Motif対応XRT/table™

プログラマーズ・ガイド&リファレンス・マニュアル

UNIX Edition ■ Version 3.0

TANGENT
株式会社タンジェント・コンピューティング
〒150-0001 東京都渋谷区神宮前6-6-11
tel:(03)3409-5364
fax: (03)3409-0147
info@tan.co.jp
http://www.tan.co.jp

1999年5月

本書は著作権法により保護されています。(株)タンジェント・コンピューティングによる事前の許可なく、本書のいかなる部分も無断転載、複製、複写を禁じます。
Copyright©1999 (株)タンジェント・コンピューティング

XRT、XRT/3d、XRT/care、XRT/field、XRT/gear、XRT/graph、XRT/table、XRT PDSはKL Group Inc.の登録商標です。
その他製品名、名称、サービス等は該当する会社または組織の商標または登録商標です。

XRT製品ライセンスについて

この度はXRT製品をご購入いただきまして誠にありがとうございます。

本製品のご利用に際しましては、次ページに記載の「XRTバイナリ使用許諾契約書」をご一読くださるようお願いいたします。下記に掲げる事項は、この契約の理解を補助するもので、契約の一部ではありません。

- XRT製品(XRT/graph、XRT/3d、XRT/table等)のライセンスは、機種(SUN SPARC、IBM RS/6000、HP 9000等)につき、またツールキット(Motif)につき許諾されます。
- XRT製品のライセンスにはシングルとネットワークの二種類があります。
- シングル・ライセンスでは、一つのCPUIDで識別されるマシンにおいてエンドユーザ・アプリケーションの開発にXRT製品をご使用いただくことができます。XRT製品を使用する開発には下記の内容が含まれます。
  - XRT製品のライブラリをエンドユーザ・アプリケーションのバイナリにコンパイル、リンクする。
  - XRT製品のビルダーを実行する。
  - XRT製品をGUIビルダーまたはUIMSツールに統合する。XRT製品が統合されているツールはXRT製品が使用許諾されているマシンでのみ使用できます。
- ネットワーク・ライセンスでは、ネットワークに接続されている同ビル内の同機種のマシンにおいてエンドユーザ・アプリケーションの開発にXRT製品をご使用いただくことができます。
- 静的にリンクされているXRT製品のライブラリを含むエンドユーザ・アプリケーションのバイナリは、ロイヤリティーやランタイム費用なしで提供していただくことができます。

XRTバイナリ使用許諾契約書

重要: ソフトウェア・パッケージを開封する前、またはソフトウェアをインストール、使用する前に必ずお読みください。

XRT製品の開封、インストール、使用、またはいずれかの左記の行為により、以下の規定につきお客様が承認されたものとさせていただきます。本契約書はお客様(個人または法人)とKL Group Inc.(以下、「KL Group」という。)の間で締結される法的な契約書です。もし、以下の規定に御同意頂けない場合には、ダウンロードやインストールされたファイルを直ちに破棄し、開封前のパッケージをライセンス費用の御支払いを示す書類と一緒に御返却ください。お客様がお支払になった料金は返却されます。

ソフトウェア・ライセンス

ライセンスの許諾 この使用許諾契約(以下、「ライセンス」という。)は、御指定のCPUと一致する一台のコンピュータにおいて、お客様が御購入されたXRTソフトウェア(以下、「ソフトウェア」という。)を御使用いただく権利をお客様に許諾するものです。ネットワーク・ライセンスを御購入された場合は、御指定のCPUと一致するコンピュータとローカル・エリア・ネットワークで接続された、同機種及び同ビル内に設置されているコンピュータにおいて、御購入いただいたライセンスで許可される使用台数を上限として「ソフトウェア」を御使用いただくことができます。また、共有ライブラリを御購入された場合、「ソフトウェア」のユーザ数は、御購入されたライセンスで許可された数を超えることはできません。「ソフトウェア」は、テンポラリ・メモリ(RAM等)にロードされた時、パーマネント・メモリ(ハード・ディスク、CD-ROM、その他の記憶装置等)にインストールされた時、エンドユーザ・アプリケーションにリンクされた時、開発ツール(コンパイラ、リンカー、GUIビルダー等)によってアクセスされた時に「使用」されたとします。他のコンピュータへ「ソフトウェア」のファイルを配布する目的のみで、ネットワークのファイル・サーバにインストールされている場合は、「使用」されているとはみなされません。

アップデート 本許諾契約書は、KL Groupから提供される「ソフトウェア」の全てのアップグレード版に適用されます。

著作権 「ソフトウェア」とそれに関連するイメージ、写真、テキスト等は、KL Groupとその供給者が所有するもので、その著作権は、カナダと米国の著作権法及び国際条約の規定で保護されています。従って、「ソフトウェア」は、書籍や音楽録音等、著作権で保護される他の製品と同じように扱ってください。「ソフトウェア」は、バックアップ目的で複製する以外は、一切複製作業を行うことはできません。また、「ソフトウェア」に付属する関連資料も同様に扱ってください。

その他の制限 「ソフトウェア」は、貸与、リース、譲渡、リバース・エンジニア、逆コンパイル、逆アセンブル、パスワードの作成をすることはできません。但し、これらの制限が司法管轄区域によって許可されていない場合は、この限りではありません。

「ソフトウェア」で構築されたアプリケーションの配布 「ソフトウェア」の"lib"または"lib-shared"ディレクトリにある、オブジェクト・コードのアプリケーション・ファイルにリンク可能な形態のファイル(以下、「XRTライブラリ・ファイル」という。)は、次の条件に従う場合にのみ、ロイヤリティなしで複製、配布することができます。(1)再リンクが不可能な方法でソフトウェアにリンクされて「XRTライブラリ・ファイル」が配布される。(2)配布されるソフトウェアは、「ソフトウェア」と同等または類似した機能を持つ開発ツールではない。(3)配布するソフトウェアの販売活動に、許可なくKL Groupの社名、ロゴ、トレード・マーク等を使用しない。(4)配布するソフトウェアには、著作権に関する注意書きを明記する。(5)配布したお客様のソフトウェアに対して第三者からなされた損害賠償請求に基づく損害等については、KL Group及びその供給者は一切その責任を負わない。それ以外の場合は、「ソフトウェア」のヘッダー・ファイルや(APIのドキュメンテーションを含む)ドキュメンテーション類を配布することはいかなる方法であってもできません。

U.S. Government Restricted Rights 「ソフトウェア」とその関連資料は、RESTRICTED RIGHTSに基づいて提供されています。本契約書で許諾される「ソフトウェア」とその関連資料は、Federal Acquisition Regulations (FAR) Part 12と Defense FAR Supplement (DFARS) 252.227-7014 (JUN 1995)で定義される"Commercial computer software"です。FAR 12.212 とDFARS 252.227.7014に基づき、この「ソフトウェア」の使用、複写、開示は、本契約書の条項に従います。製造者: KL Group Inc., 260 King Street East, Toronto, Ontario, Canada, M5A 1K3.

保証規定

品質保証 KL Groupは、「ソフトウェア」購入日より60日間、「ソフトウェア」がその関連資料記載の仕様に従って実質的に動作することを保証します。但し、品質保証の期間の制限が司法管轄区域によって許可されていない場合は、適用法例の範囲内でのみ有効です。尚、本品質保証は、第三者に譲渡することはできません。KL Groupによる故意、重大な過失により生じた損害、または製品に対する責任は、KL Groupは除外いたしません。

補償 KL Groupがその判断に基づいて行う補償は、次の事項に限定されます。(1)御支払いになられた金額を払い戻す。(2)上記記載の「品質保証」を満たさない「ソフトウェア」に対して補修、交換する。但し、(1)、(2)いずれの対応も、「ソフトウェア」と共ににその購入を証する書類を返却された場合にのみ提供されます。また、かかる事態が事故、故意若しくは過失、誤用等、KL Groupの責に帰さない理由により生じた場合は、本品質保証は無効になります。尚、交換された「ソフトウェア」の保証期間は、その受領より30日か、または交換の対象になった「ソフトウェア」の保証期間の残りか、どちらか長い方が適用されます。

権利侵害に対する補償 「ソフトウェア」が、第三者の特許、業務上の秘密、著作権等の侵害の対象となった場合、またはKL Groupがそのように判断した場合は、KL Groupは、次の補償を行うことがあります。(1)「ソフトウェア」を継続して使用できる権利を取得する。(2)「ソフトウェア」を権利侵害の対象にならないように交換、修復する。但し、交換または修復された場合も、以前と同じ機能は保つようにする。上記(1)及び(2)が適正でない場合は、(3)「ソフトウェア」のライセンス許諾を中止し、削除する。その場合、ライセンスに対して支払われた金額の内、定額法を適用した5年間の償却期間に基づいた相当金額を返却する。

保証の制限 司法管轄区域の適用法令の範囲内で、KL Group及びその供給者は、「ソフトウェア」とその関連資料の商品性、特定目的適合性の保証を含め、本保証規定に規定されていないその他の保証を、明示たると黙示たるとを問わず一切いたしません。この有限保証は、お客様に特定な法的権利を与えるものですが、州、管轄区域によっては、その他の権利が与えられることもあります。。

免責 司法管轄区域の適用法令の範囲内で、KL Group及びその供給者は、「ソフトウェア」の使用または使用不能から生ずる損害(事業利益の損失、事業の中断、事業情報の損失または金銭的損害を含むがこれらに限定されない)に関して、一切責任を負わないものとします。例え、KL Groupがかかる損害の可能性について知らされていた場合も同様です。但し、州や管轄区域によっては、必然的または偶発的な損害に対する責任の除外や制限を認めていない場合があり、その場合、前述の制限は適応されないことがあります。

適用法令 お客様がEuropean Unionに属する地域に居住されている場合、本契約書はその国の法律に準拠し、契約当事者双方は、その国の司法に従います。それ以外の場合は、本契約書は、カナダ・オンタリオ州の法律に準拠し、契約当事者双方は、オンタリオ集の司法に従います。

パブリック・ドメイン・ソフトウェア 本契約書に記載されている条項は、パブリック・ドメインと明記されているソフトウェアには適用されません。

本契約書は、KL Groupより提供される"XRT FOR MOTIF BINARY LICENSE AGREEMENT"を(株)タンジェント・コンピューティングが日本語訳したものです。表現等に不一致がある場合は、"XRT FOR MOTIF BINARY LICENSE AGREEMENT"(次ページ参照)に従います。

XRT FOR MOTIF BINARY LICENSE AGREEMENT

Important — Read Carefully Before Opening Software Package(s), Installing or Using Software On Your Computer

By opening a sealed software package, or installing, or using this XRT product, you indicate your acceptance of the following XRT for Motif Binary License Agreement. This is a legal agreement between you (either an individual or an entity) and KL Group Inc. If you do not accept and agree with all of the terms of this agreement, promptly destroy any downloaded or installed files, and return the unopened software package(s) and all other materials with proof of payment to your place of purchase, and your license fee will be refunded.

SOFTWARE LICENSE

Please Note: This is a license agreement and not an agreement for sale.

Grant of License. This License Agreement ("License") permits you to use one copy of the software you purchased (the "SOFTWARE"), on a single computer after registration of that computer's CPU identification with us. If you have purchased a network site license you may also use the SOFTWARE on computers in the same building having the same architecture as a registered computer, and connected by the same local area network, provided that the total number of such computers, on which the SOFTWARE is ever used, does not exceed the maximum number permitted by the network site license which you ordered.

If you have purchased shared library versions of the SOFTWARE, the number of users of the SOFTWARE cannot exceed the maximum number permitted by the license which you ordered.

The SOFTWARE is "in use" on a computer when it is loaded into temporary memory (i.e. RAM); when it is installed into permanent memory (e.g. hard disk, CD-ROM, or other storage device); when it is being linked into an end-user application; or when it is accessed from a software development tool such as a compiler, a linker, or a GUI builder. The SOFTWARE is not considered "in use" when installed on a network file server for the sole purpose of file distribution to other computers.

Updates. This agreement will govern the terms of any SOFTWARE updates provided to you by KL Group.

Copyright. The SOFTWARE (including any images, photographs and text incorporated into the SOFTWARE) is owned by KL Group or its suppliers and is protected by Canadian and United States copyright laws and international treaty provisions. Therefore you must treat the SOFTWARE like any other copyrighted material (e.g., a book or musical recording) and not reproduce it except that you may make a backup for archival purposes. You may not copy the printed materials accompanying the SOFTWARE.

Other Restrictions. You may not rent, lease, transfer, reverse engineer, decompile, disassemble, create passwords for or translate the Software, except to the extent such foregoing restriction is expressly prohibited by applicable law.

Distribution of applications you build with the SOFTWARE. You have a royalty-free right to reproduce and distribute files in the SOFTWARE that are found in the "lib" or "lib-shared" directories and which are in a form suitable for linking into object code application files (the "XRT Library Files") provided that (a) you distribute the XRT Library Files only when linked into your software product in such a way as to prohibit further relinking; (b) your software product is not an application development tool that provides similar functionality to the SOFTWARE licensed hereunder; (c) you do not use KL Group's name, logo or trademark to market your software product; (d) you include a valid copyright notice on your software product; and (e) you agree to indemnify, hold harmless, and defend KL Group and its suppliers from and against any claims or lawsuits, including attorney's fees, that arise or result from the use or distribution of your software product. Unless otherwise specified, you may not in any manner distribute any header files or documentation (including API documentation) provided as part of the SOFTWARE.

U.S. Government Restricted Rights. The SOFTWARE and documentation are provided with RESTRICTED RIGHTS. The SOFTWARE (including documentation) licensed in this agreement is "Commercial computer software" as defined in Federal Acquisition Regulations (FAR) Part 12 and Defense FAR Supplement (DFARS) 252.227-7014 (JUN 1995). In accordance with FAR 12.212 and DFARS 252.227.7014, notwithstanding any provision to the contrary, use, duplication and disclosure of this software is governed by the terms of this agreement. Manufacturer is KL Group Inc., 260 King Street East, Toronto, Ontario, Canada, M5A 1K3.

LIMITED WARRANTY

Limited Warranty. KL Group warrants that the SOFTWARE will perform substantially in accordance with the accompanying written materials for a period of sixty (60) days from the date of receipt. Some states/jurisdictions do not allow limitations on duration of an implied warranty, so the above limitation may not apply to you. This warranty may not be assigned. KL Group does not exclude product liability or liability for damages resulting out of intent or gross negligence on the part of KL Group or its leading personnel.

Customer Remedies. KL Group's and its suppliers' entire liability and your exclusive remedy shall be, at KL Group's option, either (a) return of the price paid, or (b) repair or replacement of SOFTWARE that does not meet KL Group's Limited Warranty and which is returned to KL Group with a copy of your receipt. This Limited Warranty is void if failure of the SOFTWARE has resulted from accident, abuse, or misapplication. Any replacement SOFTWARE will be warranted for the remainder of the original warranty period or thirty (30) days, whichever is longer.

Customer Remedies for Infringement. Should the SOFTWARE become or in KL Group's opinion be likely to become the subject of a claim for infringement of patent, trade secret or copyright, KL Group may (a) procure for you the right to continue to use the SOFTWARE; or (b) replace or modify the SOFTWARE to make it non-infringing, provided that the same function is performed by the replacement or modified SOFTWARE; or if (a) or (b) are not commercially reasonable, (c) terminate the license to use the SOFTWARE, remove the SOFTWARE, and grant you a credit for the license fee as depreciated on a straight-line five (5) year basis.

No Other Warranties or Obligations.. To the maximum extent permitted by applicable law, KL Group and its suppliers shall have no other obligations and disclaim all other warranties, either express or implied, including, but not limited to, implied warranties of merchantability and fitness for a particular purpose, with regard to the SOFTWARE and the accompanying printed materials. This limited warranty gives you specific legal rights. You may have others which vary from state/jurisdiction to state/jurisdiction.

No Liability for Consequential Damages. To the maximum extent permitted by applicable laws, in no event shall KL Group or its suppliers be liable for any damages whatsoever (including without limitation, damages for loss of business profits, business interruption, loss of information, or other pecuniary loss) arising out of the use of or inability to use this KL Group product, even if KL Group has been advised of the possibility of such damages. Because some states/jurisdictions do not allow the exclusion or limitation of liability for consequential or incidental damages, the above limitation may not apply to you.

Governing Law. If you are a resident of one of the member states of the European Union, this Agreement is governed by the laws of your country of residence, and each of the parties hereto irrevocably attorns to the jurisdiction of the courts of your country of residence. Otherwise, this Agreement is governed by the laws of the Province of Ontario, Canada. Each of the parties hereto irrevocably attorns to the jurisdiction of the courts of the Province of Ontario.

Public Domain Software. This license agreement and the warranties, remedies, limitations, conditions and restrictions contained herein do not apply to any software clearly marked as being in the public domain.

BINLIC-M-11/98

目次

はじめに 1

XRT/tableについて 1
XRT/table 3.0の新しい機能 2
XRT/table 3.0での変更点 3
前提知識 5
表記上の規約 5
関連資料 5
テクニカル・サポート 5
お客様のご意見 6

Part I　XRT/tableの使用法

1　XRT/tableの 簡単なプログラムの開発 11

1.1　環境変数 12
1.2　プログラムのコンパイルと許可 12
1.3　基本的なテーブル 13
1.4　テーブル・サイズの設定 16
1.5　列ラベルの追加とテーブルのフォーマット 16
1.6　テーブルの一部のフォーマット 18
1.7　タイトルを追加 19
1.8　ここから先に進むには 20

2　XRT/tableの基本操作 21

2.1　Motifとの統合 21
2.2　テーブルの構成 23
2.3　ウィジェットの管理 24
2.4　テーブルのContext 25
2.5　ContextとSeriesの参照 26
2.6　ポインタのリソース 28
2.7　XmStringリソースとMotif 2.xの制限 29
2.8　色 29
2.9　警告とエラーメッセージ 30
2.10　リソース・ファイルの使用 31

3 XRT/tableのプログラミング 33

3.1 リソース 33
3.2 行と列の数 40
3.3 表示されている行と列 41
3.4 フリーズされた行または列 41
3.5 行と列のラベル 42
3.5.1 ラベル値の設定と取得 43
3.5.2 ラベルの位置と間隔 44
3.5.3 ラベル作成のコールバック 45
3.6 テーブルのタイトル 46
3.7 行の高さと列幅 48
3.7.1 character指定 49
3.7.2 pixel指定 50
3.7.3 可変の高さと幅 50
3.7.4 フレキシブルな行と列 51
3.7.5 行や列を非表示にする 51
3.8 セルとラベルのマージン 52
3.9 前景色、背景色と影の色 53
3.10 セルとラベルのテキストの行揃え 55
3.11 セルとラベルのフォント 55
3.12 罫線の種類と場所 56
3.13 背景のPixmap 58
3.14 テキストのclippingとoverflow 60
3.15 セルとラベルのspanning 62
3.16 セルとラベルのフラッシュ 63
3.17 ユーザのデータ 65
3.18 テーブルの印刷 65
3.18.1 PostScriptの生成 65
3.18.2 PostScriptフォントのカスタマイズ 68
3.18.3 プリント・コールバック 69

4 XRT/tableデータ 73

4.1 XrtTblCellValues構造体 73
4.2 テーブルにデータを取り込む 76
4.3 テーブル・データの操作 77
4.3.1 セル値の設定と取得 77
4.3.2 行と列の追加 79
4.3.3 行と列の削除 80
4.3.4 行と列の移動 80
4.3.5 セル値をクリアする 81

4.3.6 テーブル全体をクリアする....81
4.4 複数のXrtTblCellValues構造体の使用....81
4.5 セル値のコールバック....82
4.6 テーブルからデータを取り出す....83

5 ユーザ・インタラクションの プログラミング....85

5.1 デフォルトのユーザ・インタラクション....85
5.2 ユーザ・インタラクションとトランスレーションのカスタマイズ....87
5.3 リスト表示モード....88
5.4 テーブルのスクロール....89
5.4.1 スクロールバー....89
5.4.2 スクロールのプログラミング....91
5.4.3 スクロール・コールバック....92
5.5 行と列のリサイズ....92
5.6 セルのトラバース....94
5.7 セルの編集....96
5.8 セルの選択....99
5.9 マウス・ポインタ....104
5.10 ドラッグ&ドロップ....106

6 高度なプログラミング....109

6.1 セル/ラベル内のウィジェット....109
6.1.1 テーブル内のウィジェット....110
6.1.2 リソース・ファイルのウィジェット....115
6.2 セル/ラベル値の検証....116
6.3 XRT/fieldを一緒に使用する....119
6.4 テーブルのサイズとリサイズ....122
6.5 複合ストリング....124
6.6 Seriesリソースのプログラミング....126
6.7 共通のセル値を持つ複数のテーブル....128
6.8 XRT/graphとXRT/3dのデータ....130
6.9 ドラッグ&ドロップのプログラミング....130
6.9.1 ドラッグ&ドロップのターゲット....130
6.9.2 区切り文字の指定....132
6.9.3 ExportとImportコールバック....133
6.9.4 独自のターゲットを定義する....134
6.9.5 データ転送フォーマット....136
6.10 XRT/tableの国際化....137
6.11 パフォーマンスについて....137
6.11.1 一般的な場合のパフォーマンス....137
6.11.2 インタラクティブ機能に関するパフォーマンス....138

6.11.3 表示に関するパフォーマンス 138
6.11.4 セル値の処理 139
6.11.5 contextを管理する 139
6.12 警告とエラー・メッセージのカスタマイズ 141
6.13 C++の使用 142
6.14 UILの使用 142
6.15 リソース・ファイル 143

Part II リファレンス

A リソースの一覧 147

A.1 ウィジェットの概要 147
A.2 XRT/tableのリソース 148
A.3 Constraintリソース 189
A.4 継承されたリソース 190

B 手続きと方法 191

C マクロ 221

D トランスレーションとアクション 223

D.1 デフォルトのトランスレーション 223
D.2 アクション・ルーチン 225

E データ型 229

F リソースの型とクラス 249

G リソース・ファイルの使用 257

G.1 構文 257
G.2 リソースファイルの例 262

H UIL引数 265

I XrtLabelウィジェット 271

I.1 ウィジェットの概要 271
I.2 XrtLabelについて 271
I.3 XrtLabelを使用する 272
I.4 XRT/tableと一緒に使用する 275
I.5 リソース 276

I.6 リソースの型とクラス 278

J サンプル・プログラム 279

J.1 例プログラム 279
J.2 デモ・プログラム 280
J.3 simple.c 281

索引 283

はじめに

XRT/tableについて ■ XRT/table 3.0の新しい機能
XRT/table3.0での変更点 ■ 前提知識
表記上の規約 ■関連資料
テクニカル・サポート
お客様のご意見

XRT/tableについて

XRT/tableウィジェット

XRT/tableは、ボタン、スライダ、メニュー等のウィジェットから構成されているMotifに新たなウィジェットを追加します。このテーブル・ウィジェットは、スクロール・ウィンドウにテキスト情報を始めpixmapやウィジェットを行と列の形式で表示し、ユーザは、このテーブルとインタラクションを行うことができます。

XRT/tableウィジェットには、テーブルの表示や動きをコントロールするリソースが提供されています。ウィジェットが一つ増えた点を除けば、XRT/tableを使用したプログラミングは、他のMotifのプログラミングとほとんど違いはありません。

XRT/tableは、XRTファミリのXRT/graphやXRT/3d、XRT/fieldと組み合わせて使用することができます。XRT/graphは2Dデータを線、棒、円グラフ等に、XRT/3dは3Dデータをサーフェス、等高線、3D棒グラフ等に表示します。XRT/fieldは、入力された値を変換、補完、検証するデータ入力フィールド・ウィジェットです。

XRT/tableには、下記の内容をコントロールするリソースがあります。

- テーブルまたは、表示される部分の行や列の数
- セルの内容(テキスト値やpixmap、ウィジェット)
- 行の高さと列幅
- 罫線、行揃え、フォント、色等のセルとラベルの属性
- 選択の方法等のユーザ・インタラクション
- セルの編集とトラバース
- セルやラベルに複数のフォントを表示できる複合ストリング

XRT/tableには、次の作業を行う手続きや方法が提供されています。

- 行や列の追加、削除、移動

- PostScript出力
- ASCIIファイルやXRT/graph、XRT/3dのデータ・セットからのセル値の読み込み
- セル値のデータ検証
- セル範囲の設定や取得の支援
- ASCIIファイルやSYLKフォーマットへのセル値の書き込み

