e.g., ticker labels for the Category Axis) that are "chopped off" by the canvas, you can use the setFitOnCanvas() method to do the necessary adjustments.

```
// Try to reposition chart location or reduce chart size
// until EspressChart is able to fit a whole chart on
// canvas. Default value is FALSE

chart.gethCanvas().setFitOnCanvas(true);
```

C.5 Changing Legend Box Position



Changing the position of the legend box

The position of the legend box can be adjusted if the default location is not what you want. Please note that the reference point is the center of the legend box.

```
// Legend Box can be moved with a specified new relative position
ILegend hLegend = chart.gethLegend();
float x = 0.2;
float y = 0.7;
hLegend.setPosition( new Position( x, y) );
```

Legend box size is dependent on the size of the text font & text strings. Therefore, you cannot set the legend box size directly, but you can get the size of the legend box. Here are some sample codes.

```
ILegend hLegend = chart.gethLegend()
float relWidth = hLegend.getRelativeWidth();
float relHeight = hLegend.getRelativeHeight();
```

To change the position of the legend in the Chart Designer, left-click on the legend and drag it to the desired position on the canvas.

C.6 Attaching Labels to Datapoints

Depending on your needs, it is sometimes necessary to have labels attached to data points. EspressChart provides a few features to fulfill these requirements. For example, you can attach annotation text to trend lines. When the trend line moves with the data, the annotation text will move in tandem with the trend line. Top labels can be set visible and appear at the top of every data point. However, in certain situations, you may want to have labels attached to just a few selected data points. How would you go about doing this?

You can use existing features to achieve this as follows: Draw a constant horizontal line with value of the desired data point. Attach an annotation text to the line. Hide the legend for the line. Then make the horizontal line invisible by simply choosing a dashed line style and making both the fill and empty pixel lengths to be 255.

The following code fragment demonstrates this technique. The following tables show the data referred to in the code. The objective is to attach labels to two data points at the far right of the chart (the maximum of both series) and to attach one label to a data point at the left side of the chart (the minimum as shown below).

Please note that the positions of the labels are updated automatically when one of the last data points changed from $700 to $500.

Dav	Value1	Value2
1	100	100
2	150	300
3	200	500
4	250	700

Original Data

Dav	Value1	Value2
1	100	100
2	150	300
3	200	450
4	250	500

New Data

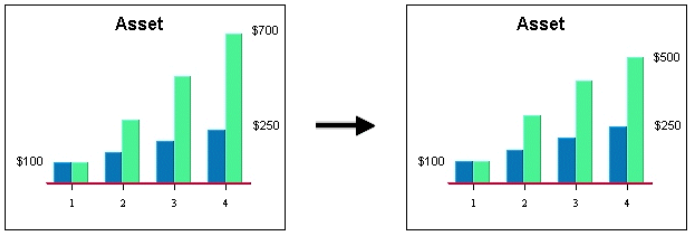


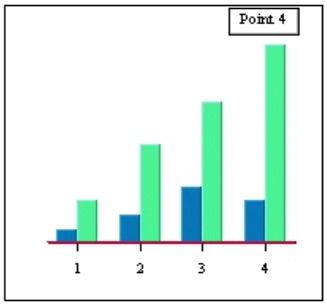
Chart with original data changing to chart with new data

```
// example finds one of the value points in the last row of a
// database query. It's not necessary to label the last value in
// particular, you can specify whichever values to label in your
// program, i.e. maximum, minimum, or average...
// get handle on the first set of data points
IRow rowInit = chart.gethInputData().getRow(0);
// get initial value. if the first series is from column 3, index is 3
int value Init = Integer.parseInt(rowInit.getObject(3).toString());

// get handle on the last set of data points
int lastRownumber = chart.gethInputData().getRowCount() - 1;
IRow lastRow = chart.gethInputData().getRow(lastRownumber);
IRow lastButOneRow = chart.gethInputData().getRow(lastRownumber-1);

// the second series last value is in the last row
int value1 = Integer.parseInt(lastRow.getObject(3).toString());
// the first series last value is in the last but one row
int value2 = Integer.parseInt(lastButOneRow.getObject(3).toString());
```

```
String labelInit = "$" + valueInit;
String label1 = "$" + value1;
String label2 = "$" + value2;

// add the line for the initial value
IDataLineSet hDataLines = chart.gethDataLines();
IHorzVertLine hvInit = hDataLines.newHorzVertLine(
IHorzVertLine.HORIZONTAL_LINE, labelInit);

hvInit.setLineValue(valueInit); // horizontal line at value0
hvInit.setLineStyle( ((255*256) + 255) *256); // set line invisible
hDataLines.add(hvInit);

// add the line for the ending value of the first series
IHorzVertLine hv1 = hDataLines.newHorzVertLine(
IHorzVertLine.HORIZONTAL_LINE, label1);

hv1.setLineValue(value1); // horizontal line at value1
hv1.setLineStyle( ((255*256) + 255) *256); // set line invisible
hDataLines.add(hv1);

// add the line for the ending value of the second series
IHorzVertLine hv2 = hDataLines.newHorzVertLine(
IHorzVertLine.HORIZONTAL_LINE, label2);
hv2.setLineValue(value2); // horizontal line at value1
hv2.setLineStyle( ((255*256) + 255) *256); // set line invisible
hDataLines.add(hv2);

// Add Annotations to the lines
IAnnotationSet hAnnotation = chart.gethAnnotations();
IAnnotation annoInit = hAnnotation.newAnnotation(labelInit, hvInit);
IAnnotation anno1 = hAnnotation.newAnnotation(label1, hv1);
IAnnotation anno2 = hAnnotation.newAnnotation(label2, hv2);

// Add annoInit to the left of the chart plot area. ($100)
hvInit.addAnnotation(annoInit);
annoInit.setRelativePosition(new Point_2D( -
    float)chart.gethChartPlot().getRelativeWidth(), 0f));

// Add annotation1 to the line, so it will appear on the chart
hv1.addAnnotation(anno1);

// Add annotation2 to the chart, so it will appear on the chart
hv2.addAnnotation(anno2);

// set legend box invisible
chart.gethLegend().setVisible(false);
```

If you want to show individual category top labels selectively, you can add a vertical trend line with the corresponding category value. Within EspressChart's graphical mechanism, category data point positions are always referenced as numbers even though they are usually not numbers. The first data point is always referenced as 0.5, and each of the following points would have 1.0 added to it. Thus, a set of points would always be referenced as 0.5, 1.5, 2.5, 3.5...etc. The only exceptions are in Scatter and Bubble Charts.

After inserting the line in the right position, we can make the line invisible by setting the line style. Annotation is then added.



Adding "Point 4" to the chart

```
// adding a label above the fourth point, "Point 4"
ITrendLine tr1 = hDataLines.newTrendLine(ITrendLine. VERTICAL_LINE,
          1, "Point 4");
tr1.setTitleVisibleInLegend(false); // don't show title on legend
tr1.setLineValue(3.5); // vertical line on the 4th data point
tr1.setLineStyle( ((255*256) + 255) *256); // set line invisible
hDataLines.add(tr1);

// Add Annotations to the lines
IAnnotationSet hAnnotation = chart.gethAnnotations();
IAnnotation annoTr1 = hAnnotation.newAnnotation("Point 4", tr1);

// Add annoTr1 to the line, so it will appear on the chart
tr1.addAnnotation(annoTr1);
```

As the scale of the chart or data point value changes, the label will always be displayed at the top of the data point.

Index

3

A

B

C

D

E

F

G

H

I

J

L

M

N

O

P

R

S

T

U

V

W

X

Z