XRT/table 3.0の新しい機能

XRT/table 3.0の主な新しい機能とそれが解説されているマニュアルのページです。

表示に関する改良点

- カーソルの設定: カーソルを設定できるようになりました。(Appendix B *"XrtTblSetAllCursors()"*、*"XrtTblRestoreAllCursors()"*)
- セル内のウィジェットの配置: セル内でのウィジェットの配列をコントロールする新しいconstraintリソースが追加されました。ウィジェットをセルのサイズに合わせてリサイズさせることができるようになりました。(Appendix A)

プログラミングに関する改良点

- Context毎のセンシティビティ: セルのセンシティビティをカレントcontext毎にセットできるようになりました。(Appendix A)
- リスト表示モードの改良: リスト表示モードの機能や動作がよりXmListに近くなりました。(P88、5.3)

編集に関する改良点

- ComboとSpin Box: セル値をcomboまたはspin boxを使用して編集できるようになりました。(P96、5.7)
- 検索機能: 行や列の値、その他の値を使用して範囲や範囲リストの検索を行うことができるようになりました。(Appendix B *"XrtTblRangeListFindRange()"*、*"XrtTblRangeListFind"*)
- プレビュー・スクロール: スクロールバーの隣に小さなウィンドウを表示するリソースが追加されました。(Appendix A)

その他

- 新しいデモと例プログラム
- バグの修正とマイナーな改良(修正されたバグについては、リリースノートを参照してください

XRT/table 3.0での変更点

XRT/table 3.0は、XRT/table 2.xと互換性があります。既存のアプリケーションでXRT/table 3.0を使用する場合は、次の変更点に留意してください。

重要な変更点	
リコンパイル	**このリリースのXRT/tableを使用するには、既存のアプリケーションを(再リンクだけではなく)再コンパイルしてください。**
Motif1.2以上が必要	Motif 1.2**以上が必要です。**Motif 2.x**もサポートされています。**
セル値の構造体 (P73、4.1)	XrtTblCellValues**構造体には、セル値を**String**または**XmString**のどちらで記憶させるかどうかを指定する新しいメンバ、***set_mode***が追加されています。**
ドラッグ&ドロップ (P106、5.10)	**ドラッグ&ドロップのAPIが変更されました。**XRT/table 2.0**のドラッグ&ドロップ機能を使用するには、テーブルの作成時に**XmNxrtImportTargets**をNULLにしてください。(ドラッグ&ドロップは、なるべくこのバージョンの機能を使用するようにしてください。)**
クローン化されるウィジェット (P110、6.1.1)	**位置が**XmNxrtTblLocation**で指定されると、ウィジェットはクローン化されます。(位置が指定された後は、**XmNxrtTblDisplayWidgetCallback**リストにコールバックは追加できません。)**
ValidateCellコールバック (P185 XmNxrtTblValidateCellCallback)	XrtTblValidateCellCallbackStruct**コールバック構造体の**XRTTBL_REASON_VALIDATE_CELL(reason)**は、**XRTTBL_REASON_VALIDATE_CELL_BEGIN**に変更されました。**

その他にも、下記のような機能が変更されています。

列挙型はunsigned char	XRT/table**の全ての列挙型は、**(OSF**の標準に準拠させるため)**unsigend char**型になりました。この型を**int**型としていなければ、影響はありません。**
ウィジェットを含むセル/ラベルの罫線	XmNborderWidth**が0以外の値にセットされている場合にのみ、ウィジェットを含むセルまたはラベルの周囲に罫線が描かれます。以前の状態を使用するには、**XmNborderwidth**を1にセットします。**
XmNxrtTblLabelCreateFunc(P45、3.5.3)	XmNxrtTblLabelCreateFunc**は、**XmNxrtTblLabelCreateCallback **に置き換わりました。**XmNxrtTblLabelCreateFunc**は、このバージョンではサポートされています。**
XmNxrtTblPSFontList-Context と XmNxrtTblPS-FontListSeries(P65、3.18)	XmNxrtTblPSFontListContext**と**XmNxrtTblPSFontListSeries**は、**XmNxrtTblPSFontMapList**に置き換わりました。これらのリソースは、このバージョンではサポートされています。**

XmNxrtTblCellValueFunc (P82、4.5)	XmNxrtTblCellValueFuncは、XmNxrtTblCellValueCallback に置き換わりました。XmNxrtTblCellValueFuncは、このバージョンではサポートされています。
XmNxrtTblRepaint(P177)	XmNxrtTblRepaintには、XrtTblRepaint型の値が必要です。有効な値は、XRTTBL_REPAINT_ON(再描画する)、XRTTBL_REPAINT_OFF(再描画はしないが、サイズは再計算する)、XRTTBL_REPAINT_NOCALC(再描画も再計算もしない)です。
XrtTblCreateWidgetCallbackStruct(P234)	Motifの標準に準拠するため、*event*フィールドが追加されました。
XrtTblDisplayWidgetCallbackStruct(P234)	Motifの標準に準拠するため、*event*フィールドが追加されました。
XrtTblResizeCallbackStruct(P243)	Motifの標準に準拠するため、*event*フィールドが追加されました。
XrtTblValidateCellCallbackStruct(P246)	Motifの標準に準拠するため、*event*フィールドが追加されました。
XrtTblScrollCallbackStruct (P229)	この構造体には、二つのメンバが追加されました。XmNxrtTblTopRowまたはXmNxrtTblLeftColumnリソースの新しい値(int型)と、プレビュー・ウィンドウに表示されるテキスト(string型)です。

前提知識

このマニュアルは、ユーザがC言語及びMotifツールキットに精通していることを前提としています。MotifやXRT/tableアプリケーションをプログラミングするには、C言語の“char *配列”、“構造体のポインタ”等を理解している必要があります。またリソース値の取得や設定等のMotifの基本的な概念も理解していなければなりません。Motifに関する資料については“関連資料”をご覧ください。

表記上の規約

`Typewriter Font`

- Cコードやファイル内容の例
- XRT/tableやMotifリソース名、定数
- 端末で入力するコマンド

斜体

- パス、ファイル名、プログラム、手続き、パラメータ
- 新しい用語や重要な用語
- 図やテーブル
- このマニュアルで参照されている参考資料名(Motif Programming Manual等)

関連資料

- *Motif Programming Manual*と*Motif Reference Manual*(O'Reilly & Associates,Inc出版)
- *X Toolkit Intrinsics Reference Manual*、*Xlib Programming Manual*、*Xlib Reference Manual*(O'Reilly & Associates,Inc.出版) これらのマニュアルはXRT/tableやMotifの開発には直接必要ありませんが、Xlibの知識はX Windowアプリケーションの開発に重要です。

テクニカル･サポート

初期の問題の多くは、インストールやセットアップに関する問題です。リリースノート&インストールガイドには、インストールやセットアップの際の問題の回避法がありますので参照してください。

XRTテクニカル･サポートへお問い合わせされる場合は、次の事項をご連絡ください。

- XRT/tableライセンスのシリアル番号
- XとMotifのバージョン
- OSのバージョン

尚、テクニカル･サポートを御受けいただくには、XRT/careと呼ばれる、保守サポート･サービスを御購入されていることが前提となります。XRT/careのお申し込み法やサービスに関する御質問等がございましたら、(株)タンジェント･コンピューティングまでご連絡ください。

オンライン・サポート

(株)タンジェント・コンピューティング及び、開発元のKL Group Inc社のwebサイトには、技術的な情報が掲載されていますので参照してください。

技術情報及びFAQ **(日本語)**	http://www.tan.co.jp/xrt/support/index.htm
技術情報(英語)	http://www.klgroup.com/cs/tech/xrt/
Knowledge Base(**英語**)	http://www.klgroup.com/cgi-bin/xrt_search.cgi
FAQ(**英語**)	http://www.klgroup.com/cs/tech/xrt/faq/

お客様のご意見

XRT製品のアップグレードで追加される新しい機能の大半は、お客様のリクエストに基づいています。XRT/tableに関するお客様のご意見、ご使用例、或は、あれば便利と思われる機能等がございましたら下記までご連絡ください。

株式会社タンジェント・コンピューティング
〒150-0001東京都渋谷区神宮前6-6-11
電話　03-3409-5364
Fax　03-3409-0147
Email info@tan.co.jp
WWW　http://www.tan.co.jp/

Part I

Using XRT/table

1

XRT/tableの簡単なプログラムの開発

環境変数 ■ プログラムのコンパイルと許可
基本的なテーブル ■ テーブル・サイズの設定
列ラベルの追加とテーブルのフォーマット
テーブルの一部のフォーマット
タイトルを追加 ■ ここから先に進むには

ここでは、サンプルのプログラムを実際にコンパイル、実行しながらXRT/tableの基本的なプログラミングを紹介していきます。このアプリケーションは、架空の会社の従業員の情報を表示します。完成したプログラム、*simple.c*は、*$XRTHOME/src/table/examples*とAppendix Jにあります。

次の表は、表示されるデータです。

Name	Address	Phone Number	Position	Salary
James Q. Doohan	1701 Planetia Blvd. Anytown, U.S.A.		President	$245,000
John F. Kricfalusi	983 Nickelodon Street Anytown, U.S.A.	(999)555-9876	Animator	$10,000
Marc Lenard	6 Gene Crescent Anytown, U.S.A.	(999)555-1212	Telemarketer	$10 / hr.
Hikaru I. Takei	134 Adelaide Street E. Suite 204 Anytown, U.S.A.	(999)594-1026	OEM Sales	50%
Melissa A. Truman	475 Woodview Line Anytown, U.S.A.	(999)555-9030	Technical Writer	$50,250
Stephanie L. Truman	388 Appleby Road Anytown, U.S.A.	(999)555-2642	System Analyst	$85,750
Bill West	1001 Spumco Way Anytown, U.S.A.	(999)555-9966	Announcer	$17,500

1.1 環境変数

開始する前にUNIXの環境をご確認ください。*XRTHOME*という環境変数が、XRT/tableのファイルのディレクトリ名に定義されていなければなりません。

次のように入力して、*XRTHOME*の現在の値を確認できます。

```
echo $XRTHOME
```

もし*XRTHOME*環境変数が正しく設定されていない場合は、リリース&インストレーション・ガイドを参照するか、システム管理者の方にお問い合わせください。

またMotifのアプリケーションがコンパイルできることもご確認ください。ご使用になっているMotifのマニュアルで、Motifのライブラリとインクルード・ファイルの位置を調べることができます。XRT/tableの使用を開始される前に、簡単なMotifのアプリケーションがコンパイル、実行できることを必ずご確認ください。

1.2 プログラムのコンパイルと許可

XやMotifのライブラリとインクルード・ファイルが標準の場所(*/usr/lib*、*/usr/include/X11*、*/usr/include/Xm*)にインストールされていれば、次のようにして、*simple.c*をコンパイル[1]し、許可を与えることができます。

```
cc    simple.c -I$(XRTHOME)/include \
  -L$(XRTHOME)/lib -o simple \
  -lxrttable -lXm -lXpm -lXt -lX11
$XRTHOME/bin/xrt_auth simple
```

もし、ライブラリとインクルード・ファイルが標準の位置になければ、`-I`と`-L`のコンパイラ・フラグが必要です。

xrt_auth

XRT/tableのライブラリが実行可能アプリケーションにリンクされたら、*xrt_auth*を使用して許可を与えます。*xrt_auth*はご使用のマシンにXRT/tableのライセンスがあるかを調べ、アプリケーションに許可を与えます。*xrt_auth*には、*-v*(verbose message;詳細メッセージ)と*-q*(quiet mode;メッセージ出力なし)の二つのフラグのオプションがあります。

```
xrt_auth [-q | -v] executable_file
```

[1] コンパイルのオプションは機種に依存します。ご使用のマシンでのコンパイルの例が*$XRTHOME/src/table/examples*の*Makefile*にあります。

エンドユーザ・アプリケーションは*xrt_auth*により許可が与えられると、どのマシンでも実行することができます。XRT/tableで作成されたアプリケーションにはランタイム費用はかかりません。

もし、アプリケーションが*xrt_auth*によって許可が与えられないと、テーブルは数秒後に消滅し、図1のメッセージが現われます。(詳細なメッセージが*stdout*に書かれます。)



図1　期限切れまたは未許可の場合のメッセージ

注意: 共用ライブラリのリンクや許可については、リリースノートを参照してください。

C++とANSI C

XRT/tableのインクルードファイルは、C++、ANSI C、"Kernighan and Ritchie" Cと互換性があります。

1.3　基本的なテーブル

*simple.c*を開始する前に、*table1.c*(図3参照)をコンパイル、実行します。*table1.c*は、ファイルからテキスト・データをロードし、Motifのシェル・ウィジェットを作成、そしてテーブル・ウィジェットを作成する最小限のCプログラムです。図2は、このプログラムの実行結果です。



図2　*table1.c*によって作成されたウィンドウ

このプログラムの大半は、一般的なMotifのウィジェットに関するコードです。9行目では、(`values`によって指されている)`XrtTblCellValues`構造体にスペースを割り当て、22行目で*simple.dat*ファイルのテキスト・データをロードしています。19行目は、Intrinsicsをイニシャライズし、標準のXt Intrinsicsの方法でトップレベルのシェル・ウィジェットを作成しています。

23から27行目では標準のXt Intrinsicsの手続き*XtVaCreateManagedWidget()*を使用して、`toplevel`の子であるテーブル・ウィジェットを生成しています。`XmNxrtTblCellValues`と`XmNxrtTblXmSetStringMode`を除く全てのテーブルのリソースは、デフォルト値のままにしておくことができます。

Motif 2.xにリンクする場合は、`XmNxrtTblXmStringSetMode`をセットし、XmString値をStringまたはXmStringsのどちらで設定/取得するかを指定します。このリソースは、テーブルの作成時にしかセットできません。

標準のXtの手続き(*XtSetValues()*、*XtGetValues()*、*XtVaSetValues()*)をテーブル・ウィジェットの操作に使用できます。XRT/tableはMotif Managerクラスのサブクラスですので、テーブルのリソースの他に、Manager、Constraint、Composite、Coreのリソースも使用できます。(例えば、テーブルの`XmNbackground`リソースを設定することができます。)

エンドユーザ・インタラクション

XRT/tableは、データを表示するだけではなく、エンドユーザとのインタラクションもサポートしています。デフォルトで、エンドユーザは次のようなインタラクションを行うことができます。

- セルの編集: XRT/tableは、標準のXmTextウィジェットを使用し、入力フォーカスがあるセル(*current cell;カレント・セル*)を編集できます。
- 行や列のリサイズ: ユーザは、インタラクティブに行の高さや列幅を変更することができます。
- テーブルのスクロール: テーブルのサイズがウィンドウより大きくなると、XRT/tableは二つの標準のXmScrollBarウィジェットを使用し、ユーザはテーブル全体をスクロールできます。
- セルの選択: ユーザは、インタラクティブにセルを選択できます。(デフォルトでは、不可になっています。)
- ドラッグ&ドロップ: ユーザはテキストを選択し、テーブルの他の部分に移動/コピーしたり、他のテーブルやXtSelectionsからテキストをコピーすることができます。

```
1 #include <Xm/XrtTable.h>
2
3 int
4 main(int argc, char *argv)
5 {
6   Widget toplevel, table;
7   XrtTblCellValues *values;
8   FILE *fp;
9
10  XtSetLanguageProc(NULL,NULL,NULL);  /* initialize locale */
11
12  /* Open data file */
13  if (!(fp = fopen("simple.dat", "r"))) {
14      perror("simple.dat");
15      exit(0);
16  }
17
18  /* Initialize Intrinsics */
19  toplevel = XtInitialize(argv[0], "Simple", NULL, 0, &argc, argv);
20
21  /* Read data values, and load into table */
22  values = XrtTblReadAscii(toplevel, fp, ' , ');
23  table = XtVaCreateManagedWidget("table",
24      xtXrtTableWidgetClass, toplevel,
25      XmNxrtTblCellValues, values,
26      NULL);
27  XrtTblDestroyCellValues(values);
28  fclose(fp);
29
30  XtRealizeWidget(toplevel);
31  XtMainLoop();
32 }
```

図3 *table1.c*

セル値のロード

XRT/tableのプログラミングでは、テキスト・データ(*cell values*)をXRT/tableが理解できるフォーマット(`XrtTblCellValues`構造体)にロードすることが必要です。ファイルからセル値を割り当て、ロードするには、*XrtTblReadAscii()*を使用します。[1]

図4は、ここの例で使用されている*simple.dat*ファイルの内容です。このファイルは、*XrtTblReadAscii()*が理解できるフォーマットに構成されたASCIIファイルです。各行はテーブルの1行分のセル値を持ち、各列はカンマで区切られています。特殊文字の前には、バック・スラッシュを配置します。(例えば、"\,"とすると、カンマがテキストの一部として扱われます。)

[1] より一般的な方法として、CでXRT/tableのデフォルトの`XrtTblCellValues`構造体をセットする方法があります。

```
James Q. Doohan,1701 Planetia Blvd.\nAnytown\, U.S.A.,,President,$245\,000
John F. Kricfalusi,983 Nickelodeon Street\nAnytown\, U.S.A.,(999) 555-9876,Animator,$10\,000
Marc Lenard,6 Gene Crescent\nAnytown\, U.S.A.,(999) 555-1212,Telemarketer,$10 / hr.
Hikaru I. Takei,134 Adelaide Street E.\nSuite 204\nAnytown\, U.S.A.,(999) 594-1026,OEM Sales,50%
Melissa A. Truman,475 Woodview Line\nAnytown\, U.S.A.,(999) 555-9030,Technical Writer,$50\,250
Stephanie L. Truman,388 Appleby Road\nAnytown\, U.S.A.,(999) 555-2642,System Analyst,$85\,750
Bill West,1001 Spumco Way\nAnytown\, U.S.A.,(999) 555-9966,Announcer,$17\,500
```

図4 *simple.dat*ファイル

1.4 テーブル･サイズの設定

図2にあるテーブルの終わりには、3行の空白行があります。テーブルの行や列の数を指定しなければ、テーブルはデフォルトで10行、5列にセットされます。

テーブルの全てのリソースは、テーブルの作成時に指定することができ、*XtSetValues()*または*XtVaSetValues()*を使用して、プログラムが実行している時にいつでも変更できます。*XtGetValues()*または*XtVaGetValues()*を使用すると、リソースの現在の値を調べることができます。[1]

この例では、7人の従業員の情報があるので、`XmNxrtTblNumRows`と`XmNxrtTblNumColumns`をセットするように、次の2行を*XtVaCreateManagedWidget()*のコールに追加します。[2]

```
XmNxrtTblNumRows,  7,
XmNxrtTblNumColumns,   5,
```

1.5 列ラベルの追加とテーブルのフォーマット

*table1.c*で表示されているテーブルには、列ラベルがなく、フォーマットも設定されていません。ここでは、ラベルの設定やテーブルのフォーマットを解説します。

ラベルの設定

テーブルに列ラベルを追加するには、次の行を*XtVaCreateMangedWidget()*のコールに加えます。

```
XmNxrtTblColumnLabels, &labels,
```

[1] 多くのリソース値は、ユーザ･インタラクション等により、頻繁に変更されます。

[2] テーブルの行/列と`XrtTblCellValues`構造体の関係は、実際にはもう少し複雑です。詳細は、第4章"XRT/tableのデータ"を参照してください。

このリソースは、XrtTblXmStringList構造体を値として取ります。次のコードは、列ラベルのテキストをXrtTblXmStringList構造体にロードします。

```
XrtTblXmStringList labels;

static String label_list[] =
  { "Name", "Address", "Phone Number",
      "Position", "Salary" };
labels.length = XtNumber(label_list);
labels.list = (XmString *)label_list;
```

テーブルのリサイズ

図2のテーブルは、Address列のテキストを全て表示していません。セルの右下に表示されている*clipping arrows*は、そのセルが全てのテキストを表示していないことを示しています。

ユーザ自身が全てのテキストを見るために行や列をリサイズすることもできますが、各セルのテキストが最初から全て表示されている方が便利なこともあります。次の例のように*XtVaCreateManagedWidget()*のコールに追加することにより、XRT/tableはセルに含まれている全てのテキストが表示できるよう、行や列を自動的にリサイズします。

```
XmNxrtTblPixelWidthContext,     XRTTBL_VARIABLE,
XmNxrtTblPixelHeightContext,    XRTTBL_VARIABLE,
```

テーブルのフォーマット

図5は、今までの変更とテーブルの背景と前景色を設定した結果です。以下のコードは、新しいテーブルを作成します。

```
table = XtVaCreateManagedWidget("table",
  xtXrtTableWidgetClass,        toplevel,
  XmNxrtTblNumRows,             7,
  XmNxrtTblNumColumns,      5,
  XmNxrtTblCellValues,      values,
  XmNxrtTblColumnLabels,        &labels,
  XmNxrtTblBackgroundContext,   "honeydew1",
  XmNxrtTblForegroundContext,   "blue4",
  XmNxrtTblPixelWidthContext,   XRTTBL_VARIABLE,
  XmNxrtTblPixelHeightContext,  XRTTBL_VARIABLE,
  NULL);
```



図5　基本的なフォーマットが加えられたtable1

1.6 テーブルの一部のフォーマット

テーブルは、ユーザのニーズに合うようカスタマイズすることができます。ここでは、列ラベルをセルと区別し、更に、”President”のサラリーをハイライトしてみます。

テーブルの一部の属性だけを変更する時、*XtVaSetValues()*は、テーブルの特別なリソース`XmNxrtTblContext`をセットする必要があります。このリソースは、*context*と呼ばれるテーブルの“部分”を指定します。[1] 現在のcontextに適用されるリソースは、Contextリソースと呼ばれ、名前の終わりに“Context”が付きます。

contextを設定(*XrtTblSetContext()*を使用)せずにContextリソースをセットしようとすると、XRT/tableは一番最後にセットされたcontextを使用します。*table1.c*ではcontextを設定していないので、Contextリソースはテーブル全体(`XmNxrtTblContext`のデフォルト値)に適用されます。

以下のコードは、contextを全ての列ラベルにセットし、そのcontextの色と行揃えを指定しています。

```
XtVaSetValues(table,
  XmNxrtTblContext,XrtTblSetContext(XRTTBL_LABEL, XRTTBL_ALL),
  XmNxrtTblBackgroundContext, "blue",
  XmNxrtTblForegroundContext, "white",
  XmNxrtTblAlignmentContext, XRTTBL_ALIGNMENT_TOPCENTER,
  NULL);
```

次のコードは、”President”のサラリーを表示するセルのcontextをセットし、異なる前景色を指定します。

```
XtVaSetValues(table,
  XmNxrtTblContext,             XrtTblSetContext(0,4),
  XmNxrtTblForegroundContext,   "red",
  NULL);
```

[1] *context*は、次の章に解説されています。

1.7 タイトルを追加

タイトルを付けるとテーブルが更に見易くなります。ここでは、テーブル・ウィジェットの上にXrtLabelウィジェットを追加してタイトルを表示します。[1] 次のコードは、XrtLabelを追加し、幾つかのXrtLabelリソースを設定しています。

```
title = XtVaCreateWidget("title",
  xtXrtLabelWidgetClass,    table,
  XmNxrtLblBorderType, XRTLBL_BORDER_SHADOW,
  XmNlabelString,       "Employee Listing",
  XmNxrtTblLocation,         XRTTBL_LOCATION_TOP,
  XmNxrtTblTopOffset,        10,
  XmNxrtTblBottomOffset,     20,
  XtVaTypedArg,              XmNfontList,
  XmRString,
  "-*-times-bold-r-*-*-*-180-*-*-*-*-iso8859-1",
  strlen("-*-times-bold-r-*-*-*-180-*-*-*-*-iso8859-1")+1,
  NULL);
```

この章で*table1.c*に加えられた変更は、*simple.c*にまとめてあります。このプログラムは、*$XRTHOME/src/table/examples*とAppendix Jにあります。図6は、*simple.c*がコンパイル、実行された結果です。



図6　simple.cのウィンドウ

[1] XRT/tableの上には、他のMotifウィジェットを置くこともできます。XrtLabelは、テーブルのタイトル用に設計されたPrimitiveのウィジェットです。詳細はAppendix Iを参照してください。

1.8 ここから先に進むには

短期間でXRT/tableのプログラミングを習得するための二つの提案です。

- 第2章は、XRT/tableの基本的な操作の解説です。この章を読むことにより、XRT/tableの基礎知識を得ることができます。
- XRT/tableには、例やデモ・プログラムが提供されています。これらを実行したり、そのソースコードを調べることにより、XRT/tableプログラミングのテクニックを習得することができます。Appendix Jに、これらのプログラムの簡単な解説があります。

2

XRT/tableの基本操作

Motifとの統合 ■ テーブルの構成
ウィジェットの管理 ■ テーブルのContext
ContextとSeriesの参照 ■ ポインタのリソース
XmStringリソースとMotif 2.xの制限 ■ 色
警告とエラーメッセージ ■ リソース・ファイルの使用

XRT/tableウィジェットは、テキストやpixmap、ウィジェットを行や列に表示するスクロール可能なウィンドウです。XRT/tableは、行や列のリサイズ、テキストの編集、スクロール、セルの選択等のユーザ・インタラクションを自動的に処理します。

2.1 Motifとの統合

XRT/tableはMotifツールキットに新たなウィジェットを追加します。このウィジェットは、他のMotifプログラミングと同様、Xt Intrinsicsで操作することができます。

Xt Intrinsicsは、ウィジェットをクラス編成するオブジェクト指向アーキテクチャを定義します。例えば、XmPushButtonウィジェットは、Xt Coreウィジェットのサブクラスの Xm Primitiveのサブクラスの XmLabelのサブクラスであり、XRT/tableウィジェットはXmManagerのサブクラスとなります。これは、XRT/tableウィジェットで定義されるリソースの他にXmManagerとCoreのリソースも設定できることを意味します。例えば、`XmNbackground`リソースを用いてデフォルトの背景色を設定することができます。

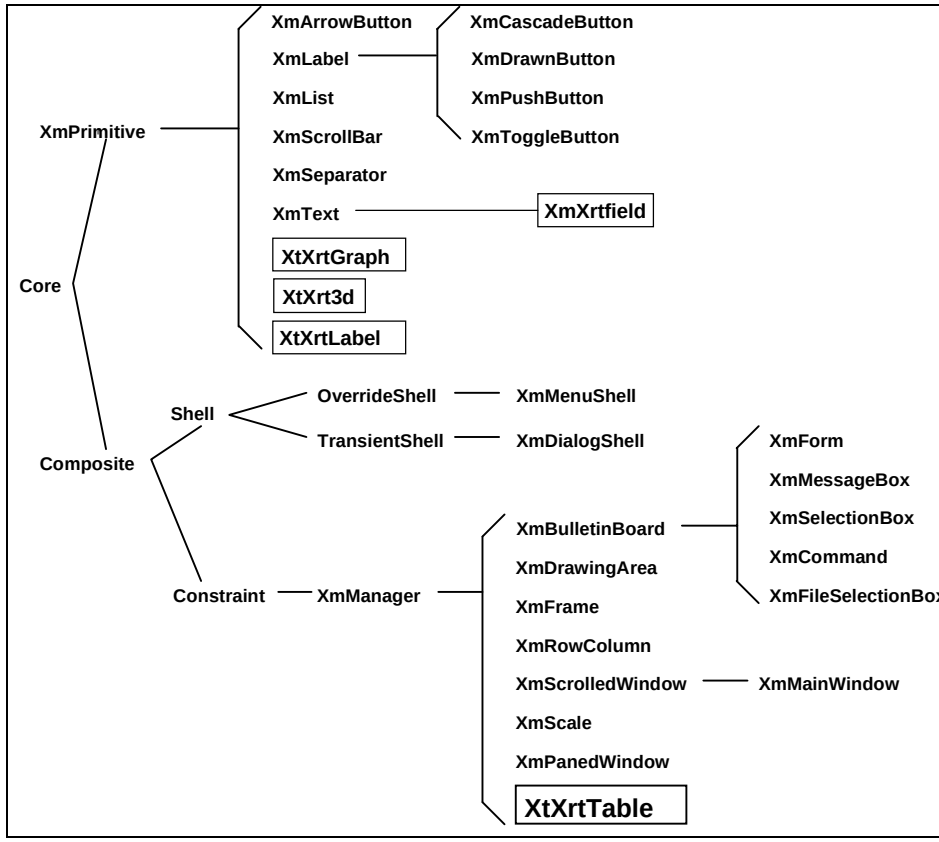


図7　Motifクラス階層内のXRTウィジェットの位置

2.2 テーブルの構成

次の図は、XRT/tableウィジェットの構成です。

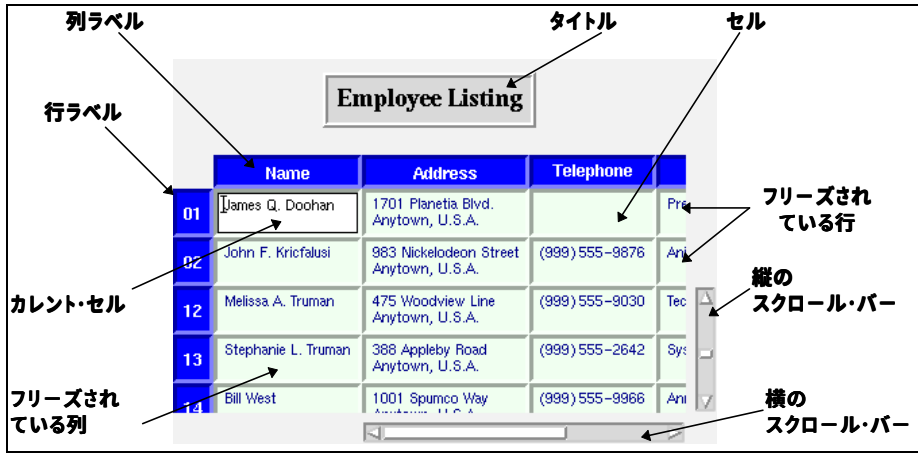


図8　テーブル・ウィジェットの構成

以下は、一般的なMotifプログラミングでは使用されないXRT/tableの用語です。

セル

テーブル・データを表示する独立した箱です。セルは、テキスト、ウィジェット、背景のpixmapを持つことができます。スクロールすることによりセルを*表示/非表示*にすることができます。セル値は、`XrtTblCellValues`構造体に記憶されます。表示されるセル全体は、*cell area(セル領域)*と呼ばれます。

ラベル

テーブルの最上行、または最左列にある編集不可のセルです。セルと同様、ラベルはテキストやウィジェット、背景にpixmapを表示することができます。但し、セルのようにビュー(表示領域)の外にスクロールすることができません。ラベル値は、セル値とは別に、行ラベル、列ラベルとして二つの`XrtTblXmStringList`構造体に記憶されます。

カレント・セル

現在、入力フォーカスを持つセルです。エンドユーザは、このセルの値を入力したり編集することができます。(編集不可に設定することもできます。)

2.3 ウィジェットの管理

XRT/tableは、幾つかの子ウィジェットを作成、管理します。標準のMotifのウィジェットである、一つのXmTextと二つのXmScrollBarの他に、セルやラベルの表示される領域を切り取る特別なウィジェットも作成、管理します。[1]

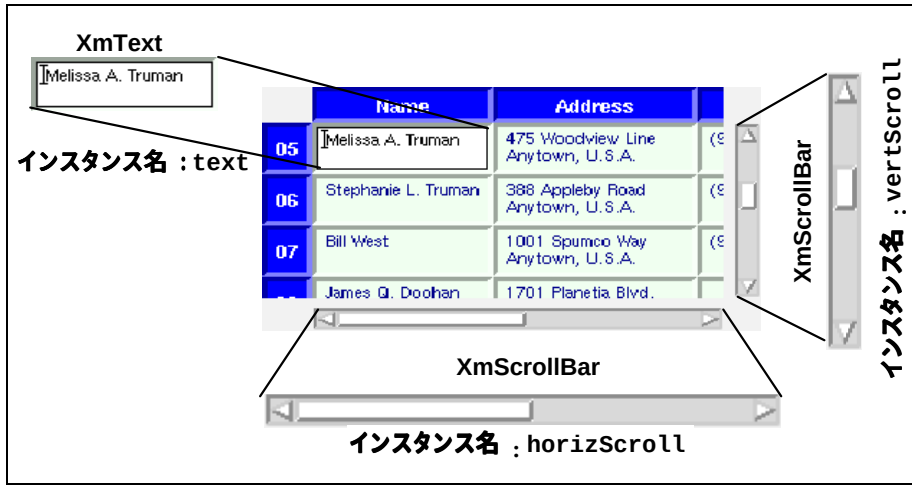


図9　XRT/tableに使用される標準のMotifウィジェット

XmTextウィジェット

このウィジェットはカレント・セルです。エンドユーザは、このセルの値を入力したり、編集することができます。ユーザがセルからセルにトラバースすると、XRT/tableは、XmTextウィジェットをアンマップし、次のセルへマップします。

XmScrollBarウィジェット

テーブルの行や列の数が、スクリーンで表示できる行や列の数より多い時に、これらのウィジェットは作成、表示されます。エンドユーザは、これらのウィジェットを使用して、テーブル全体をスクロールすることができます。

アプリケーションは、これらのウィジェットのリソースにアクセスし、セットすることができます。

[1] XmScrolledWindowウィジェットも同様に、作業するウィンドウ(ユーザがスクロールバーを利用して操作する領域)のビューを制限するためにclip window widgetを使用します。

2.4 テーブルのContext

contextは、行と列のインデックス(0 based)で構成されています。*current context*(カレントcontext)は、アプリケーションが属性を設定/取得できる、テーブルの部分を指定します。カレントcontextはXmNxrtTblContextで定義します。

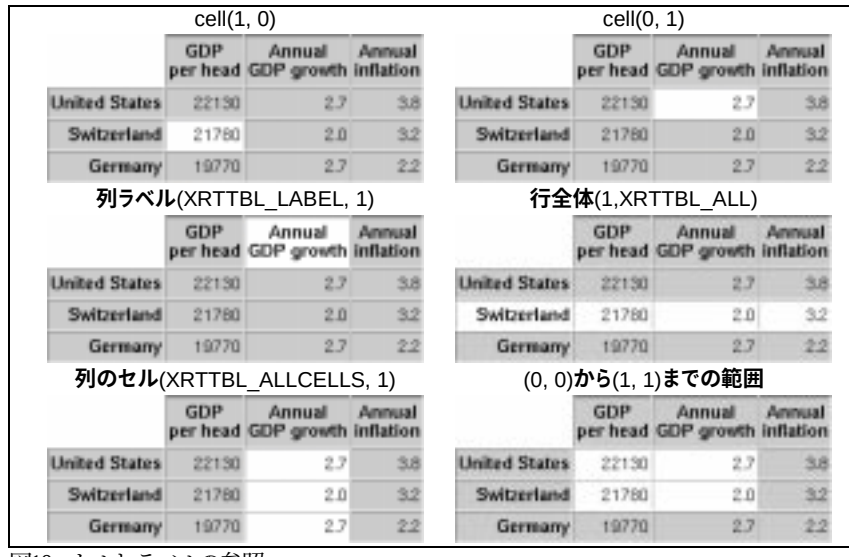


図10　セルとラベルの参照

以下は、参照できるテーブルcontextの種類です。

セル

行インデックスと列インデックス(ゼロから開始)で参照されます。例えば、2行目の2番めのセルの位置は、(1, 1)です。

行ラベル

行インデックスと定数XRTTBL_LABELで参照されます。例えば、2行目のラベルの位置は、(1, XRTTBL_LABEL)です。行ラベル全てを参照する場合は、(XRTTBL_ALL, XRTTBL_LABEL)です。

列ラベル

列インデックスと定数XRTTBL_LABELで参照されます。例えば、2列めのラベルの位置は、(XRTTBL_LABEL, 1)です。列ラベル全てを参照する場合は、(XRTTBL_LABEL, XRTTBL_ALL)です。

行または列の全てのセル

定数XRTTBL_ALLCELLSで参照されます。例えば、2行目の全てのセルを参照するには、(1, XRTTBL_ALLCELLS)と指定します。

行または列全体

定数XRTTBL_ALLで参照され、ラベルも含みます。例えば、2列め全体を参照するには、(XRTTBL_ALL, 1)と指定します。

テーブルの全てのセル

(XRTTBL_ALLCELLS, XRTTBL_ALLCELLS)で参照されます。

テーブル全体

(XRTTBL_ALL, XRTTBL_ALL)で参照されます。(ラベルも含まれます。)

範囲

範囲の、左上のセル/ラベルと右下のセル/ラベルの位置で参照されます。範囲とは、セルと(または)ラベルのブロックで、一つのcontextとして参照されます。

2.5 ContextとSeriesの参照

XRT/tableには、ContextとSeriesという、セル/ラベルの属性を指定する二つの効率的なメカニズムが提供されています。これらを利用して、テーブルの異なったレベルの範囲に属性を適用させることができます。

カレントcontextに作用するリソースを*Context*リソースと呼びます。全てのContextリソースには、対応する*Series*リソースがあります。(例:`XmNxrtTblBackgroundContext`と`XmNxrtTblBackgroundSeries`)

Seriesリソースは、テーブル全体に対する一連のリソース設定を指定します。SeriesとContextのどちらのリソースも、テーブル全体の属性を設定する同じ内部リストに影響を与えます。Contextリソースをセットすると、対応するSeriesリソースもアップデートします。ほとんどの場合、ContextリソースとSeriesリソースの使用には、パフォーマンスに大きな差はありません。[1]

Contextリソースは、アプリケーション・プログラムに使用されるように設計されています。一度、contextが設定されると、どのContextリソース設定も、そのcontextに適用されます。これを利用すると、テーブルの同じ部分に、複数の属性設定を簡単に行うことができます。Contextリソースは、XRT/tableの基本的なプログラミング・インターフェースですので、このマニュアルの解説や例では、Contextリソースを使用します。

Seriesリソースは、リソース・ファイルやfallbackリソースに使用されるように設計されています。seriesの各メンバには、contextが含まれています。リソース・ファイルでは、一つのリソースを幾つかのcontextに設定することができます。但し、seriesにはテーブル全体の設定全てが含まれているので、プログラミングや管理が難しくなります。

Contextリソースの設定

カレントcontextの指定には、`XmNxrtTblContext`または`XmNxrtTblRangeContext`を使用します。ほとんどの種類のcontextの設定には、`XmNxrtTblContext`を使用します。`XmNxrtTblRangeContext`を使用すると、範囲内の全てのセルにcontextを設定できます。これらのリソースの内一つを設定すると、もう一つが無視されます。

[1] 多くのセルまたはcontextに同じリソースを設定する場合は、Seriesリソースを使用した方が速いです。

ほとんどの種類のcontextの設定(XmNxrtTblContext)には、*XrtTblSetContext()*を使用します。

```
XtVaSetValues(table,
  XmNxrtTblContext, XrtTblSetContext(XRTTBL_LABEL,
                    XRTTBL_ALL),
  XmNxrtTblBackgroundContext, "grey",
  XmNxrtTblForegroundContext, "black",
  NULL);
```

範囲にcontextを設定(XmNxrtTblRangeContext)する場合には、*XrtTblSetRange()*を使用してください。

```
XtVaSetValues(table,
  XmNxrtTblRangeContext,    XrtTblSetRange(1,1,5,5),
  XmNxrtTblBackgroundContext,   "grey",
  XmNxrtTblForegroundContext,   "black",
  NULL);
```

Contextリソースの取得

カレントcontextではないcontextのContextリソース値を取得するには、*XtVaGetValues()*を呼び出す前に、そのcontextをセットしてください。

```
XtVaSetValues(table,
  XmNxrtTblContext,         XrtTblSetContext(2,2),
  NULL);
XtVaGetValues(table,
  XmNxrtTblBackgroundContext, &bg_color,
  NULL);
```

Contextリソースのリセット

Contextリソースをデフォルト値に戻すには、XRTTBL_REMOVEを使用します。

```
XtVaSetValues(table,
  XmNxrtTblRangeContext,    XrtTblSetRange(1,1,5,5),
  XmNxrtTblBackgroundContext,   "grey",
  NULL);
XtVaSetValues(table,
  XmNxrtTblRangeContext,    XrtTblSetRange(1,1,5,5),
  XmNxrtTblBackgroundContext,   XRTTBL_REMOVE,
  NULL);
```

*XtVaSetValues()*の二番目のコールが、最初の設定を元に戻しています。

他のcontext設定は、XRTTBL_REMOVEに影響されることはありません。

```
XtVaSetValues(table,
  XmNxrtTblRangeContext,    XrtTblSetContext(2,2),
  XmNxrtTblBackgroundContext,   "green",
  NULL);
XtVaSetValues(table,
  XmNxrtTblRangeContext,    XrtTblSetRange(1,1,5,5),
  XmNxrtTblBackgroundContext,   XRTTBL_REMOVE,
  NULL);
```

この場合、セル(2,2)の背景色は"green"のままです。

セットされていないcontextにXRTTBL_REMOVEが指定された場合は、無視されます。

Seriesリソースの設定

Seriesリソースは、一連のリソース設定のリストです。リソース・ファイルで、各メンバは、括弧で囲まれたcontextと値で構成されています。次のコードは、リソース・ファイルの一部で、背景色を幾つかのcontextにセットしています。

```
*.xrtTblBackgroundSeries: (ALL ALL gray) (2 2 blue) \
(2 2 red) (ALL 4-7 yellow) (ALL 4 red) \
(LABEL ALL yellow) (LABEL 2 red) (LABEL 5 black) \
(ALL LABEL yellow) (1 LABEL red)
```

Seriesの設定は、読み込まれる順に処理されます。始めに、(LABEL ALL yellow)は全ての列ラベルの背景色を黄色にセットしますが、次に続く(LABEL 2 red)と(LABEL 5 black)が優先されます。リソース・ファイルでのリソース指定は、Appendix Gに詳細が解説されています。

Seriesリソースのプログラミング

Seriesリソースのプログラミングは複雑で、あまり使用されません。詳細は、6.6をご覧ください。

2.6 ポインタのリソース

幾つかのXRT/tableのリソースは、*XtGetValues()*コールで使用されると、構造体かストリングのポインタを返します。これはウィジェットの内部ストーレッジのポインタですので、このポインタを使用してメモリ内のデータを変更したり、このポインタを変更または解放しないでください。ポインタが指すものは全て"read only"で"temporary"です。

XRT/tableは、ポインタ・リソースで渡された全てのデータの内部コピーを作成します。

XRT/tableの複写の手続き(*XrtTblSeriesCopy()*等)と、*XtNewString()*のようなXtの手続きは、データのコピー作成に使用できます。XRT/tableの解放の手続き(*XrtTblSeriesDestroy()*等)と、*XtFree()*のようなXtの手続きは、これらのコピーの解放に使用できます。XRT/tableの手続きのリストは、Appendix Bを参照してください。

次の例は、返されたポインタをコピーしています。

```
XmString value1;

XtVaSetValues(table,
  XmNxrtTblContext, XrtTblSetContext(0,0),
  NULL);
XtVaGetValues(table,
  XmNxrtTblCellValueContext, &value1,
  NULL);
value1 = XmStringCopy(value1);
```

2.7 XmStringリソースとMotif 2.xの制限

Motif 1.2を使用している場合、XmStringに対してStringを使用することができます。(これは、Motif 2.xでは許可されていません。) XmStringsは、次のような場合に使用されます。

- XmNxrtTblCellValueContextとXmNxrtTblLabelContextリソースの値
- XmNxrtTblColumnLabelsとXmNxrtTblRowLabelsリソースをセットするリスト
- XmNxrtTblCellValueCallbackとXmNxrtTblLabelCreateCallbackリソースによって指定されるコールバック
- *XrtTblAddColumns()*と*XrtTblAddRows()*に渡される配列

Stringが指定されると、XRT/tableは自動的にこれをXmStringに変換します。

XmNxrtTblXmStringSetModeリソースを使用して、このデフォルトで行われる変換を置き換えることができます。通常はXmStringsを使用するが、Stringsも使用できるリソースや手続きは、XmStringsだけ、または、Stringsだけを使用するようにセットできます。

XmNxrtTblXmStringGetModeリソースは、上記のリソース値が取り出される時のフォーマットをコントロールします。通常、取り出された値はXmStringsですが、Stringsにセットすることもできます。

2.8 色

セル、ラベルまたは選択した部分に前景色や背景色を指定することができます。何も指定されていなければ、XRT/tableは、親のXmNforegroundとXmNbackgroundからデフォルトの色を継承します。

色の指定

色は、Pixel色値かまたはストリングで指定します。ストリングは、色名かまたはRGBの16進数です。

```
char * my_named_color = "LemonChiffon";
char * my_rgb_color = "#228B22";
```

色名で指定する場合は、Xサーバのカラー・データベースから選択してください。通常、Unixシステムでは、このファイルは */usr/lib/X11/rgb.txt*です。

色名の代わりにRGB値を使用する場合は、ストリングは、#で始まる、R(赤)、G(緑)、B(青)それぞれ二桁の16進数です。上記の例では、my_rgb_colorは、赤が0x22、緑が0x8B、青が0x22となります。[1]

[1] 色の指定の構文は、この通りでなくても良く、Xlibの手続き*XParseColor()*がサポートできるストリングならどれでも使用することができます。

XRT/tableで作成されたアプリケーションをモノクロのサーバで使用する場合は、次のように表示されます。

- 背景色では、黒以外の色は白
- 他の白以外の色は黒

XRT/tableウィジェットが作成された後は、アプリケーションでカラーマップの変更やアップデートはしないでください。色の変更は、ウィジェットのリソース(`XmNxrtTblBackgroundContext`等)で行います。

カラーマップのフラッシュ

Xサーバのカラーマップは共有リソースです。ある状況においては、全てのアプリケーションで表示中の色の数が、使用可能なカラーマップのエントリ数より多いことがあります。この場合、画面上でユーザがマウスを動かした時、その色が他の色に変更(フラッシュ)してしまいます。

サーバが*static*な色だけを提供する場合は、このカラー・フラッシュを防ぐことができます。アプリケーションには、指定通りの色ではありませんが、ウィンドウのカラーマップ内の一番近い色が選択されます。

しかし、多くのアプリケーションではダイナミックな色を必要とします。この場合、各アプリケーションには指定通りの色を使用できますが[1]、サーバの使用可能な色の数やアプリケーションの数と状態によってカラーマップ・フラッシュが起こります。

ダイナミックなカラーマップで実行中、XRT/tableは、アプリケーションの要求に従って自動的にサーバのカラー・セルの割り当てまたは、割り当て解除を行い、カラーマップ・リソースへの要求を最小限にします。XRT/tableのアプリケーションが可能な範囲以外の色を要求したときは、代わりに白か黒で表示されます。

テーブルのセルまたはラベルにpixmapがある時は、エンドユーザのサーバが表示できない色を使用しないように注意してください。

2.9 警告とエラーメッセージ

`XmNxrtTblDebug`を`True`にセットすると、Xt Error Databaseから警告またはエラー・メッセージを*stderr*に出力します。例えば、テーブルの行数より多い行数のフリーズを指定すると、エラー・メッセージが現れます。

[1] ハードウェアの表示能力の範囲内

2.10 リソース･ファイルの使用

リソースの値を設定するには、Cコードより、リソース･ファイルを使用した方がよい場合があります。リソース･ファイルには次の利点があります。

- スピード ― コンパイルしないので、変更が瞬時に確認できます。
- 柔軟性 ― エンドユーザがニーズに合わせてリソース値を変更できます。

XRT/tableのほとんどのリソース(セル値も含む)は、リソース･ファイルで設定できます。但し、多くのリソースは、エンドユーザが変更を加えたりするには複雑すぎるかもしれません。アプリケーションの全てのウィジェットのリソース･ファイルは、組み合わせてアプリケーションのデフォルト･ファイルを作成することができます。

リソース･ファイルとその使用法の詳しい説明は、Xt Intrinsicsの資料等をご覧ください。尚、Appendix Gには、リソース･ファイルでの構文の解説やサンプルのファイルがあります。

3

XRT/tableのプログラミング

リソース ■ 行と列の数 ■ 表示されている行と列
フリーズされた行または列 ■ 行と列のラベル ■ テーブルのタイトル
行の高さと列幅 ■ セルとラベルのマージン ■ 前景色、背景色と影の色
セルとラベルのテキストの行揃え ■ セルとラベルのフォント
罫線の種類と場所 ■ 背景のPixmap ■ テキストのclippingとoverflow
セルとラベルのspanning ■ セルとラベルのフラッシュ
ユーザのデータ ■ テーブルの印刷

3.1 リソース

この章では、XRT/tableのリソースについて解説します。下の表は、リソースの一覧ですが、全て覚えていただく必要はありません。ほとんどのリソースは、デフォルトの設定のままにしておくことができます。

Appendix AにはXRT/tableの全てのリソースの一覧がありますので参照してください。

Resource	Type
XmNxrtTblAllowCellResize	XrtTblResize
XmNxrtTblAllowResize	XrtTblResize
XmNxrtTblCellValueFunc	XrtTblXmStringProc
XmNxrtTblCellValues	XrtTblCellValues *
XmNxrtTblColumnLabelOffset	int
XmNxrtTblCursorCorner	Cursor
XmNxrtTblCursorHorizontal	Cursor
XmNxrtTblCursorNontraversable	Cursor
XmNxrtTblCursorTraversable	Cursor
XmNxrtTblCursorVertical	Cursor
XmNxrtTblDataTypeList	XrtTblDataTypeRoutine *
XmNxrtTblDisplayClipArrows	Boolean
XmNxrtTblDoubleBuffer	Boolean

次ページに続く

Resource	Type
XmNxrtTblExportDelimiter	String
XmNxrtTblExportTargets	XrtTblImportExportStruct **
XmNxrtTblFrameBorderType	XrtTblBorderType
XmNxrtTblFrameShadowThickness	Dimension
XmNxrtTblFrozenColumnPlacement	XrtTblColumnPlacement
XmNxrtTblFrozenColumns	int
XmNxrtTblFrozenRowPlacement	XrtTblRowPlacement
XmNxrtTblFrozenRows	int
XmNxrtTblImportDelimiter	String
XmNxrtTblImportTargets	XrtTblImportExportStruct **
XmNxrtTblLeftColumn	int
XmNxrtTblMarginHeight	Dimension
XmNxrtTblMarginWidth	Dimension
XmNxrtTblMinCellVisibility	int
XmNxrtTblMinFlexibleHeight	Dimension
XmNxrtTblMinFlexibleWidth	Dimension
XmNxrtTblMode	XrtTblMode
XmNxrtTblNumColumns	int
XmNxrtTblNumRows	int
XmNxrtTblPSFontMapList	XrtTblPSFontMap *
XmNxrtTblRepeatBackgroundColors	String *
XmNxrtTblRepeatForegroundColors	String *
XmNxrtTblRowLabelOffset	int
XmNxrtTblSelectedBackground	String
XmNxrtTblSelectedCellList	XrtTblRange **
XmNxrtTblSelectedForeground	String
XmNxrtTblSelectionPolicy	XrtTblSelectionPolicy
XmNxrtTblSetTextCursorPos	Boolean
XmNxrtTblShadowThickness	Dimension
XmNxrtTblSpanList	XrtTblRange **
XmNxrtTblTextInitSelect	Boolean
XmNxrtTblTopRow	int
XmNxrtTblTrackCursor	Boolean
XmNxrtTblValidatePolicy	XrtTblValidatePolicy
XmNxrtTblVisibleColumns	int
XmNxrtTblVisibleRows	int
XmNxrtTblWidgetClassList	XrtTblWidgetClass *
XmNxrtTblXmStringGetMode	XrtTblXmStringGet
XmNxrtTblXmStringSetMode	XrtTblXmStringSet

図11　テーブル全体のリソース

Resource	Type
XmNxrtTblAlignmentContext	XrtTblAlignment
XmNxrtTblArgListContext	ArgList
XmNxrtTblBackgroundContext	String
XmNxrtTblBackgroundPixelContext	Pixel
XmNxrtTblBorderSidesContext	XrtTblBorderSides
XmNxrtTblBorderTypeContext	XrtTblBorderType
XmNxrtTblBottomShadowContext	Pixel
XmNxrtTblCharHeightContext	int
XmNxrtTblCharWidthContext	int
XmNxrtTblClipPixmapContext	Pixmap
XmNxrtTblContext	XrtTblContext *
XmNxrtTblDataTypeContext	XrtTblDataType
XmNxrtTblFieldArgListContext	ArgList
XmNxrtTblFontListContext	XmFontList
XmNxrtTblForegroundContext	String
XmNxrtTblForegroundPixelContext	Pixel
XmNxrtTblOverflowContext	XrtTblOverflow
XmNxrtTblPixelHeightContext	int
XmNxrtTblPixelWidthContext	int
XmNxrtTblPixmapContext	Pixmap
XmNxrtTblPixmapLayoutContext	XrtTblPixmapLayout
XmNxrtTblPixmapShadowContext	Boolean
XmNxrtTblPSFontListContext	String
XmNxrtTblRangeContext	XrtTblRange *
XmNxrtTblSensitiveContext	Boolean
XmNxrtTblTopShadowContext	Pixel
XmNxrtTblUserDataContext	XtPointer
XmNxrtTblWidgetContext	Widget
XmNxrtTblWidgetShadowContext	Boolean
XmNxrtTblWordWrapContext	Boolean

図12　セル/ラベルのContextリソース

Resource	Type
XmNxrtTblCaseContext	XrtTblCase
XmNxrtTblCellValueContext	XmString
XmNxrtTblCurrencyPickListContext	double *
XmNxrtTblDatePickListContext	struct tm **
XmNxrtTblEditableContext	Boolean
XmNxrtTblFloatPickListContext	double *
XmNxrtTblIntPickListContext	long *
XmNxrtTblMaxLengthContext	int
XmNxrtTblMultilineContext	Boolean
XmNxrtTblStringPickListContext	String
XmNxrtTblTraversableContext	Boolean

図13　セルだけのContextリソース

Resource	Type
XmNxrtTblColumnLabelPlacement	XrtTblRowPlacement
XmNxrtTblColumnLabels	XrtTblXmStringList *
XmNxrtTblLabelContext	XmString
XmNxrtTblLabelCreateFunc	XrtTblXmStringProc
XmNxrtTblRowLabelPlacement	XrtTblColumnPlacement
XmNxrtTblRowLabels	XrtTblXmStringList *

図14　ラベルだけのContextリソース

Resource	Type
XmNxrtTblAlignmentSeries	XrtTblSeries *
XmNxrtTblArgListSeries	XrtTblSeries *
XmNxrtTblBackgroundSeries	XrtTblSeries *
XmNxrtTblBorderSidesSeries	XrtTblSeries *
XmNxrtTblBorderTypeSeries	XrtTblSeries *
XmNxrtTblCharHeightSeries	XrtTblSeries *
XmNxrtTblCharWidthSeries	XrtTblSeries *
XmNxrtTblClipPixmapSeries	XrtTblSeries *
XmNxrtTblDataTypeSeries	XrtTblSeries *
XmNxrtTblFieldArgListSeries	XrtTblSeries *
XmNxrtTblFontListSeries	XrtTblSeries *
XmNxrtTblForegroundSeries	XrtTblSeries *
XmNxrtTblOverflowSeries	XrtTblSeries *
XmNxrtTblPixelHeightSeries	XrtTblSeries *
XmNxrtTblPixelWidthSeries	XrtTblSeries *
XmNxrtTblPixmapSeries	XrtTblSeries *
XmNxrtTblPixmapLayoutSeries	XrtTblSeries *
XmNxrtTblPixmapShadowSeries	XrtTblSeries *
XmNxrtTblPSFontListSeries	XrtTblSeries *
XmNxrtTblSensitiveSeries	XrtTblSeries
XmNxrtTblUserDataSeries	XrtTblSeries *
XmNxrtTblWidgetSeries	XrtTblSeries *
XmNxrtTblWidgetShadowSeries	XrtTblSeries *
XmNxrtTblWordWrapSeries	XrtTblSeries *

図15　セル/ラベルのSeriesリソース

Resource	Type
XmNxrtTblCaseSeries	XrtTblSeries *
XmNxrtTblCurrencyPickListSeries	XrtTblSeries *
XmNxrtTblDatePickListSeries	XrtTblSeries *
XmNxrtTblEditableSeries	XrtTblSeries *
XmNxrtTblFloatPickListSeries	XrtTblSeries *
XmNxrtTblIntPickListSeries	XrtTblSeries *
XmNxrtTblMaxLengthSeries	XrtTblSeries *
XmNxrtTblMultilineSeries	XrtTblSeries *
XmNxrtTblStringPickListSeries	XrtTblSeries *
XmNxrtTblTraversableSeries	XrtTblSeries *

図16　セルだけのSeriesのリソース

Resource	Type
XmNxrtTblDisplayHorizScrollBar	XrtTblDispScrollBar
XmNxrtTblDisplayVertScrollBar	XrtTblDispScrollBar
XmNxrtTblHorizScrollBarLeftAttachment	XrtTblAttachment
XmNxrtTblHorizScrollBarPlacement	XrtTblRowPlacement
XmNxrtTblHorizScrollBarPosition	XrtTblSBPosition
XmNxrtTblHorizScrollBarRightAttachment	XrtTblAttachment
XmNxrtTblJumpScroll	XrtTblJumpScroll
XmNxrtTblPreviewScrolling	XrtTblPreviewScroll
XmNxrtTblSpace	Dimension
XmNxrtTblVertScrollBarBottomAttachment	XrtTblAttachment
XmNxrtTblVertScrollBarPlacement	XrtTblColumnPlacement
XmNxrtTblVertScrollBarPosition	XrtTblSBPosition
XmNxrtTblVertScrollBarTopAttachment	XrtTblAttachment

図17　スクロールバーのリソース

Resource	Type
XmNxrtTblClipChildren	Widget *
XmNxrtTblClipChildrenPersistent	Boolean
XmNxrtTblHorizontalScrollBar	Widget
XmNxrtTblText	Widget
XmNxrtTblVerticalScrollBar	Widget

図18　子ウィジェットのリソース

Resource	Type
XmNxrtTblBottomOffset	int
XmNxrtTblLeftOffset	int
XmNxrtTblLocation	XrtTblLocation
XmNxrtTblResizeToCell	Boolean
XmNxrtTblRightOffset	int
XmNxrtTblTopOffset	int
XmNxrtTblWidgetLocation	XrtTblContext *

図19　Constraintリソース

Resource	Type
XmNxrtTblCellValueCallback	XtCallbackList
XmNxrtTblContextValueCallback	XtCallbackList
XmNxrtTblCreateWidgetCallback	XtCallbackList
XmNxrtTblDisplayWidgetCallback	XtCallbackList
XmNxrtTblEnterCellCallback	XtCallbackList
XmNxrtTblExportCallback	XtCallbackList
XmNxrtTblExposeCallback	XtCallbackList
XmNxrtTblImportCallback	XtCallbackList
XmNxrtTblLabelCreateCallback	XtCallbackList
XmNxrtTblPrintCallback	XtCallbackList
XmNxrtTblPrintWidgetCallback	XtCallbackList
XmNxrtTblResizeCallback	XtCallbackList
XmNxrtTblResizeCellCallback	XtCallbackList
XmNxrtTblScrollCallback	XtCallbackList
XmNxrtTblSelectCallback	XtCallbackList
XmNxrtTblSortCallback	XtCallbackList
XmNxrtTblTrackCursorCallback	XtCallbackList
XmNxrtTblTraverseCellCallback	XtCallbackList
XmNxrtTblValidateCellCallback	XtCallbackList
XmNxrtTblWidgetSetValueCallback	XtCallbackList

図20　コールバックのリソース

Resource	Type
XmNxrtTblDebug	Boolean
XmNxrtTblRepaint	XrtTblRepaint

図21　その他のリソース

3.2 行と列の数

XmNxrtTblNumRowsとXmNxrtTblNumColumnsリソースは、テーブルの行と列の数を設定します。これらは、表示されるテーブルのサイズを定義しますので、ほとんどのテーブルでは設定する必要があります。

これらのリソースは、セル値を記憶しているXrtTblCellValues構造体から独立しています。XmNxrtTblNumRowsとXmNxrtTblNumColumnsは、テーブルのセル値の構造体の行や列の数より小さくても大きくても構いません。(図22参照)

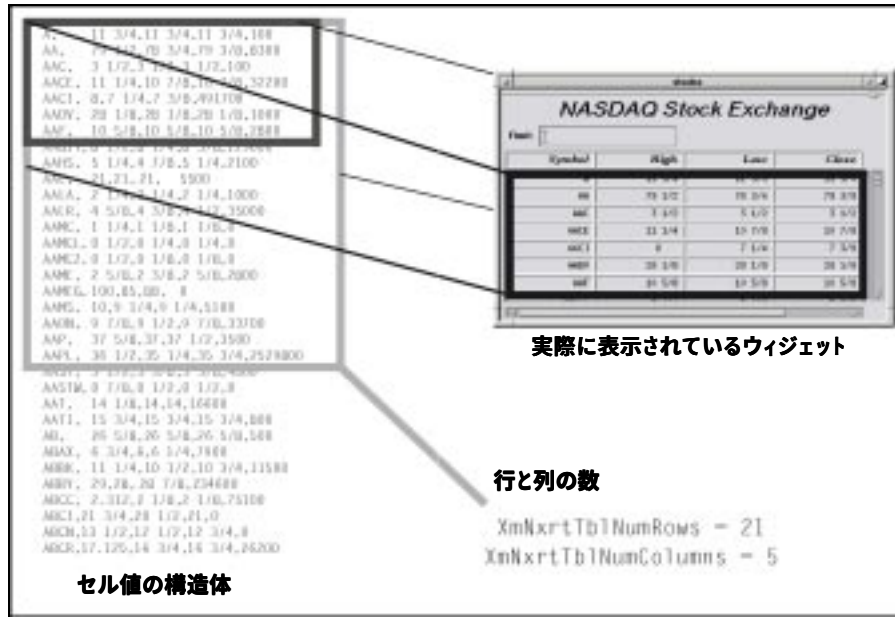


図22 XmNxrtTblNumRowsを使用してスクロール可能な行数を制限する

例えば、XmNxrtTblNumRowsの値を減らすことにより、テーブルの下部から行を一時的に削除することができます。一時的に削除されている行のセル値は、XrtTblCellValues構造体に存在したままです。XmNxrtTblNumRowsの値を増やせば、行は元に戻ります。

行/列の数は、フリーズされる行/列の数より大きくなければなりません。

3.3 表示されている行と列

現在のウィンドウ内に表示される行と列の数は、XmNxrtTblVisibleRowsとXmNxrtTblVisibleColumnsリソースによって指定します。[1]

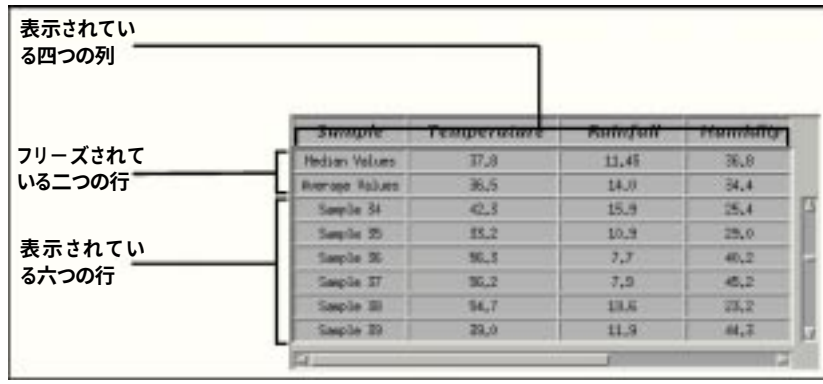


図23　表示/フリーズされている行と列

これらのリソースは、他のリソースが変更された時や、ユーザがウィジェットとインタラクションを行った時は、ダイナミックにアップデートします。図23にあるように、これらのリソースが指定する「表示される行/列」は、フリーズされている行/列を含みません。

テーブルに表示させる行数または列数を指定する時は、これらのリソースを設定します。表示される行や列が設定されると、それらを表示するようにウィジェットはリサイズし、XmNxrtTblAllowResizeでセットされたリサイズの方法が置き換えられます。

表示させる行や列の数は、対応するXmNxrtTblNumRowsやXmNxrtTblNumColumnsの値より、大きく設定することはできません。

テーブル全体の表示

テーブル全体を表示するには、XmNxrtTblVisibleRowsとXmNxrtTblVisibleColumnsを0にセットします。どちらかのリソースを0に設定することにより、テーブルの全ての行/列が表示されるようにリサイズする、特別なフラグがセットされます。このフラグがセットされている時は、行/列の数が変更されるとテーブルは常にリサイズします。

3.4 フリーズされた行または列

XmNxrtTblFrozenRowsとXmNxrtTblFrozenColumnsを使用して、スクロールできない行や列を指定することができます。XmNxrtTblFrozenRowsリソースは、ウィンドウの上部または下部に固定される、スクロールできない行を指定します。XmNxrtTblFrozenColumnsは、ウィンドウの左または右の部分に固定される、スクロールできない列を指定します。

フリーズされている行/列は、必ずテーブルの開始ポイントから始まります。これらの行/列は、編集またはトラバースが(不可に設定されていない限り)可能です。

[1] 部分的に表示されている行/列も、これらのリソース値に含まれています。

行または列をフリーズして、テーブルがスクロールされた時に大切な情報が常に表示されるようにしたり(例:テーブルの最下部に合計額を固定する）、もう一つのラベルの代わりに使用することができます。

行または列をフリーズする

行/列をフリーズするには、*XrtTblMoveRows()*または*XrtTblMoveColumns()*を呼び出し、フリーズする行/列をテーブルの開始ポイントに移動します。次にXmNxrtTblFrozenRowsまたはXmNxrtTblFrozenColumnsの値を開始ポイントからフリーズしたい行数または列数に設定します。

次の例は、行5をテーブルの開始ポイントに移動し、フリーズします。

```
XrtTblMoveRows(table, 5, 1, 0, True);
XtVaSetValues(table,
  XmNxrtTblFrozenRows, 1,
  XmNxrtTblFrozenColumns,   0,
  NULL);
```

フリーズされた行/列の位置

フリーズされる行はウィジェットの表示の上部か下部、フリーズされる列は右か左に配置することができます。この位置は、XrtTblCellValues構造体の行/列の位置には影響しません。

フリーズされている行の位置を変更するには、XmNxrtTblFrozenRowPlacementをXRTTBL_PLACE_TOPまたはXRTTBL_PLACE_BOTTOMにセットします。

フリーズされている列の位置を変更するには、XmNxrtTblFrozenColumnPlacementをXRTTBL_PLACE_LEFTまたはXRTTBL_PLACE_RIGHTにセットします。



図24　ウィンドウの上部と下部に配置されたフリーズされた行

3.5　行と列のラベル

XmNxrtTblColumnLabelsで指されているXrtTblXmStringList構造体は、列ラベルを指定します。XmNxrtTblRowLabelsで指されているXrtTblXmStringList構造体は行ラベルを指定します。行/列ラベルは、セル値とは別に記憶されます。

ユーザは、インタラクティブにラベルを編集することはできません。

3.5.1 ラベル値の設定と取得

全てのラベルの設定

行または列の全てのラベルを指定する時は、XmNxrtTblColumnLabelsまたはXmNxrtTblRowLabelsをラベルの新しいリストにセットします。(下の例参照)

```
XrtTblXmStringList labels;
String label_list[] = { "Row 0", "Row 1", "Row 2" };

labels.length = XtNumber(label_list);
labels.list = (XmString *)label_list;
XtVaSetValues(table,
  XmNxrtTblRowLabels, &labels,
  NULL);
```

デフォルトでは、XmNxrtTblColumnLabelsとXmNxrtTblRowLabelsには、XmStringsまたはStringのリストどちらも使用できます。XmNxrtTblXmStringSetModeリソースを使用して、どちらかのリストだけに制限する方法は、4.3.1に記載されています。[1]

各ラベルの設定

各ラベルを設定するには、ラベルのcontextをセットし、次にXmNxrtTblLabelContextリソースをセットします。(下の例参照)

```
String labels[] = { "Name", "Address", "Phone Number",
                    "Position", "Salary", NULL };
for (i=0; labels[i]; i++)
  XtVaSetValues(table,
    XmNxrtTblContext, XrtTblSetContext(XRTTBL_LABEL,
                      i),
    XmNxrtTblLabelContext, labels[i],
    NULL);
```

ラベルの取得

ラベル値を取り出すには、次の二つの方法があります。

1 contextが一つのラベルの時は、XmNxrtTblLabelContextを使用してラベルの値を取り出します。

```
XmString value1;
XtVaSetValues(table,
  XmNxrtTblContext, XrtTblSetContext(XRTTBL_LABEL,0),
  NULL);
XtVaGetValues(table,
  XmNxrtTblLabelContext, &value1,
  NULL);
value1 = XmStringCopy(value1);
```

2 ある位置の行または列ラベルの値は、LABEL_VALUEマクロを使用して取り出します。

[1] Motif 2.xにリンクする場合は、XmNxrtTblColumnLabelsとXmNxrtTblRowLabelsは、XmStringとStringの両方を使用できません。XmNxrtTblXmStringSetModeを使用して、使用する型を指定してください。

ラベル型の指定

XmNxrtTblXmStringGetModeを使用して、セル/ラベルから取り出される値を、String型またはXmString型に指定できます。XmNxrtTblXmStringSetModeを使用すると、アプリケーションによってセットされるラベルをStringまたはXmStringに指定できます。尚、これらのリソースは、4.3.1に解説されています。

3.5.2 ラベルの位置と間隔

ラベルの位置

スクリーン上の行/列のラベルの位置は、XmNxrtTblColumnLabelPlacementとXmNxrtTblRowLabelPlacementで指定します。

列ラベルは、テーブルの上部(XRTTBL_PLACE_TOP)か下部(XRTTBL_PLACE_BOTTOM)、行ラベルは、テーブルの左(XRTTBL_PLACE_LEFT)か右(XRTTBL_PLACE_RIGHT)に表示できます。

Magnetic Porcelain Whiteboard 48"x72"	940-874	100	1
Elbattide Adjustable Footrest	940-538	50	2
Premium Leather Business Card Holder (holds 120)	940-249	250	3
BIC-mate Premium Ballpoint Pens (pkg. 10)	940-859	1000	4
Foldback Binder Clips (pkg. 20)	940-903	2500	5
Item Description	SKU #	Qty	

図25 デフォルトの行と列ラベルの位置を逆にする

ラベルの間隔

XmNxrtTblRowLabelOffsetリソースは行ラベルとセル領域、XmNxrtTblColumnLabelOffsetは列ラベルとセル領域の間隔をそれぞれpixel値で指定します。負の値が指定されると、ラベルはセル領域に重なって表示されます。

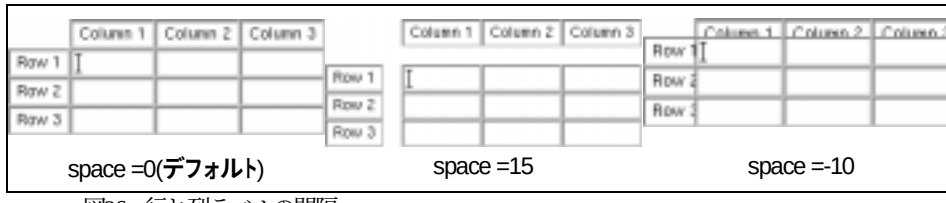


図26 行と列ラベルの間隔

3.5.3 ラベル作成のコールバック

LabelCreateコールバックを使用してもラベル値をセットすることができます。若干の作業が必要になりますが、大きなテーブルを扱う場合は、良いパフォーマンスが得られます。

XmNxrtTblLabelCreateCallbackリソースは、空白のラベル[1]が一回目に表示される際、XRT/tableが実行時に呼び出すコールバック・ルーチンを指定します。尚、ラベル値の追加、削除、移動、ソートは、アプリケーションの役割です。

作成したコールバックは、返されたXmString(XRT/tableによって解放)にメモリを割り当てたり、または静的データのポインタを渡すようにセットできます。このコールバックには、次の構造体が渡されます。

```
typedef struct {
  int reason; // Read-only
  XEvent  *event; // Read-only
  int row,     column; // Read-only
  XmString     string;
  Boolean store;
  Boolean free_string;
} XrtTblLabelCreateCallbackStruct;
```

デフォルトでは、*string*は、XmStringまたはStringのどちらかです[2] 。*string*をどちらかに制限する場合は、XmNxrtTblXmStringSetModeリソースをセットします。

次の例は、簡単なLabelCreateコールバックです。

```
static XmString
labelcreateCB(Widget table, XtPointer data,
  XrtTblLabelCreateCallbackStruct *cbs)
{
  static char buf[20];

  if (IS_ROWLABEL(cbs->row, cbs->column))
    /* Row Label */
    sprintf(buf, "%d", cbs->row+1);
  else /* Column Label */
    sprintf(buf, "%d", cbs->column+1);
  cbs->string = buf;
  cbs->free_string = False;
}
```

この例では、ラベル値の記憶に静的バッファを使用し、XrtTblLabelCreateCallbackStructの*free_string*メンバをセットして、ラベルの使用が終了してもXRT/tableにラベルを解放しないように伝えています。

LabelCreateコールバック・リストにこのコールバックを追加するには、*XrtTblAddCallback()*を呼び出します。

```
XrtTblAddCallback(table, XmNxrtTblLabelCreateCallback,
  labelcreateCB, NULL);
```

---

[1] XrtTblXmStringListのエレメントがNULL値のラベルは空白になります。

[2] Motif 2.xにリンクする場合、*string*はXmStringとStringの両方を使用できません。XmNxrtTblXmStringSetModeを使用して、どちらかの型に指定してください。

*XrtTblAddCallback()*を呼び出すことにより、このコールバックの存在がXRT/tableによって認識されます。

ラベル作成の構造体の*store*メンバは、ストリングが内部的に記憶されるかどうかを指定します。*store*が`True`であれば、値は記憶されコールバックはもう呼び出されません。デフォルトでは、*store*は`False`で、値は記憶されません。

XRT/tableには、スプレッドシート形式の行と列ラベルを作成する手続き、*XrtTblCreateSpreadsheetLabel()*があります。

3.6 テーブルのタイトル

アプリケーションは、複数のタイトルをテーブルのどの側にでも置くことができます。タイトルは、テーブルのセルとラベルの外に位置するウィジェットです。テーブルに一度追加されると、XRT/tableがconstraintリソースを使用して、タイトルのサイズや位置をコントロールします。

タイトルにはどのウィジェットも使用できますが、XRT/tableには、テーブル用に新しいラベル・ウィジェットが提供されています。XrtLabelは、標準のMotif XmLabelウィジェットに似ていますが、その他に、影付きの枠線やラベルの内容の回転、PostScript印刷(タイトルとして使用された場合)をサポートしています。XrtLabelの詳細は、Appendix Iにあります。

タイトルは、テーブルのヘッダーやフッター、特定なセルの注釈等、多くの用途に使用できます。

タイトルの追加

テーブルにタイトルを追加するには、テーブルの子としてウィジェットを作成します。次の例は、XrtLabelウィジェットをテーブルのタイトルとして作成しています。

```
title = XtVaCreateWidget("title",
  xtXrtLabelWidgetClass, table,
  XmNlabelString, "A Table Title",
  XmNxrtTblLocation, XRTTBL_LOCATION_TOP,
  NULL);
```

タイトル・ウィジェットは管理されない(unmanaged)状態で作成し、テーブル・ウィジェットに管理されるようにしてください。

タイトルの位置

XRT/tableのconstraintリソースは、各タイトルの位置とサイズをコントロールします。constraintリソースは、テーブルではなく、タイトル・ウィジェット自身にセットしてください。

`XmNxrtTblLocation`は、タイトルをテーブルのどの側に置くか指定します。図27は、タイトルの有効な位置を示しています。`XRTTBL_LOCATION_NONE`にセットすると、このウィジェットは表示されません。Managerウィジェットをタイトルとして使用するには、`XmNxrtTblLocation`リソースをセットする前に、全ての子ウィジェットをManagerウィジェットに追加してください。

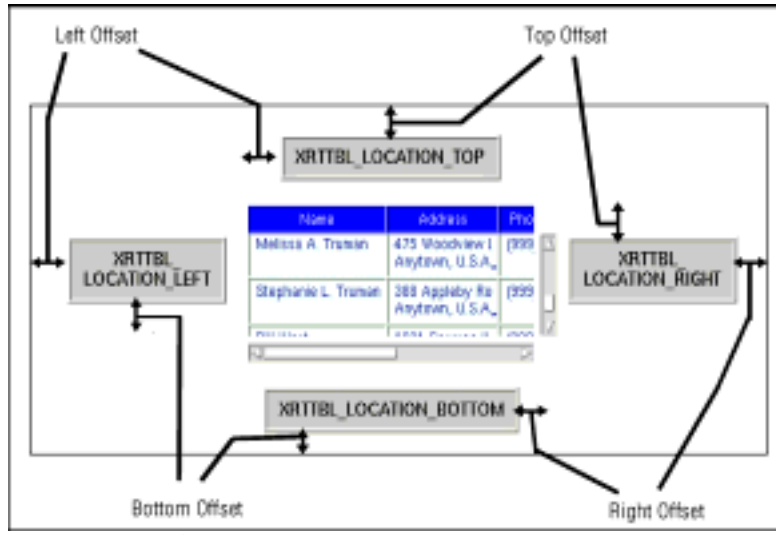


図27　タイトルの位置とオフセット

タイトルの端とテーブルの間のスペースは、四つのオフセットのリソースで調節できます。(XmNxrtTblTopOffset 、 XmNxrtTblBottomOffset 、 XmNxrtTblLeftOffset 、 XmNxrtTblRightOffset) これらのリソースは、タイトルの周囲のスペースを指定します。図27は、タイトルのそれぞれの位置で、オフセットがどのように測られるかを示しています。オフセットは、正、負、XRTTBL_NOVALUEに指定できます。

正の値は、タイトルとテーブルの間のスペースを追加します。負の値は、スペースを削除します。オフセットを0にすると、タイトルは、セル/ラベルまたはテーブルウィジェットの端に接します。

XRTTBL_NOVALUEは、固定されていないオフセットを指定し、実際の位置は、反対のオフセットのリソースの設定に依存します。例えば、XmNxrtTblLeftOffset とXmNxrtTblRightOffsetがXRTTBL_NOVALUEにセットされていると、XRT/tableはタイトルを中央に配置します。一方のオフセットのリソースが明示的にpixel値にセットされている場合は、タイトルはその値に固定されます。図28は、オフセットの指定と位置の関係を示しています。



図28　タイトルのオフセット

回転されたタイトル

XrtLabelウィジェットを使用することにより、テキストやpixmapを回転することができます。XrtLabelのテキストを回転するには、`XmNxrtLblRotation`を`XRTLBL_ROTATE_90`または、`XRTLBL_ROTATE_270`にセットします。

リサイズの処理

テーブルの初期表示とリサイズされた時、XRT/tableは始めにタイトルに必要なスペースを計算し、セルとラベルには残りのスペースを使用します。テーブルのサイズが非常に小さい時は、予期しない結果が得られることがありますので注意してください。

絶対的な位置

アプリケーションは、`XmNx`と`XmNy`リソースによって、(テーブルに対する)タイトルの左上角の座標を調べることができます。

タイトルの削除

表示からタイトルを取り去るには、`XmNxrtTblLocation`を`XRTTBL_LOCATION_NONE`にセットしてください。永久的にタイトルを破棄するには、*XtDestroyWidget()*を呼び出します。

3.7　行の高さと列幅

デフォルトでは、行の高さはテキスト1行分、列幅は10文字分にセットされています。セル値や、pixmap、ウィジェットがセルに入り切らない時は、セルはデフォルトでclipping arrowを表示します。各行は独自の高さを、各列は独自の幅を持つことができます。

行の高さと列幅の指定には、*character*と*pixel*の二つの方法があります。character指定は、高さと幅を行数と文字数で決定します。pixel指定は、高さと幅をpixel数で指定します。

行または列には、どちらか一つの方法しか使用できません。pixel指定は、character指定より優先します。

注意: ユーザがインタラクティブに行/列をリサイズした時、行の高さと列幅は、アプリケーションの指定に関らず、pixelで指定されます。ResizeCellコールバックを用いて、ユーザによるリサイズの通知を受けることができます。

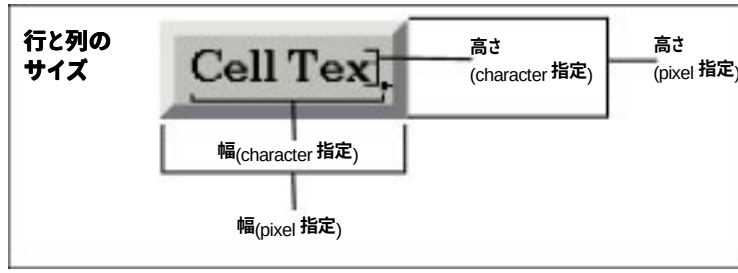


図29　characterとpixelの行/列の指定

3.7.1 character指定

XmNxrtTblCharWidthContext は列が表示できる文字数、XmNxrtTblCharHeightContextは行が表示できる行数を指定します。これらのリソースに行の高さと列幅をコントロールさせるには、XmNxrtTblPixelWidthContextとXmNxrtTblPixelHeightContextをXRTTBL_NOVALUEにしてください。

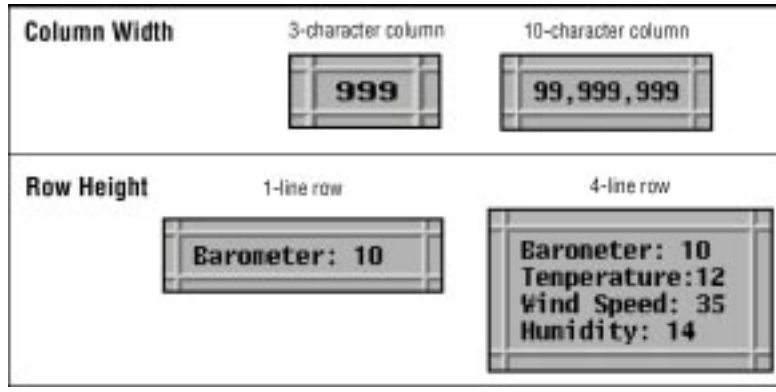


図30　character指定

character指定は、表示する文字数があらかじめわかっている時に便利です。XRT/tableは、指定された文字数がセルやラベルに必ず表示できるよう、一番幅の広い文字や一番背の高い文字を用いてサイズを決めますので、*non-proportional*[1] フォントを使用する時は良い結果が得られます。

[1] non-proportaional(固定幅)のフォントは全て同じ高さと幅を持っています。

次の例は、列10を15文字幅にセットしています。

```
XtVaSetValues(table,
  XmNxrtTblContext, XrtTblSetContext(XRTTBL_ALL, 10),
  XmNxrtTblPixelWidthContext, XRTTBL_NOVALUE,
  XmNxrtTblCharWidthContext, 15,
  NULL);
```

高さと幅がcharacterでセットされている行や列のpixelを知るには、*XrtTblGetCellDimensions()*を使用してください。

3.7.2 pixel指定

XmNxrtTblPixelWidthContextとXmNxrtTblPixelHeightContextは、列幅と行の高さをpixelで指定します。これらのリソースは、pixel値、XRTTBL_NOVALUE、XRTTBL_VARIABLE、XRTTBL_FLEXIBLEにセットできます。

XRTTBL_NOVALUE(デフォルト)にセットされていない限り、これらのリソースが(character指定の代わりに)行の高さや列幅をコントロールします。図31では、XmNxrtTblPixelHeightContextをpixel値にセットした例です。

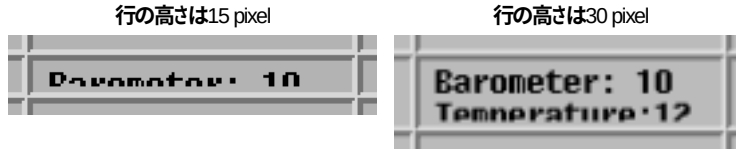


図31　pixel指定

次のコードは、列10を50 pixelにセットしています。

```
XtVaSetValues(table,
  XmNxrtTblContext, XrtTblSetContext(XRTTBL_ALL, 10),
  XmNxrtTblPixelWidthContext, 50,
  NULL);
```

3.7.3 可変の高さと幅

XmNxrtTblPixelWidthContextとXmNxrtTblPixelHeightContextをXRTTBL_VARIABLEにセットして、行や列がテーブルの内容を全て表示できるよう、XRT/tableに自動的にサイズを決めさせることができます。アプリケーションがセルの内容に影響を与える属性を変更すると、XRT/tableはそれに合うよう行や列をリサイズします。[1]

[1] 可変の高さの行(または可変の幅の列)に値がない時は、行(または)列は、非表示になります。

セルがセル値と背景のpixmapを持っている時、XRT/tableは二つの内大きい方に合うようサイズを決定します。セルがウィジェットを含んでいる時は、そのウィジェットの理想的なサイズ(*XtQueryGeometry()*で調べられます)に合うようにセルのサイズを決めます。

注意: ユーザが行または列をリサイズすると、サイズはpixel数によって指定されてしまうので、行と列は可変(variable)ではなくなります。ユーザによるセルの編集は、自動的に行や列のサイズを変更しません。

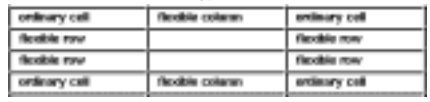
アプリケーションがテーブルに多くの変更を加える場合は、`XmNxrtTblRepaint`を`False`にセットして再描画をオフにし、変更を加えた後でオンに戻してテーブルをアップデートしてください。

可変の高さや幅の行/列のpixel数は、*XrtTblGetCellDimensions()*で知ることができます。

3.7.4 フレキシブルな行と列

`XmNxrtTblPixelHeightContext` または `XmNxrtTblPixelWidthContext` を`XRTTBL_FLEXIBLE`にセットして、アプリケーションはある行または列をフレキシブルに指定できます。このリソースがセットされると、図32のようにテーブルのリサイズに従って、これらの行や列もリサイズするようになります。

リサイズ前のテーブル



リサイズ後のテーブル



図32 フレキシブルな行と列を持ったテーブルのリサイズ

フレキシブルな行/列がリサイズされる場合、これらの行の高さや列幅は全て同じ値にリサイズされます。

全てのフレキシブルな行/列に最小の高さと幅をセットするには、`XmNxrtTblMinFlexibleHeight`と`XmNxrtTblMinFlexibleWidth`リソースを使用します。例えば、次のコードは、最小の行の高さを20、最小の列幅を50としています。

```
XtVaSetValues(table,
    XmNxrtTblMinFlexibleHeight,   20,
    XmNxrtTblMinFlexibleWidth,50,
    NULL);
```

3.7.5 行や列を非表示にする

`XmNxrtTblPixelHeightContext/XmNxrtTblPixelWidthContext`を0 pixelにして、アプリケーションは、行や列をエンドユーザから見えないようにセットできます。[1] 行/列は消えてしまったように見えますが、アプリケーションでは属性をセットしたりセル値を変更することができます。(図33参照)

カレント・セルがある行/列は、非表示にできません。

Before After

[1] ユーザはインタラクティブに行/列を表示・非表示に切り替えできません。

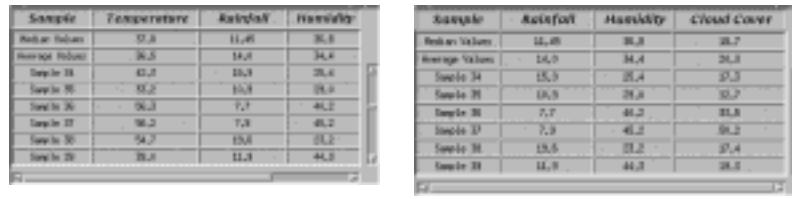


図33 “Temperature”の列を非表示

非表示になっている行/列を表示するには、pixel指定の高さまたは幅をpixel値か、XRTTBL_NOVALUE(character指定を使用する場合)にセットします。

3.8 セルとラベルのマージン

セルの罫線とセルの内容の間のスペースは、XmNxrtTblMarginHeightとXmNxrtTblMarginWidthで調節できます。図34は、異なったマージンの例です。これらのリソースは、テーブル内の全てのセルとラベルに影響します。(マージンは各セル毎に設定できません。)

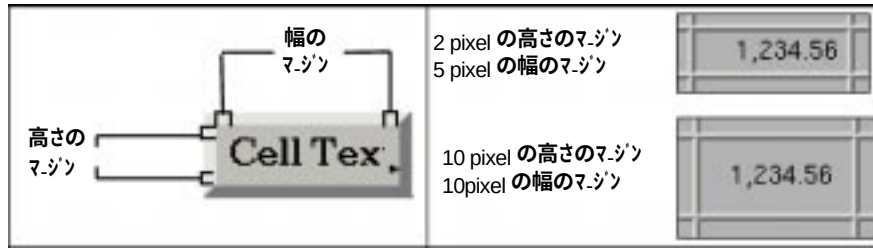


図34 マージンの高さと幅

マージンのリソースは、XmNhighlightThicknessの値より小さくは設定できません。(カレント・セルのXmTextウィジェットは、マージンのスペースにハイライトを描きます。)

3.9 前景色、背景色と影の色

色の指定

XRT/tableの背景/前景色のリソースは、色のストリングを値として取ります。(詳細は2.8参照) 背景色と前景色の設定には、テキストが読みやすいようにコントラストに注意してください。下の例のように、”LightBlue”や”Wheat”等のパステル色は背景色に、濃い色は前景色に適しています。

```
XtVaSetValues(table,
  XmNxrtTblContext, XrtTblSetContext(5, XRTTBL_ALL),
  XmNxrtTblBackgroundContext, “LightBlue”,
  XmNxrtTblForegroundContext, “NavyBlue”,
  NULL);
```

pixel値の使用

背景/前景色にPixel値を使用する場合は、XmNxrtTblBackgroundPixelContextとXmNxrtTblForegroundPixelContextリソースをセットします。

XmNxrtTblBackgroundPixelContextがセットされると、XmNxrtTblBackgroundContextが変更されます。(逆の場合も同じです。) 同様にして、XmNxrtTblForegroundPixelContextは、XmNxrtTblForegroundContextの値を変更します。

背景色

テーブルの背景色は、継承されたXmNbackgroundリソースによって決定されます。各セルとラベルは、XmNxrtTblBackgroundContextで指定される独自の背景色を持つことができます。デフォルトでは、全てのセル/ラベルの背景色は、XmNbackgroundから継承されています。

前景色

テーブルの前景色は、継承されたXmNforegroundリソースによって決定されます。各セルとラベルは、XmNxrtTblForegroundContextで指定される独自の前景色を持つことができます。デフォルトでは、全てのセル/ラベルの前景色は、XmNforegroundから継承されています。

選択されたセル

選択されているセルの背景/前景色は、XmNxrtTblSelectedBackgroundとXmNxrtTblSelectedForegroundで指定されます。デフォルトでは、選択されたセルの背景色は黒、前景色は白です。カレント・セルは、罫線にこの色を表示します。

セルに定義されている前景色と背景色(XmNxrtTblForegroundContextとXmNxrtTblBackgroundContext)を使用するには、XmNxrtTblSelectedBackgroundをNoneにします。こうすると、セルを選択しても、スクリーン上では何も変化は起こりませんが、定義されているXmNxrtTblSelectCallbackは呼び出されます。

子ウィジェット

テーブルに管理される子ウィジェット(スクロールバー、XmTextカレント・セル、タイトル)の色は、ウィジェットのXmNbackgroundとXmNforegroundリソースに

よって指定できます。セル/ラベルの中のウィジェットの色をセットするには、そのセル/ラベルのXmNxrtTblBackgroundContextとXmNxrtTblForegroundContextを使用します。

繰り返して色を指定する

行や列には、ある一定のパターンで繰り返す背景/前景色を簡単に指定することができます。繰り返すパターンの背景色を指定するには、XmNxrtTblRepeatBackgroundColorsリソースを次の例のようにセットします。

```
String backcolors[] = {"grey95", "grey65", NULL};

XtVaSetValues(table,
  XmNxrtTblContext, XrtTblSetContext(XRTTBL_ALL,XRTTBL_ALL),
  XmNxrtTblBackgroundContext, XRTTBL_REPEAT_ROW,
  XmNxrtTblRepeatBackgroundColors, backcolors,
  NULL);
```

繰り返すパターンの前景色のリストは、XmNxrtTblRepeatForegroundColorsリソースをセットして作成します。

繰り返すパターンには何色でも指定できます。また、色は常にリストにある順に選択されていきます。

カレントcontextの背景色を、XmNxrtTblRepeatBackgroundColorsで定義される色のリストにセットするには、XmNxrtTblBackgroundContextをXRTTBL_REPEAT_ROW(行の順に繰り返す)または、XRTTBL_REPEAT_COLUMN(列の順に繰り返す)にセットします。

XRTTBL_REPEAT_ROW



XRTTBL_REPEAT_COLUMN

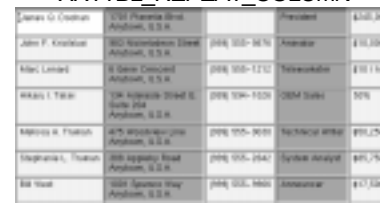


図35　繰り返すパターンの背景色

カレントcontextの前景色を、XmNxrtTblRepeatForegroundColorsで定義される色のリストにセットするには、XmNxrtTblForegroundContextをXRTTBL_REPEAT_ROWまたは、XRTTBL_REPEAT_COLUMNにセットします。

影の色

XmNxrtTblBottomShadowContextとXmNxrtTblTopShadowContextリソースは、カレントcontextの上部と下部の影の色を指定します。これらのリソースがXRTTBL_NOVALUEにセットされている時は、XRT/tableは自動的に適した影の色を決定します。

影の幅は、XmNxrtTblShadowThicknessリソースでコントロールします。

3.10 セルとラベルのテキストの行揃え

セルとラベル内でのテキストの行揃え(縦方向と横方向)は、XmNxrtTblAlignmentContextで指定します。セル/ラベル値は、中央を始め、セル/ラベルのどの側にも配置できます。図36は、このリソースの値が、(XmNstringDirectionがXmSTRING_DIRECTION_L_TO_Rの時)どのようにテキストの表示に影響するかを示しています。

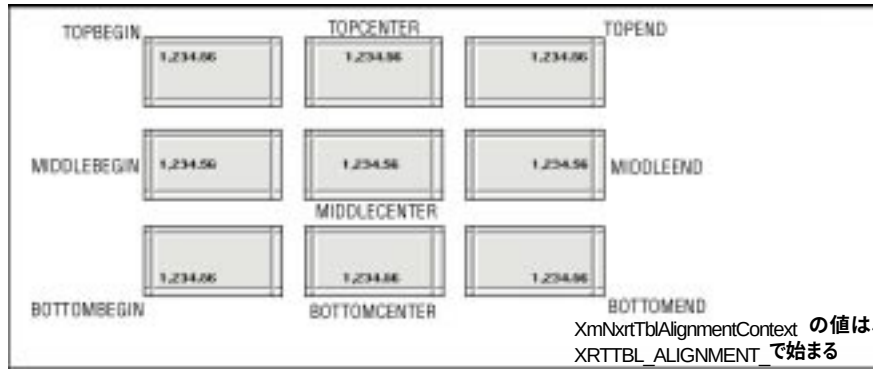


図36 行揃え

テキストが実際にどのように表示されるかは、継承されるXmNstringDirectionの設定に依存します。例えば、XmNstringDirectionがXmSTRING_DIRECTION_R_TO_Lの時に、XmNxrtTblAlignmentContextをXRTTBL_ALIGNMENT_TOPBEGINにすると、テキストは右から左に右揃えに表示されます。

3.11 セルとラベルのフォント

XmNxrtTblFontListContextで、セルまたはラベルのテキストのフォントを指定できます。各セル/ラベルには、二つ以上のフォントがサポートされています。Compound String(複合ストリング)の作成と使用法は6.5に解説されています。

XRT/tableは、Xサーバで使用可能なXフォントならどれでも使用できます。フォント名で指定する場合は、X Windowプログラムの*xlsfonts*または、Xlibコールの*XListFonts()*を用いて、どのフォントがサーバで使用可能か調べてください。Xlibコールの*XLoadFont()*または*XLoadQueryFont()*で、フォントのリソースIDが取得できます。フォントの検索と設定の詳細は、Xプログラミングに関する資料を参照してください。

Motifのリソース変換プログラムを使用して、フォント・ストリングをXmNxrtTblFontListContextが要求するXmFontListデータ型に変換すると、フォント・リストが簡単に指定できます。

```
XtVaSetValues(table,
  XmNxrtTblContext, XrtTblSetContext(XRTTBL_LABEL,XRTTBL_ALL),
    XtVaTypedArg, XmNxrtTblFontListContext,
    XmRString, "-*-times-medium-*-*--18-*-iso8859-*",
    strlen("-*-times-medium-*-*--18-*-iso8859-*") +1,
  NULL);
```

3.12 罫線の種類と場所

全てのセルとラベルの周囲には罫線があります。罫線の種類と場所は、各セルとラベル毎にカスタマイズできます。罫線の幅は、テーブル全体に指定されます。

その他、セル、ラベルを囲むフレームや、プレビュー・スクロール・ウィンドウの罫線もカスタマイズできます。。

罫線の種類

XmNxrtTblBorderTypeContextは、セルまたはラベルの周囲に描かれる罫線の種類を指定します。図37は、用意されている罫線の種類です。

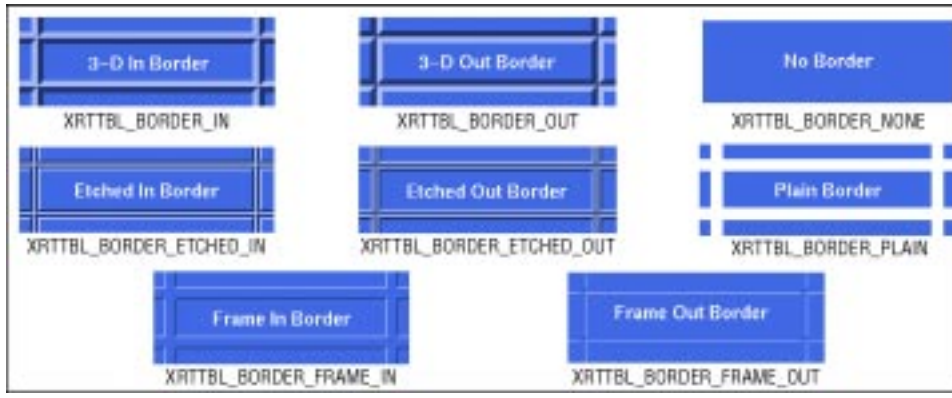


図37　罫線の種類

次の例は、ある行に空の罫線(=罫線なし)をセットします。

```
XtVaSetValues(table,
  XmNxrtTblContext, XrtTblSetContext(10, XRTTBL_LABEL),
  XmNxrtTblBorderTypeContext, XRTTBL_BORDER_NONE,
  NULL);
```

罫線の幅

罫線の幅は、XmNxrtTblShadowThicknessで指定します。このリソースは、テーブル全体に適用されます。デフォルトは、2 pixel幅です。図38は、異なる幅の罫線の例です。



図38　罫線の幅

罫線の場所

XmNxrtTblBorderSidesContextは、(XmNxrtTblBorderTypeContextで指定された)罫線を表示する場所を指定します。デフォルトでは、罫線はセル/ラベルの四辺に表示されます。図39は、罫線が表示される場所を変更した例です。

15 1/8	9 1/8	ISG Tech	ISGTF	10 1/2	10	10 1/4		16
28 1/2	9 1/4	Immune Rsp	IMNR	12 5/8	12	12 1/8	-1/4	823
27 1/4	13 3/8	Informix	IFMX	22	21	22	+1/2	6311

図39　カスタマイズされたセルの罫線

XmNxrtTblBorderSidesContextの有効な値は、次の通りです。

- XRTTBL_BORDERSIDE_LEFT
- XRTTBL_BORDERSIDE_RIGHT
- XRTTBL_BORDERSIDE_TOP
- XRTTBL_BORDERSIDE_BOTTOM
- XRTTBL_BORDERSIDE_ALL
- XRTTBL_BORDERSIDE_NONE

罫線を描く場所は、ORを使用して指定します。

```
XtVaSetValues(table,
  XmNxrtTblContext, XrtTblSetContext(10, 10),
  XmNxrtTblBorderSidesContext,
    XRTTBL_BORDERSIDE_LEFT
    | XRTTBL_BORDERSIDE_TOP
    | XRTTBL_BORDERSIDE_BOTTOM,
  NULL);
```

フレームの属性

XmNxrtTblFrameBorderTypeリソースを使用して、セルとラベル領域を囲むフレームの線の種類を指定します。このリソースの値は、(XRTTBL_BORDER_FRAME_INとXRTTBL_BORDER_FRAME_OUTは除く)XmNxrtTblBorderTypeCotnextの値と同じです。デフォルト値は、XRTTBL_BORDER_NONEです。

XmNxrtTblFrameShadowThicknessリソースは、セルとラベル領域を囲むフレームの影の幅を指定します。デフォルト値はゼロ(フレームの影はなし)です。

```
XmNxrtTblFrameBorderType=XRTTBL_BORDER_IN
 XmNxrtTblFrameShadowThicknes=2
```

Employee Listing

Name	Address	Phone Number	Position	Salary
James Q. Doohan	1701 Planetia Blvd. Anytown, U.S.A.		President	$245,000
John F. Kricfalusi	983 Nickelodeon Street Anytown, U.S.A.	(999) 555-9876	Animator	$10,000
Marc Lenard	6 Gene Crescent Anytown, U.S.A.	(999) 555-1212	Telemarketer	$10 / hr.

図40　フレームが表示されたテーブル

影の色は、テーブルの背景色より計算されます。この色を変更したい時は、リソースファイルを使用します。または*XtNameToWidget()*を使用して、該当するフレームのウィジェットIDを取得し、フレームのXmNtopShadowColorとXmNbottomShadowColorリソースをセットすることもできます。

プレビュー・ウィンドウの枠線

このウィンドウのデフォルトはXRTTBL_BORDER_NONEですが、*XrtTblSetPreviewProperties()*を使用してこの値を変更することができます。

3.13　背景のPixmap

XRT/tableは、テーブルのセルまたはラベルにpixmapを表示できます。pixmapは、セルのマージンの内側に表示され、セルにセル値が含まれている場合は、pixmapはその背景に表示されます。[1]

イメージのフォーマット

XRT/tableは、XBMまたはXPMフォーマットのpixmapをサポートします。その他のフォーマット(TIFF、GIF、Sun Rasterfile)は、XBMまたはXPMに変換してください。標準のX bitmapや、サンプルのXPM pixmap(*$XRTHOME/src/table/samples*)を使用して、この機能を試すことができます。

pixmapの指定

XmNxrtTblPixmapContextは、カレントcontextのセルまたはラベルの背景に表示するpixmapを指定します。このリソースは、pixmapを値として取ります。XBM/XPMイメージは、Cコードに直接入れることもできますが、これらのイメージをファイルで管理して必要な時に読み込む方が簡単です。pixmapは、使用後必ず解放してください。XRT/tableは、セル/ラベルに表示されているpixmapをコピーしません。

[1] セルにウィジェットが含まれている場合は、XRT/tableはウィジェットのXmNbackgroundPixmapリソースをそのpixmapにセットします。

XmNxrtTblClipPixmapContextは、pixmapの表示部分を切り取るclippingマスクを指定します。

ファイルに保管されているイメージを指定する場合、Motifのリソース変換プログラムを利用して、ファイル名のストリングをXmNxrtTblPixmapContextが要求するPixmapデータ型に変換する方法が一番簡単です。

```
XtVaSetValues(table,
  XmNxrtTblContext, XrtTblSetContext(1,1),
    XtVaTypedArg, XmNxrtTblPixmapContext,
    XmRString, "my_image.xpm",
    strlen("my_image.xpm") + 1,
  NULL);
```

pixmapの削除

テーブルからpixmapを削除するには、XmNxrtTblPixmapContextをXmUNSPECIFIED_PIXMAPにセットしてください。

pixmapのレイアウト

セル内のpixmapの位置は、XmNxrtTblPixmapLayoutContextリソースで指定します。pixmapは、セルの中央や片側に配置したり、タイルのように並べることもできます。図41は、このリソースがどのようにpixmapを配置するか示しています。

タイルのように並べられているとき、pixmapは、XmNxrtTblMarginHeightとXmNxrtTblMarginWidthリソースの値を無視して、セル/ラベルの罫線まで表示されます。

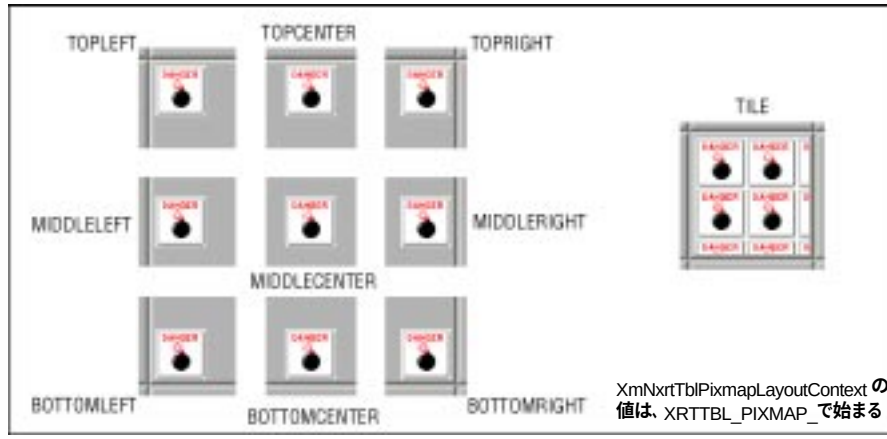


図41　pixmapのレイアウト

影とpixmap

pixmapがセルまたはラベルに表示されている時、セル/ラベルの影が描かれないように指定することができます(pixmapはセルの罫線まで表示されます)。この場合、XmNxrtTblPixmapShadowContextリソースをFalseにセットしてください。

```
XtVaSetValues(table,
  XmNxrtTblContext, XrtTblSetContext(1,1),
  XmNxrtTblPixmapShadowContext, False);
```

デフォルトでは、XmNxrtTblPixmapShadowContextはTrueで、セル/ラベルの影は表示されます。(XmNxrtTblShadowThicknessで、影の幅を指定します。)

clip pixmapの指定

XmNxrtTblClipPixmapContextは、カレントcontextにpixmapが描かれる場合、clip pixmap(clipマスク)を指定します。clip pixmapを使用すると、pixmapの一部だけが表示されるように、イメージを切り取ったり、マスクをかけることができます。デフォルトでは、XmNxrtTblClipPixmapは、XmUNSPECIFIED_PIXMAP(clip pixmap指定なし)です。

3.14 テキストのclippingとoverflow

セル値やラベル値がセル/ラベルに入り切らない時は、セル値を切り取ったり(clip)、横隣の空白のセルに渡って表示(overflow)させることができます。XmNxrtTblOverflowContextリソースで、どのように表示するか指定します。図42は、各設定による、表示結果です。

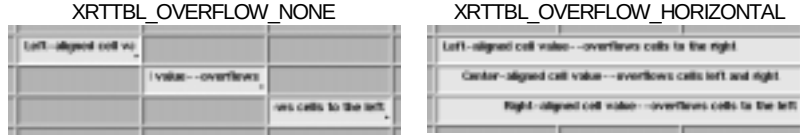


図42　切り取られた場合と隣のセルに渡って表示された場合のセル値

clipping

XmNxrtTblOverflowContextがXRTTBL_OVERFLOW_NONEの時、セル/ラベル値は、セルの縦方向にも横方向にも入り切らない場合、切り取られます。

切り取られた値を持つセルは、右下にclipping arrowを表示します(デフォルト)。[1] これは、表示されている以外にもテキストが存在することを示します。このclipping arrowを非表示にするには、XmNxrtTblDisplayClipArrowsをFalseにしてください。

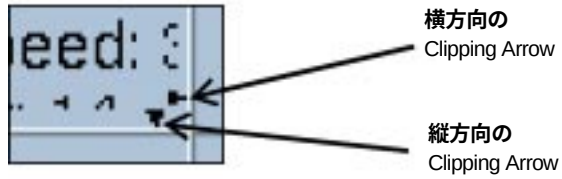


図43　clipping arrowの表示

overflow

XmNxrtTblOverflowContextがXRTTBL_OVERFLOW_HORIZONTALの場合、セル/ラベル値が横方向にセルに入り切らなければ、隣の空白のセルに渡って表示されます。

[1] カレント・セルはXmTextウィジェットなので、clipping arrowを表示しません。

このoverflowはダイナミックで、セル/ラベルの値に依存します。値がセル/ラベルへ入り切るかどうかを左右する変更がテーブルに加えられると、このoverflowも変化します。

セルは、元のセルから開始して、セル/ラベル値全体が横方向に表示されるまで、または、空白でないセルに到達するまで、(視覚的に)延長されます。overflowの方向は、XmNxrtTblAlignmentContextによって決定されます。(図44参照)

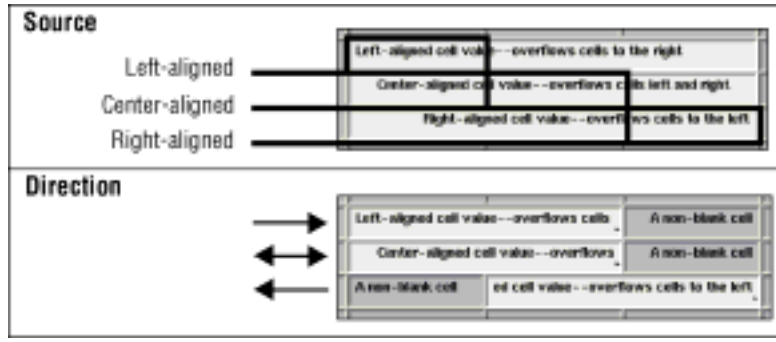


図44　overflowする元のセルとその方向

あるセル値が複数のセルにまたがって表示(overflow)されている時、これらのセルは結合して一つのセルのように見えますが、実際は、これらのセルはトラバースも編集もできます。ユーザが、これらのセルのうちの一つに値を入力すると、overflowしている値はそのセルで切り取られます。overflowしている値を編集する時は、元のセルにトラバースしてください。

同じテーブルで、セル値のoverflowとCellValue/LabelCreateコールバック(XmNxrtTblCellValueCallbackとXmNxrtTblLabelCreateCallbackで指定)を使用する時は、注意してください。コールバックによってセットされた値は、overflowしているセルによって隠されることがあります。

ワードラップ

セル内のテキストがワードラップして表示されるように指定することができます。この場合、セルでの改行は、空白、タブ、newlineのある場所でのみ行われます。但し、セル/ラベルの値は、表示範囲に入りきらない時は、clipまたはoverflowします。

ワードラップを指定するには、XmNxrtTblWordWrapContextをTrueにセットします。デフォルトでは、XmNxrtTblWordWrapContextはFalseにセットされ、ワードラップは実行されません。

overflowとspanning

セル値が複数のセルにまたがって表示されるoverflowと、セルのspanning(次の項参照)は、同じように見えますが、その用途は異なります。overflowは、ユーザがセル値に対するコントロールが必要なスプレッドシート形式のアプリケーションや、複数のセルにまたがる列ラベルに適しています。pixmapやウィジェットをoverflowさせる時や、複雑なフォーマットの表が必要な時は、spanningを利用してください。

3.15 セルとラベルのspanning

spanningは、セルやラベルの範囲を結合し、一つのセル/ラベルとして処理します。結合されたセル範囲は、複数の行と(または)列に渡って一つのセル/ラベルであるように動作します。spanningは、複雑な表を作成したり、大きなpixmapやウィジェットを表示する場合を始め、多くの用途に利用できます。

spanningを実行する時は、結合する範囲の左上角のセルから全体を選択します。左上のセルが元のセルとなり、その値と属性は結合された範囲全体に適用されます。(範囲内の他のセル/ラベルの値と属性が隠されます。)

結合する範囲は、必ずその範囲の左上角から開始してください。セルとラベルを一緒に結合することはできません。

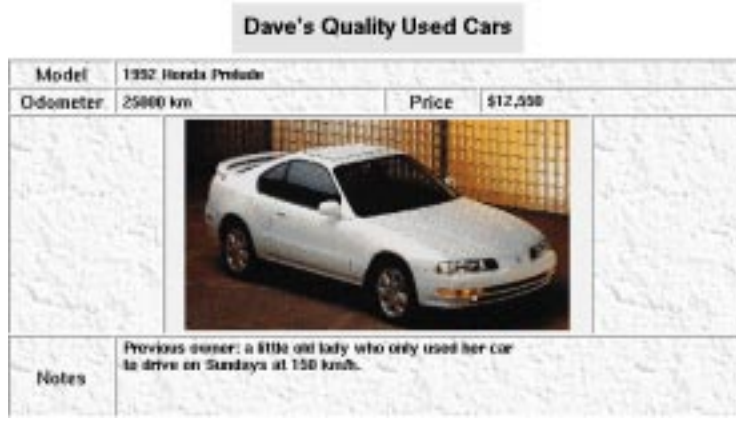


図45　spanningの使用例

XmNxrtTblSpanListは、現在結合されている範囲のリスト(NULLで終了)を指定します。リストの各範囲は、XrtTblRange構造体です。

```
typedef struct {
  int start_row, start_column;
  int end_row, end_column;
} XrtTblRange;
```

次の例は、二つのspanningされた範囲を定義しています。

```
static XrtTblRange span1 = {0, 0, 4, 2};
static XrtTblRange span2 = {7, 1, 7, 4};
static XrtTblRange *all_spans[] =
  {&span1, &span2, NULL};

XtVaSetValues(table,
  XmNxrtTblSpanList, all_spans,
  NULL);
```

spanningの追加

現在spanningされている範囲のリストに、範囲を追加するには、*XrtTblRangeListAdd()*を使用します。この手続きは、次のプロトタイプを持っています。

```
XrtTblRange **
XrtTblRangeListAdd(
  XrtTblRange **list,
  int position,
  XrtTblRange *range
  )
```

*XrtTblRangeListAdd()*は、追加した範囲を含む、新しい範囲のリストを返します。XmNxrtTblSpanListをこのリストにセットすることができます。次の例は、現在の選択をspanningされている範囲のリストに追加しています。

```
XrtTblRange selection, **span_list;

if (!XrtTblGetSelectedRange(table, 0, &selection))
  return;

XtVaGetValues(table, XmNxrtTblSpanList, &span_list, NULL);
span_list = XrtTblRangeListCopy(span_list);
span_list = XrtTblRangeListAdd(span_list, XRTTBL_MAXINT,
  &selection);

XtVaSetValues(table,
  XmNxrtTblSpanList,        span_list,
  XmNxrtTblSelectedCellList,    NULL,
  NULL);
XrtTblRangeListFree(span_list);
```

spanningされている範囲の処理

全てのContextリソースとその他の属性は、spanningされている範囲の(左上にある)元のセルにセットされていなければなりません。範囲内の他のセルの属性もセットしたり取り出すことができますが、元のセルと結合されているため変更は見えません。

セルまたはラベルが、現在spanningされている範囲の一部であるかどうかは、*XrtTblGetSpanByRowCol()*を呼び出して調べることができます。

spanningの削除

現在のリストからspanningされた範囲を削除するには、リストのコピーで*XrtTblRangeListDelete()*を使用し、XmNxrtTblSpanListを返された範囲のリストにセットします。XmNxrtTblSpanListをNULLにすると、spanningされている範囲を全て削除できます。

3.16 セルとラベルのフラッシュ

テーブルにハイライトさせたい大切なデータがある時、そのセルの色をフラッシュさせると、エンドユーザに効果的に知らせることができます。*XrtTblFlash()*を使用して、異なったcontextをフラッシュさせることができます。

カレント・セルは、フラッシュするセルにトラバースされても、フラッシュしません。(但し、罫線のフラッシュは表示されます)

以下は、*XrtTblFlash()*のプロトタイプです。

```
Boolean
XrtTblFlash(
  Widget       table,
  XrtTblContext    *context,
  int      count,
  String       bg_color,
  String       fg_color,
  unsigned long    on_duration,
  unsigned long    off_duration
  )
```

この手続きは、context、フラッシュする回数、フラッシュの色、フラッシュのオンとオフの時間を取ります。フラッシュをセットすると、それ以前にセットされていたフラッシュは置き換えられます。

フラッシュのおおよその時間は、次の式で求められます。

total duration = count(on_duration + off_duration)*

*on_duration*と*off_duration*のパラメータは、それぞれフラッシュのオンとオフをミリ秒で指定します。*count*は、フラッシュする回数を指定します。*count*が`XRTTBL_MAXINT`であると、フラッシュは永久に継続します。

背景色と前景色の組み合わせで異なったフラッシュを作成できます。

- 反対色のフラッシュ: *bg_color*を`NULL`にセットしセルの前景色を使用、*fg_color*を`NULL`にセットしセルの背景色を使用する。
- テキストのフラッシュ: *bg_color*をcontextの背景色と同じ色にセットし、*fg_color*をフラッシュさせたい色にセットする。
- 背景のフラッシュ: *fg_color*をcontextの前景色と同じ色にセットし、*bg_color*をフラッシュさせたい色にセットする。

次の例は、全ての行ラベルを無限回数フラッシュさせます。

```
XrtTblFlash(table,
  XrtTblSetContext(XRTTBL_LABEL, XRTTBL_ALL),
  XRTTBL_MAXINT, "red", "yellow", 100, 100);
```

注意: 複数のセルにおいてオンとオフの間隔が同じに設定されている場合、それらのフラッシュは同時に行われます。

フラッシュの停止

一つのcontextでフラッシュを停止するには、*count*が0にセットされた*XrtTblFlash()*を呼び出します。その他のパラメータは無視されます。

テーブル全体で全てのフラッシュを停止するには、*context*をテーブル全体に、*count*を0にセットした*XrtTblFlash()*を呼び出します。

```
XrtTblFlash(table,
  XrtTblSetContext(XRTTBL_ALL, XRTTBL_ALL),
  0, NULL, NULL, 0, 0);
```

3.17 ユーザのデータ

XmNxrtTblUserDataContextリソースを使用して、アプリケーションはセルやラベルのcontextのデータを記憶することができます。XRT/tableは、このリソースを内部で使用しません。このリソースは、テーブルの一部に二次的なデータ(例: セルが最後に編集された時間等)を付属させる時に便利です。

3.18 テーブルの印刷

多くのアプリケーションでは、エンドユーザに画面で表示されているテーブルのハードコピーを提供する必要があります。XRT/tableは、テーブルのPostScriptイメージを生成します。このイメージは、デバイスに依存しないPostScriptフォントとオペレータで作成されます。このPostScriptファイルは、PostScriptプリンタや他のドキュメントにインポートすることができます。

3.18.1 PostScriptの生成

テーブルのPostScriptイメージ生成には、*XrtTblDrawPS()*と*XrtTblVaDrawPS()*の二つの手続きがあります。これらは同じ出力を生成しますが、異なるインターフェースを提供しています。*XrtTblVaDrawPS()*は、*XtVaSetValues()*と同様に*VarArgs-style*インターフェースを使用し、デフォルトから変更する必要のある引数だけをセットするようになっています。一方、*XrtTblDrawPS()*は*ArgList-style*インターフェースを使用し、異なる印刷の要求に対して、それぞれ異なる引数のセットを保持できるようになっています。どちらの手続きもAppendix Bに解説されていますが、引数は、*XrtTblVaDrawPS()*のみ詳細があります。

テーブルにタイトルや他のラベル・ウィジェットがある場合、これらはテーブル・ウィジェットによって管理(managed)された場合にのみプリントされます。

ヘッダーとフッターの定義

テーブルが印刷されるページにヘッダーとフッターを定義するには、*XrtTblDrawPS()*または*XrtTblVaDrawPS()*を呼び出す時に、XRTTBL_PS_HEADERとXRTTBL_PS_FOTTERをセットします。

ヘッダーとフッターに使用されるフォントは、XRTTBL_PS_HEADER_FONTとXRTTBL_PS_FOOTER_FONTによってコントロールされます。また、ヘッダーとフッターのマージンは、XRTTBL_PS_HEADER_MARGINとXRTTBL_PS_FOOTER_MARGINがコントロールします。

PostScript出力のガイドライン

下記は、*XrtTblDrawPS()*と*XrtTblVaDrawPS()*を使用する場合のガイドラインです。

- XRTTBL_PS_PAPERSIZE_とXRTTBL_PS_MARGIN_は、*image*の上部でなく*paper*の上部に相対します。例えば、テーブルの方向が横向きでも、XRTTBL_PS_PAPERSIZE_TOPは、イメージの上部でなくページの上部を測定します。

- 複数のウィジェットを一ページに印刷する時は、XRTTBL_PS_MARGIN_を使用して、各テーブルの位置を重なり合わないように指定し、最後のイメージ以外は、XRTTBL_PS_SHOWPAGEをFalseに設定してください。
- XRTTBL_PS_ORIENTATIONに、XRTTBL_PS_BESTを指定する場合、印刷されるテーブルのイメージは実行される度に分析されるので、同じテーブルでも異なったサイズの範囲が指定された時は、異なった方向に印刷されることがあります。

PostScript出力の制限事項

印刷手続きには、幾つか制限事項があります。

- これらの手続きは、1ページに少なくとも1行または1列を印刷します。もし、セルのサイズや印刷する際のスケーリングにより、1行または1列が1ページに収まり切らないときは、PostScript出力ではカットされてしまいます。
- 自動色指定(XRTTBL_PS_COLORをXRTTBL_PS_AUTO)するときは、注意してください。モノクロのPostScriptプリンタによっては、間違ってカラー・プリンタと認識してしまうものがあります。(SPARCprinter等) これらのプリンタでモノクロの出力をするには、ファイルにある"/color? false def"のコメントを取ってください。
- XRT/tableの印刷は、64 x 64 pixelの比較的小さなpixmapが繰り返してプリントされる場合に最適化されます。大きなpixmapが印刷される場合は、パフォーマンスが悪くなります。

XrtTblVaDrawPS()の例

次の例は、幾つかの列にページ区切りを挿入し、横方向にテーブルの範囲を出力しています。その他の属性は全てデフォルトのままです。

```
FILE *fp;
int col_pagebreaks[] = { 5, 10, 25, -1 };
int numpages;
XrtTblRange range = { 10, 0, 40, 30 };
fp = fopen("mytable.ps", "w");
if (fp) {
  numpages = XrtTblVaDrawPS(table, fp,
    XRTTBL_PS_ORIENTATION,      XRTTBL_PS_LANDSCAPE,
    XRTTBL_PS_COL_PAGEBREAKS,       col_pagebreaks,
    XRTTBL_PS_SHOW_FROZEN_COLS, XRTTBL_PS_ALL,
    XRTTBL_PS_SHOW_FROZEN_ROWS, XRTTBL_PS_ALL,
    XRTTBL_PS_CELL_RANGE, &range,
    NULL);
  printf("%d page%s written.\n", numpages,
    (numpages==1)? "" : "s");
}
else {
  printf("error opening file\n");
}
```

XrtTblDrawPS()の例

次の例は、A4サイズの用紙にテーブルを印刷する際に必要な引数のリストを作成し、*XrtTblDrawPS()*を呼び出しています。

```
int ac, num_pages;
FILE *fp;
double scale = 100.0;
double a4_width = 21.0;
double a4_height = 29.7;
double a4_margin = 1.0;
XrtTblPSarg al[20];

num_pages = 0;
fp = fopen("mytable.ps", "w");
if (fp) {
  ac = 0;
  XtSetArg(al[ac], XRTTBL_PS_SCALE, &scale); ac++;
  if (paper_type == METRIC_A4){
    XtSetArg(al[ac], XRTTBL_PS_UNITS, XRTTBL_PS_CM); ac++;
    XtSetArg(al[ac], XRTTBL_PS_PAPERSIZE_HEIGHT, &a4_height);
      ac++;
    XtSetArg(al[ac], XRTTBL_PS_PAPERSIZE_WIDTH, &a4_width);
      ac++;
    XtSetArg(al[ac], XRTTBL_PS_MARGIN_TOP, &a4_margin); ac++;
    XtSetArg(al[ac], XRTTBL_PS_MARGIN_BOTTOM, &a4_margin); ac++;
    XtSetArg(al[ac], XRTTBL_PS_MARGIN_LEFT, &a4_margin); ac++;
    XtSetArg(al[ac], XRTTBL_PS_MARGIN_RIGHT, &a4_margin); ac++;
  }
  num_pages = XrtTblDrawPS(table, fp, al, ac);
  fclose(fp);
}
printf("%d page%s written.\n", num_pages,
  (num_pages == 1) ? "" : "s");
}
```

サンプルのポップアップ･ウィンドウ

*XrtTblVaDrawPS()*を使用し、*lp*へのファイルのパイプの方法を示すポップアップ･ウィンドウのソースコードが、*$XRTHOME./src/table/tools/pdbox_table.c*にあります。



図46　サンプルの印刷ダイアログ・ボックス

3.18.2　PostScriptフォントのカスタマイズ

PostScriptの出力手続きは、テーブルで使用されているXフォントに対応するPostScriptフォントを自動的に選択します。または、`XmNxrtTblPsFontMapList`を使用して、Xフォントに対応するglobal PostScriptのリストを定義することもできます。このリストの各エレメントは、`XrtTblPsFontMap`型で、次のように定義されます。

```
typedef struct {
  String  font_name;
  String  tag;
  String  PS_font;
  double  point_size;
} XrtTblPSFontMap;
```

この構造体では、*PS_font*はPostScriptフォント、*point_size*はそのポイント・サイズです。(図47 標準のPostScriptフォント名参照) Xフォントは、次の二つの方法で指定できます。

- *font_name*を使用してXフォント名を指定する
  例: `"-*-times-bold-r-normal-*-140-*"`.
- *tag*を使用して、`XmNfontList`のフォント・タグ(`bold`等)を指定する

このリストは、*PS_font*が`NULL`にセットされたエレメントを定義することにより終了します。

*point_size*が0にセットされると、PostScriptフォントのポイント・サイズは、対応するXフォントのサイズより推測されます。

Times-Roman	Helvetica	NewCenturySchlbk-Roman
Times-Bold	Helvetica-Bold	NewCenturySchlbk-Bold
Times-Italic	Helvetica-Oblique	NewCenturySchlbk-Italic
Times-BoldItalic	Helvetica-BoldOblique	NewCenturySchlbk-BoldItalic
Palatino-Roman	Bookman-Light	Helvetica-Narrow
Palatino-Bold	Bookman-Demi	Helvetica-Narrow-Bold
Palatino-Italic	Bookman-LightItalic	Helvetica-Narrow-Oblique
Palatino-BoldItalic	Bookman-DemiItalic	Helvetica-Narrow-BoldOblique
Courier	AvantGarde-Book	Symbol
Courier-Bold	AvantGarde-Demi	ZapfChancery-MediumItalic
Courier-Oblique	AvantGarde-BookOblique	ZapfDingbats
Courier-BoldOblique	AvantGarde-DemiOblique	

図47　35の標準のLaserWriter IIフォント

次の例は、XmNxrtTblPsFontMapListを使用して、Helvetica XフォントにHelvetica Narrow PostScriptフォントが対応するように指定しています。

```
XrtTblPSFontMap fontlist[] = {
  {"-*-helvetica-medium-r-*-*-*-*-*-*-*-*-iso8859-1",
    NULL, "Helvetica-Narrow", 0},
  {"-*-helvetica-bold-r-*-*-*-*-*-*-*-*-iso8859-1",
    NULL, "Helvetica-Narrow-Bold", 0},
  {"-*-helvetica-medium-o-*-*-*-*-*-*-*-*-iso8859-1",
    NULL, "Helvetica-Narrow-Oblique", 0},
  {"-*-helvetica-bold-o-*-*-*-*-*-*-*-*-iso8859-1",
    NULL, "Helvetica-Narrow-BoldOblique", 0},
  {NULL, NULL, NULL, 0}
};
...
XtVaSetValues(table,
  XmNxrtTblPSFontMapList,fontlist,
  NULL);
```

3.18.3　プリント･コールバック

XmNxrtTblPrintCallbackリソースは、テーブルの各ページが印刷される前と後に呼び出されるコールバックのリストを指定します。これらのコールバックを使用して、印刷プロセスを監視したり、出力されるプリント･ファイルを変更することができます。このリストの各コールバックには、次の構造体が渡されます。

```
typedef struct }
  int     reason;      // Read-only
  XEvent       *event;      // Read-only
  XrtTblRange      **range;     // Read-only
  int     current_page;    // Read-only
  int     total_pages; // Read-only
  int     across_pages;    // Read-only
  int     down_pages;  // Read-only
  int     page_number;
  String       header, footer;
  FILE         *fp;
  Boolean      write_file_header;
  Boolean      write_file_trailer;
  Boolean      doit;        // initially True
} XrtTblPrintCallbackStruct;
```

reason は XRTTBL_REASON_PRINT_BEGIN(ペ ー ジ が 印 刷 さ れ る 前) ま た は 、 XRTTBL_REASON_PRINT_END(ページが印刷された後)です。*range*はそのページのセル範囲、*current_page*は印刷されるページ、*total_pages*は印刷するページ数(*across_pages* x *down_pages*)です。

この構造体の次のメンバを変更して、テーブルの印刷プロセスをコントロールすることができます。

- *header*と*footer*はページのヘッダーとフッターです。(*XrtTblVaDrawPS()*のコールによって指定されます。)
- *page_number*は、ヘッダーまたはフッターに表示されるページ番号です。
- *fp*は、テーブルが出力されるファイルです。(この値を新しいファイルに変更する場合、始めにその新しいファイルに書き込む時に、*write_file_header*をTrueにセットしてください。XRT/tableは、PostScriptファイルのヘッダーを再プリントします。) *fp*を変更することにより、例えば、奇数ページだけを別のファイルに書くというようなことが実行できます。
- ファイルの最終ページが書かれている時、そのページがテーブルの最終ページではない場合、*write_file_trailer*をTrueにセットする必要があります。これは、XRT/tableにPostScriptファイルtrailerを書くように伝えます。*write_file_trailer*は、*current_page*が*total_page*(プリントされているページがテーブルの最終ページ)であれば、自動的にTrueにセットされます。
- *doit*は、カレント・ページを印刷するかどうか指定します。ページをスキップするには、*reason*がXRTTBL_REASON_PRINT_BEGINの時に、*doit*をFalseにセットしてください。(*doit*の設定は、*current_page*または*total_pages*には影響ありません。)

Print Widgetのコールバック

XmNxrtTblPrintWidgetCallbackリソースを使用して、セル/ラベル内にあるウィジェットが印刷される前と後に呼び出されるコールバックのリストを定義できます。

これを利用して、サイズが大きかったり、または複雑なウィジェットの印刷をスキップすることができます。このリストの各コールバックには、次の構造体が渡されます。

```
typedef struct {
  int reason;      // Read-only
  XEvent  *event;      // Read-only
  Widget  widget;      // Read-only
  int row, column; // Read-only
  FILE    *fp;         // Read-only
  int current_page;    // Read-only
  int page_number; // Read-only
  Boolean doit;        // initially True
} XrtTblPrintWidgetCallbackStruct;
```

reason は XRTTBL_REASON_PRINTWIDGET_BEGIN(印刷前)または、XRTTBL_REASON_PRINTWIDGET_END(印刷後)です。*widget*は印刷されるウィジェットを指定し、*row*と*column*はそのウィジェットのある位置を示します。ウィジェットを印刷しない場合は、*doit*をFalseにセットします。

この構造体の他のメンバは、(前の項で説明されている)Printコールバックと同じように動作します。PrintWidgetコールバック内からは、メンバを変更することはできません。

4

XRT/tableデータ

XrtTblCellValues構造体 ■ テーブルにデータを取り込む
テーブル・データの操作 ■ 複数のXrtTblCellValues構造体の使用
セル値のコールバック ■ テーブルからデータを取り出す

XrtTblCellValues構造体の詳細と、この構造体の割り当てや操作法を解説します。

4.1 XrtTblCellValues構造体

XrtTblCellValues構造体は、大きな表形式のデータを効率的に処理できるように設計された、ダイナミックに管理される複雑なデータ構造体です。

XRT/tableが各セルを描画または再描画する時[1]、セル値は、XmNxrtTblCellValuesに指されているXrtTblCellValues構造体から取り出されます。(このポインタとポインタに指されている構造体は、*read-only*です。) 構造体がなかったり、構造体にセルの値が存在しない場合は、そのセルは空白になります。[2]

プログラムまたはユーザによってセル値が入力されると、XRT/tableは自動的にXrtTblCellValues構造体をアップデートし、維持します。

この構造体はテーブルの属性には結び付いていません。XmNxrtTblCellValuesは、表示されている属性(セルの背景色やフォント等)に影響を与えずに、幾つかのXrtTblCellValues構造体を表示するようにセットできます。図48は、セル値とテーブルの関係を示しています。

[1] XRT/tableは、スクロールまたはエクスポーズ・イベントの結果、セルを描画します。

[2] セル値の関数が定義されている時は、その関数を呼び出して空白のセルに値を表示します。



図48 XrtTblCellValues構造体とテーブル

構造体の定義

XrtTblCellValues構造体は、次のように定義されます。

```
typedef struct {
  int          rows;
  XrtTblXmStringSet set_mode;
  XrtTblXmStringList    **values:
} XrtTblCellValues;
```

*values*は値のリスト。*set_mode*は*values*がXmStringsまたはStringsのどちらの形式かを示します。これは、XRTTBL_SET_STRINGまたはXRTTBL_SET_XMSTRINGのどちらかです。XrtTblCellValues構造体を作成する手続きや方法等が*set_mode*をセットします。デフォルトでは、*set_mode*は、XRTTBL_SET_STRINGにセットされています。

簡易マクロ

<*Xm/XrtTable.h*>には、XrtTblCellValues構造体の部分(とテーブルの他の部分)へのアクセスを手助けするマクロがあります。

CELL_VALUEは、XrtTblCellValues構造体からセル値を取り出します。その他のマクロは、セルが値を含むかどうか、contextがセルかラベルか等を調べたりすることができます。Appendix Cにマクロのリストがありますので参照してください。

簡易手続き

XRT/tableには、XrtTblCellValues構造体を操作する手続きや方法が提供されています。これらはAppendix Bに解説があり、その一部のソースコードが、*$XRTHOME/src/table/util*にあります。

XrtTblAddColumns() *XrtTblAddRows()*	テーブルのXrtTblCellValues構造体に新しい行/列を挿入し、XmNxrtTblNumRowsとXmNxrtTblNumColumnsを増加させます。既存の行/列のセル値と一緒に属性も維持します。
XrtTblDeleteColumns() *XrtTblDeleteRows()*	テーブルのXrtTblCellValues構造体から行/列をその属性と共に削除し、XmNxrtTblNumRowsとXmNxrtTblNumColumnsを減少させます。
XrtTblMoveColumns() *XrtTblMoveRows()*	XrtTblCellValues構造体内で行または列を移動します。既存の行/列のセル値と一緒に属性も維持します。
XrtTblGetCell()	テーブルから値を取得したり、または、セル値の関数やCellValueコールバックの結果を取り出します。XmNxrtTblXmStringSetModeがXRTTBL_SET_STRINGであれば、ストリングが戻ります。
XrtTblGetCellValues()	新しいXrtTblCellValues構造体を作成し、テーブルのXrtTblCellValues構造体からコピーされた値をロードします。
XrtTblSetCell()	セル値をセットします。(オプションで、値を検証したり、セルを再描画させることができます。) Motif 1.2では、StringまたはXmStringが使用できます。XmNxrtTblXmStringSetModeをセットして、Stringを指定することができます。これは、一つのセルを最も早くアップデートする方法ですので、セルのリサイズは行われません。
XrtTblSetCellValues()	提供されたXrtTblCellValues構造体からテーブルのXrtTblCellValues構造体へセル値をコピーします。
XrtTblCellValuesCopy()	セル値の構造体のコピーを作成します。
XrtTblSortTable()	テーブルのXrtTblCellValues構造体の行を列に基づいてソートします。XmNxrtTblSortCallbackリソースを使用して、*XrtTblSortTable()*の後に呼び出されるコールバックを指定することができます。

4.2 テーブルにデータを取り込む

XRT/tableの開発において、表示するデータを取得し、テーブルに表示するためにそのデータを`XrtTblCellValues`構造体に挿入することが最も難しい作業かもしれません。

ファイルのデータ(スタティック)

データがファイル形式になっている場合、*XrtTblReadAscii()*を使用して`XrtTblCellValues`構造体を作成し、値を読み込んでください。この手続きは、値がカンマで区切られたフォーマットになっているデータを読み込みます。(詳細はAppendix Bを参照してください。また、図3にも使用例があります。) この手続きのソースは、*$XRTHOME/src/table/util*にあります。使用するデータのフォーマットに合わせて、このルーチンのコピーを修正することもできます。

注意: *XrtTblReadAscii()*を使用する際は、`XmNxrtTblNumRows`と`XmNxrtTblNumColumns`をセル値の構造体のサイズにセットしてください。セル値の構造体は、テーブルの行/列の数には依存していません。次のコードは、その例です。

```
int last_row, last_column;
...
values = XrtTblReadAscii(toplevel, fp, ',');
XrtTblGetCellValuesSize(values, &last_row,
    &last_column, NULL);
table = XtVaCreateManagedWidget("table",
    xtXrtTableWidgetClass, toplevel,
    XmNxrtTblNumRows, last_row + 1,
    XmNxrtTblNumColumns, last_column + 1,
    ...
    NULL);
```

もう一つの方法として、`XmNxrtTblCellValueContext`を使用して、ウィジェットが作成された後に、`XrtTblCellValues`構造体をセットすることができます。XRT/tableは、ウィジェットが作成された時に構造体がない場合、`XrtTblCellValues`構造体を自動的に割り当て、作成します。

変化するデータ(ダイナミック)

ダイナミックに変化するデータの場合は、注意が必要です。Xt Intrinsicには、ランタイム・イベントを処理するコールバックの手続きが提供されています。特に、*XtAppAddInput()*は、ファイル記述子にペンディングされている入力がある時に呼ばれる、アプリケーションのコールバック・ルーチンの登録に使用できます。ファイル記述子は、ファイル、パイプ、または他のマシンで実行されている処理のソケット・コネクションとすることができます。コールバック・ルーチンは、`XmNxrtTblCellValueContext`を使用して、このデータをテーブルの`XrtTblCellValues`構造体に加えることができます。

他には、アプリケーションが*XtAddTimeOut()*を使用してインターバル・タイマーを定義する方法があります。指定されたインターバルが切れる度に、アプリケーションのコールバック・ルーチンが実行されます。この時、コールバック・ルーチンは、追加するデータがあるか調べ、`XmNxrtTblCellValueContext`を使用してそのデータをテーブルに追加することができます。

4.3 テーブル･データの操作

ここでは、XmNxrtTblCellValuesで指されているXrtTblCellValues構造体のデータの操作について解説します。

4.3.1 セル値の設定と取得

セル値の設定

現在のXrtTblCellValues構造体のセル値を指定するには、次の例のようにcontextとXmNxrtTblCellValueContextリソースをセットします。

```
String values[] =
  { "value 1", "value 2", "a multi-\nline value", NULL };
for (i=0; values[i]; i++)
  XtVaSetValues(table,
    XmNxrtTblContext,       XrtTblSetContext(20, i),
    XmNxrtTblCellValueContext, values[i],
    NULL);
```

デフォルトでは、XmNxrtTblCellValueContextのデータ型はXmStringですが、Stringも指定できます。XRT/tableは内部で自動的にStringをXmStringに変換します。

ひとつのセルを最も早くアップデートするには、*XrtTblSetCell()*を使用してください。この方法は、指定されたセルの値をセットし、オプションで値を検証したり、セルを再描画したりします。XmNxrtTblCellValueContextと同様に、*XrtTblSetCell()*もXmString型を必要としますが、String型も使用することができます。尚、このルーチンは、XmNxrtTblXmStringSetModeがセットされていても、新しいデータに対応するようにセルをリサイズしません。

セル(とラベル)をセットする時に使用される値の型を指定するには、XmNxrtTblXmStringSetModeリソースを使用します[1]。このリソースは、次の値にセットできます。

XRTTBL_SET_EXAMINE　値はXmStringまたはString(デフォルト)[2]

XRTTBL_SET_XMSTRING　値はXmString

XRTTBL_SET_STRING 値はString

XmNxrtTblXmStringSetModeは、テーブルの作成時にのみセットできます。

ユーザによる編集

ユーザがセル値を編集すると、前のセル値と置き換えられます。セルの編集を不可に設定しない限り、ユーザは、アプリケーションでセットされているどのセル値も変更することができます。インタラクティブなセルの編集については、5.7を参照してください。

[1] このリソースは、行/列ラベルのリスト(XmNxrtTblRowLabelsとXmNxrtTblColumnLabels)と、セル/ラベル値のコールバック(XmNxrtTblCellValueCallbackとXmNxrtTblLabelCreateCallback)の型も決定します。

[2] Motif 2.xを使用している場合は、XmNxrtTblXmStringSetModeはXRTTBL_SET_EXAMINEにセットできません。

セル値の取得

テーブルのXrtTblCellValues構造体からセル値を取り出すには、次の二つの方法があります。

■ シングル・セルの値(カレントcontextは、一つのセル)を取り出すには、XmNxrtTblCellValueContextを使用する。

```
XmString value1;
XtVaSetValues(table,
  XmNxrtTblContext, XrtTblSetContext(0,0),
  NULL);
XtVaGetValues(table,
  XmNxrtTblCellValueContext, &value1,
  NULL);
value1 = XmStringCopy(value1);
```

または、

■ *XrtTblGetCell()*を使用して指定されたセルから値を取り出すか、または、XmNxrtTblCellValueFuncかXmNxrtTblCellValueCallbackの結果を取得する。

```
int row=1,col=1;
XmString value1;
value1=XrtTblGetCell(table,row,col);
value1=XmStringCopy(value1);
```

または、

■ CELL_VALUEを使用して、ある位置のセル値を取り出す。

```
XrtTblCellValues *cells;
String string;
int i, j, rows, columns;
XmString value;

XtVaGetValues(table, XmNxrtTblCellValues, &cells, NULL);
XrtTblGetCellValuesSize(cells, &rows, &columns, NULL);
for (i=0; i <= rows; i++) {
  for (j=0; j <= columns; j++) {
    if ((value = CELL_VALUE(cells, i, j))) {
      string = XrtTblCvtXmStringToString(value));
      printf("%d %d: %s\n", i, j, string);
      XtFree(string);
    }
  }
}
```

デフォルトでは、セル値はXmStringとして取り出されます。Stringとしてセル値を取り出すには、XmNxrtTblXmStringGetModeをXRTTBL_GET_STRINGにセットします。

値の型を指定

XmNxrtTblXmStringGetModeリソースを使用して、セル(とラベル)から取り出される値、及び、行と列ラベルのリストの型を指定することができます。このリソースは次の値にセットすることができます。

XRTTBL_GET_XMSTRING　XmStringとして取り出す(デフォルト)

XRTTBL_GET_STRING Stringとして取り出す

注意: Motif 2.xにリンクしている場合、XmNxrtTblXmStringSetModeをXRTTBL_SET_EXAMINEにセットすることはできません。アプリケーションが、このようにセットしようとするとエラー・メッセージがプリントされ、そのアプリケーションは終了してしまいます。

4.3.2 行と列の追加

*XrtTblAddColumns()*と*XrtTblAddRows()*は、テーブルに新しい行と列を追加し、(オプションで)新しいセルに値をセットします[1]。*XrtTblAddColumns()*は、既に存在しているセル値と属性を右に、*XrtTblAddRows()*は、それらを下にシフトします。シフトされる行や列と一緒にラベルもシフトするかどうかを指定できます。XmNxrtTblNumRowsまたはXmNxrtTblNumColumnsは、挿入が行われた場合はアップデートされます。

*XrtTblAddColumns()*と*XrtTblAddRows()*を使用する時は、次の点に注意してください。

- 新しいセルに値を提供しない時、これらのセルは空白です。新しい行や列のラベルの値は、別に指定してください。挿入を開始する行または列は、XmNxrtTblNumRowsまたはXmNxrtTblNumColumnsより大きくできません。
- 可変の高さ/幅を持つ新しい行または列に値が指定されていないと、行/列は表示されません。(存在しますが、隠れています。)可変の行/列は、3.7に解説されています。
- カレント・セルが、挿入が開始される行/列またはその下にある場合は、セルの編集はキャンセルされ、XmTextウィジェットはアンマップされます。
- 値は、XmStringsの配列またはStringsの配列として指定されます。XmNxrtTblXmStringSetModeは、どちらが使用されるかを指定します。

次は、*XrtTblAddColumns()*のプロトタイプです。

```
Boolean
XrtTblAddColumns(
  Widget  table,
  int position,
  int num_columns,
  Boolean shift_labels,
  void    **values,
  int num_values
)
```

Appendix Bに、手続きについての解説があります。次の例は、列0の前に5列挿入し、一番めの列を三つの値でセットします。

```
String values[] = { "a", "b", "c" };
XrtTblAddColumns(table, 0, 5, False, values, 3);
```

[1] 変更する前に構造体のコピーを保管するには、*XrtTblCellValuesCopy()*を使用します。

4.3.3 行と列の削除

XrtTblDeleteColumns() と *XrtTblDeleteRows()* は、テーブルから行と列を削除します。*XrtTblDeleteColumns()*は、既に存在するセル値と属性を左に、*XrtTblDeleteRows()*はこれらを上にシフトします。シフトされる行や列と一緒にラベルもシフトするかどうかを指定できます。XmNxrtTblNumRowsまたはXmNxrtTblNumColumnsは、削除が行われた場合は、アップデートされます。

*XrtTblDeleteColumns()*と*XrtTblDeleteRows()*を使用する時は、次の点に注意してください。

- 削除が開始される行または列は、XmNxrtTblNumRowsまたはXmNxrtTblNumColumnsより大きくできません。
- カレント・セルが、削除が開始される行/列またはその下にある場合は、セルの編集はキャンセルされ、XmTextウィジェットはアンマップされます。
- XmNxrtTblNumRowsまたはXmNxrtTblNumColumnsを減少させても、テーブルから行または列は削除されません。これらは、スクロール可能な領域を小さくするだけです。

次は、*XrtTblDeleteColumns()*のプロトタイプです。

```
Boolean
XrtTblDeleteColumns(
  Widget  table,
  int position,
  int num_columns,
  Boolean shift_labels
  )
```

Appendix Bにこれらの手続きの解説があります。次の例は、列2から列5を削除し、行と一緒に行ラベルを上にシフトします。

```
XrtTblDeleteRows(table, 2, 4, True);
```

4.3.4 行と列の移動

*XrtTblMoveColumns()*と*XrtTblMoveRows()*は、テーブル内で行と列(オプションでラベルも)移動します。*XrtTblMoveColumns()*と*XrtTblMoveRows()*は、移動によって影響されたセル値と属性をシフトします。移動される行または列と一緒にラベルも移動するかどうかを指定できます。

移動元と移動先の行または列は、対応するXmNxrtTblNumRowsまたはXmNxrtTblNumColumnsの値より大きくすることはできません。

カレント・セルが、移動によって影響を受ける行/列またはその下にある時は、セルの編集はキャンセルされ、XmTextウィジェットは、アンマップされます。次は、*XrtTblMoveColumns()*のプロトタイプです。

```
Boolean
XrtTblMoveColumns(
  Widget  table,
  int source,
  int num_columns,
  int destination,
  Boolean shift_labels
  )
```

Appendix Bにこの手続きについての詳細があります。次の例は、ラベルをシフトせずに、行4から行6を移動し、行1の前から始まるようにします。

```
XrtTblMoveRows(table, 4, 3, 1, False);
```

4.3.5 セル値をクリアする

セル、行、列またはテーブル全体のセル値は、そのcontextのXmNxrtTblCellValueContextをNULLにすることにより、簡単にクリアすることができます。セルの視覚的な属性は、変更されません。

4.3.6 テーブル全体をクリアする

テーブル全体をクリアするには、XmNxrtTblCellValuesをNULLにセットします。XmNxrtTblNumRowsとXmNxrtTblNumColumnsリソースや、そのテーブルに定義されている属性は変更されません。

4.4 複数のXrtTblCellValues構造体の使用

XRT/tableでは、テーブルの視覚的な属性を変更せず、全てのセル値を変更することができます。アプリケーションで幾つかのXrtTblCellValues構造体を作成し、XmNxrtTblCellValuesをそれらの構造体の一つを指すようにセットして切り替えることができます。

テーブルのセル値を、異なるXrtTblCellValues構造体で置き換えるには、はじめに次の方法で新しい構造体を作成します。

- *XrtTblReadAscii()*を呼び出して、新しいXrtTblCellValues構造体を作成し、ファイルの値をロードする
- *XrtTblGetCellValues()*を呼び出して、新しいXrtTblCellValues構造体を作成し、テーブルの値をロードする
- *XrtTblGetGraphData()*または*XrtTblGet3dData()*を呼び出して、新しいXrtTblCellValues構造体を作成し、XRT/graphまたはXRT/3dのデータセットから変換された値をロードする

次に、XmNxrtTblCellValuesリソースを新しく作成された構造体を指すようにセットします。XRT/tableがその構造体を内部メモリにコピーした後は、使用したコピーは破棄することができます。次のコードは、その例を示しています。

```
XrtTblCellValues *values;
...
values = XrtTblReadAscii(toplevel, fp, ',');
XtVaSetValues(table,
  XmNxrtTblCellValues, values,
  NULL);
XrtTblDestroyCellValues(values);
```

4.5 セル値のコールバック

XrtTblCellValues構造体を使用してテーブルのセル値を記憶する代わりに、CellValueコールバックを定義することができます。特に大きなテーブルでは、この方法を使用した方が良いパフォーマンスが得られます。

XmNxrtTblCellValueCallbackリソースは、XRT/tableが空白のセルを最初に表示する時にランタイムに呼び出すコールバック・ルーチンを指定します。セル値の追加、削除、移動、ソートは、アプリケーションの役割です。

作成したコールバックは、返されたXmString(XRT/tableによって解放されます)を割り当てたり、スタティック・ポインタを渡すようにセットできます。このコールバックには、次の構造体が渡されます。

```
typedef struct {
  int reason;      // Read-only
  XEvent  *event;      // Read-only
  int row, column; // Read-only
  XmString    string;
  Boolean store;       // initially False
  Boolean free_string; // initially True
} XrtTblCellValueCallbackStruct;
```

次のコードは、簡単なCellValueコールバックの例です。

```
static void
cellvalueCB(Widget table, XtPointer data,
  XrtTblCellValueCallbackStruct *cbs)
{
  static char buf[30];
  sprintf(buf, "Row: %d, Column: %d",
    cbs->row, cbs->column);
  cbs->string = (XmString) buf;
  cbs->free_string = False;
}
```

この例では、セル値の記憶に静的バッファを使用し、XrtTblCellValueCallbackStructの*free_string*メンバをセットして、ストリングの使用が終わってもセル値を解放しないように指示しています。

このコールバックをCellValueコールバック・リストに追加するには、*XrtTblAddCallback()*を呼び出します。

```
XrtTblAddCallback(table, XmNxrtTblCellValueCallback,
  CellValueCB, NULL);
```

コールバック構造体の*store*メンバによって、ストリングが内部的に記憶されるかどうかを指定できます。*store*がTrueにセットされていると、値は記憶され、コールバックはもう呼び出されません。デフォルトでは、*store*はFalseで値は記憶されません。

デフォルトでは、*string*はXmStringまたはStringです。[1]どちらかに指定するには、XmNxrtTblXmStringSetModeをセットしてください。

[1] Motif 2.*x*にリンクしている場合は、*string*はXmStringsとStringsの両方を使用することができません。XmNxrtTblXmStringSetModeによってどちらかの型を指定してください。

ユーザによる編集

デフォルトでは、ユーザはCellValueコールバックによって作成された値を編集できません。編集を可能にするには、ValidateCellコールバックを使用して、編集されたセルをXRT/tableが破棄する前に捕らえ、別に格納してください。次のコードは、CellValueコールバックによって生成されたテーブルからユーザによる入力を取り出し、テーブルをアップデートしています。

```
static void
validateCB(Widget table, XtPointer data,
      XrtTblValidateCellCallbackStruct *cbs)
{
      if (!cbs->value_changed)
            return;
      XtFree(values[cbs->row][cbs->column]);
      values[cbs->row][cbs->column] =
            XtNewString(cbs->value);
}
```

ValidateCellコールバック・ルーチンは、ユーザによるセルの編集を構造体に格納し、この構造体がテーブル・データのセットに使用されます。この場合、XmNxrtTblCellValueContextは使用できません。

4.6 テーブルからデータを取り出す

ASCII出力

*XrtTblWriteAscii()*は、指定した範囲のセル値を区切りのあるASCIIファイルに書きます。このファイルは、スプレッドシートやデータベースのアプリケーションにエクスポートすることができます。この方法は、ラベル値を出力しません。範囲を指定しない場合は、空白でないセルは全て出力されます。この手続きはAppendix Bに解説されています。

SYLK出力

*XrtTblWriteSylk()*は、指定した範囲のセル値をMicrosoftのSymbolic Link(SYLK)フォーマットのファイルに書きます。SYLKを利用すると、セル値を一般のスプレッドシートにインポートさせることができます。範囲を指定しない場合は、空白でないセルを全てSYLKファイルに書き出します。この手続きはAppendix Bに解説されています。

5

ユーザ･インタラクションのプログラミング

デフォルトのユーザ･インタラクション
ユーザ･インタラクションとトランスレーションのカスタマイズ
リスト表示モード ■ テーブルのスクロール ■ 行と列のリサイズ
セルのトラバース ■ セルの編集 ■ セルの選択
マウス･ポインタ ■ ドラッグ&ドロップ

ここでは、XRT/tableのユーザ･インタラクション機能(ユーザがどのようにテーブルとインタラクションを行えるか、またアプリケーションがどのようにそれをコントロールするか)を解説します。

5.1 デフォルトのユーザ･インタラクション

デフォルトで、エンドユーザは、テーブルのスクロール、各行/列のリサイズ、セルからセルへのトラバース、テキストのドラッグ&ドロップ、セル値の編集を行うことができます。また、設定により、セルを選択することもできます。図49は、XRT/tableのデフォルトのトランスレーションを使用したインタラクションの例です。

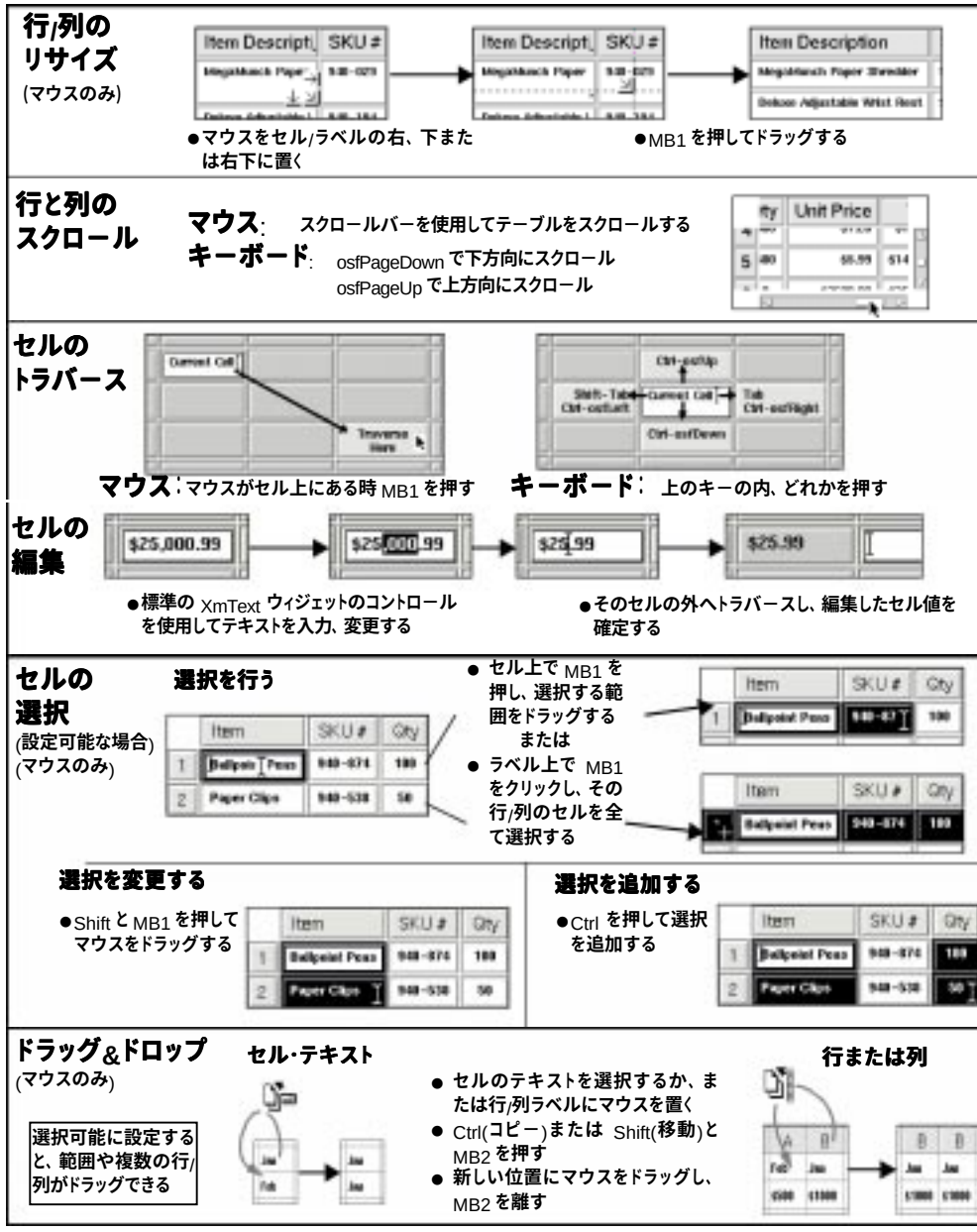


図49　XRT/tableのデフォルトのユーザインタラクション

5.2 ユーザ・インタラクションとトランスレーションのカスタマイズ

XRT/tableのデフォルトのトランスレーション・テーブルは、ウィジェットでインタラクティブなアクションを起こすユーザ・イベントを定義します。次の方法で、このユーザ・インタラクションをカスタマイズすることができます。

- **トランスレーション・テーブル**: アプリケーションは、デフォルトのトランスレーション・イベントまたはアクションを変更したり、新しいアクションのトランスレーションを追加することができます。この方法は、この章で解説され、Appendix Dには、XRT/tableのデフォルトのトランスレーションとアクションのリストがあります。
- **コールバック**: ユーザ・インタラクションの開始前、実行中、終了後に呼び出されるコールバック・ルーチンを登録することにより、アプリケーションはユーザがウィジェットとインタラクションを行った時に通知を受けることができます。コールバック・ルーチンは、各インタラクションを不許可にしたり制限することができます。コールバックの使用法については、各インタラクションの解説を参照してください。

トランスレーションの設定

既存のトランスレーションのイベントと(または)アクションを変更したり、新しいトランスレーションをテーブルに追加するには、テーブルとそれに管理されるarea-clippingウィジェット(表示部分を切り取る子ウィジェット)のトランスレーションをセットする必要があります。(テーブルの子ウィジェットについては、2.3を参照してください。) `XmNxrtTblClipChildren`は、これらの全てのウィジェットのリストを指定します。リストの最後のエレメントは、`NULL`にセットされていなければなりません。次の例は、テーブルとclipウィジェットに新しいトランスレーションを追加しています。

```
Widget *child;
int i;
static XtActionsRec actions[] = {
  {"DisplayMenu", (XtActionProc) DisplayMenu }
};
static char trans[] = "<Btn3Down>: DisplayMenu()";
...
XtAppAddActions(XtWidgetToApplicationContext(table),
     actions, 1);
XtVaGetValues(table, XmNxrtTblClipChildren, &child, NULL);
XtOverrideTranslations(table,
     XtParseTranslationTable(trans));
for (i=0; child[i]; i++)
  XtOverrideTranslations(child[i],
    XtParseTranslationTable(trans));
```

注意: トランスレーションを正しく設定するには、次のガイドラインを参照してください。

- トランスレーションの設定には、リソースファイルを使用しない。
- ウィジェットの作成時には、トランスレーションをセットしない。
- トランスレーションを設定する時は、常に*XtOverrideTranslations()*を使用する。

トランスレーションを不可にする

ある特定のインタラクションを不可にするには、そのデフォルトのトランスレーションが*NoAction()*ルーチンを呼ぶように置き換えると簡単です。

新しいアクションの定義

アクション手続きを作成する時は、どのセルまたはラベルでイベントが起こったかを知っておくと便利です。*XrtTblEventToRowCol()*を呼び出してこれを知ることができます。(この手続きは、Appendix Bに解説されています。)

ダブル・クリック

XRT/tableは、シングル・クリックでトラバースやセルのリサイズ、選択、編集を指示させるので、複数のクリックの処理には、特別なコーディングが必要です。ユーザがセル上でクリックする度に(XmNmultiClickTime の制限内)、XRT/table は XrtTblSelectCallbackStruct構造体の*click_count*メンバをアップデートします。アプリケーションは、Selectコールバックを使用して、セルまたはラベル上の各クリックの通知を受けることができます。次のSelectコールバックは、ユーザが列ラベルでダブル・クリックを行った時には、ソートの関数を呼び出します。この例を使用する時は、選択の方法は、_RANGE または_MULTIRANGEにセットしてください。

```
selectCB(Widget table, XtPointer data,
XrtTblSelectCallbackStruct *cbs)
{

  /* Allow double-clicks */
  if (IS_COLUMNLABEL(cbs->row, cbs->column)) {
    if (cbs->reason == XRTTBL_REASON_UNSELECT) {
      if (cbs->click_count > 1)
        cbs->doit = False;
    }

  /* Double click on a column label - sort column */
    else if (cbs->reason == XRTTBL_REASON_SELECT_END
        && cbs->click_count == 2)
    sortCB();
  }
}
```

5.3 リスト表示モード

XRT/tableでは、テーブルがXmListウィジェットのように表示、操作されるように指定することができます。XmNxrtTblModeリソースをXRTTBL_MODE_LISTにセットしてください。テーブルがリスト表示モードの場合、セルが選択されると、その行全体が選択されます。

注意: リスト表示モードの場合、既に選択されている行でセルを選択しようとすると、行全体が選択解除されます。XmListと同様に、XRT/tableでもCtrlキーを押しながら、選択された行をクリックすることにより、(他の選択された行に影響を与えずに)その行の選択をオン、オフにすることができます。

XmListウィジェットの動作と表示を類似させるには、次のようにしてください。

- context(ALL, ALL)のXmNxrtTblTraversableContextをFalseにセットして、全てのセルをトラバース不可に指定する。
- XmNxrtTblBorderTypeContextをXRTTBL_BORDER_NONEにセットしてセルの罫線を削除する。

■ XmNxrtTblSelectionPolicyリソースをXRTTBL_SELECT_SINGLEにセットし、ユーザが一度に一行しか選択できないようにする。

デフォルトでは、XmNxrtTblModeはXRTTBL_MODE_TABLEにセットされており、テーブルは通常通り操作できるようになっています。

5.4 テーブルのスクロール

テーブルのサイズがウィンドウより大きい場合、エンドユーザは、マウスやキーボードを使用してテーブルをスクロールすることができます。XRT/tableは、マウスによるスクロールには二つのXmScrollBarウィジェット(縦と横)を作成、コントロールし、キーボードによるスクロールには*Scroll()*アクション・ルーチンを使用します。

XmNxrtTblTopRowとXmNxrtTblLeftColumnリソースは、テーブルがスクロールされるとダイナミックにアップデートされます。

XRT/tableは、セルのトラバースや選択等の他のインタラクションによってもテーブルをスクロールします。スクロールをしてもカレント・セルの位置は変わりません。

5.4.1 スクロールバー

アプリケーションは、スクロールバーをどのようにウィジェットに接続するか、いつ表示するか、どこに配置するか等をコントロールできます。また、スクロールバー自身の動きもコントロールすることができます。

スクロールバーの接続

テーブルにどのようにスクロールバーが接続されるかは、テーブルのスタイルに依存します。*Standard-style*のテーブルは、スクロールバーをセル/ラベルに接続し、表示領域のサイズの変化に応じて移動します。*XmScrolledList-style*のテーブルは、スクロールバーをテーブル・ウィジェット自身に接続し、表示領域のサイズが変化しても移動しません。(ウィジェットのサイズの変更があった場合にのみ、スクロールバーは移動または自身のサイズを変更します。)

横向きのスクロールバーの接続は、XmNxrtTblHorizScrollBarPosition、XmNxrtTblHorizScrollBarLeftAttachment、XmNxrtTblHorizScrollBarRightAttachmentの三つのリソースで指定します。

縦向きのスクロールバーの接続は、XmNxrtTblVertScrollBarPosition、XmNxrtTblVertScrollBarTopAttachment、XmNxrtTblVertScrollBarBottomAttachmentの三つのリソースで指定します。

*standard-style*のテーブルのスクロールバーを指定するには、次のようにしてください。

■ PositionとAttachmentのリソースをデフォルト値のままにする。

*XmScrolledList-style*のテーブルのスクロールバーを指定するには、次のようにします。

■ 縦と横のPositionリソースをXRTTBL_SBPOSITION_SIDEにセットする。

■ 左右、上下のAttachmentリソースをXRTTBL_ATTACH_SIDEにセットする。

- XmNxrtTblDisplayHorizScrollBar と XmNxrtTblDisplayVertScrollBar を XRTTBL_DISPSB_ALWAYSにセットする。
- XmNxrtTblSpaceを0にセットする。

XmNxrtTblSpaceは、テーブルとスクロールバーの間のオフセットを指定します。このオフセットは、通常、スクロールバーとテーブルのセル/ラベルの間に適用されますが、スクロールバーがウィジェットに接続されている時は、スクロールバーとウィジェットの間のオフセットに適用させることができます。(但し、最後の行/列とウィジェットの端の間にスペースがある場合にのみ)

スクロールバーの表示

デフォルトでは、テーブルがスクリーン上で表示されている行/列の数より大きくなった時に各スクロールバーを表示します。スクロールバーを常に表示させるには、XmNxrtTblDisplayHorizScrollBarと(または)XmNxrtTblDisplayVertScrollBarをXRTTBL_DISPSB_ALWAYSにセットします。XRTTBL_DISPSB_NEVERにセットすると、スクロールバーは、表示されません。[1]

スクロールバーの位置

各スクロールバーは、XmNxrtTblHorizScrollBarPlacement と XmNxrtTblVertScrollBarPlacementリソースを使用して配置させることができます。図50は、両方のスクロールバーの位置を変更した場合の例です。



図50　デフォルトと反対に設定した場合のスクロールバーの位置

ジャンプ･スクロール

スクロールバーの動きは、XmNxrtTblJumpScrollで指定できます。デフォルト(XRTTBL_JUMP_VERTICAL)で、縦のスクロールバーは、一回に一行分づつテーブルを“jump-scroll”します。両方のスクロールバーでjump-scrollさせるには、このリソースをXRTTBL_JUMP_ALLにセットしてください。テーブル上をスムーズにスクロールさせる場合は、XRTTBL_JUMP_NONEにセットします。横向きのスクロールバーでjump-scrollさせるには、XRTTBL_JUMP_HORIZONTALにセットしてください。

プレビュー･スクロール

XmNxrtTblPreviewScrollingがアクティブな場合、スクロールバーのつまみの横に小さなウィンドウが表示されます。このウィンドウには、スクロールバーでのドラッグが終了した(スクロールバーつまみが放された)時にテーブルの左上に表示される行/列を表示します。

[1] スクロールバーが非表示でも、キーボードを使用してスクロールすることができます。インタラクティブなスクロールを不可にするには、“インタラクティブなスクロールを不可に設定”の項を参照してください。

このウィンドウの背景/前景色や罫線、フォント等の属性は、*XrtTblSetPreviewProperties()*または*XrtTblGetPreviewProperties()*を使用してセットできます。

XmScrollBarの属性

どちらのスクロールバーのリソースも設定したり取り出すことができます。XmNxrtTblHorizontalScrollBarは横向きのスクロールバーのウィジェットID、XmNxrtTblVerticalScrollBarは縦向きのスクロールバーのウィジェットIDを指定します。次のコードは、縦向きのスクロールバーの背景色をセットします。

```
Widget scrollbar;
XtVaGetValues(table,
  XmNxrtTblVerticalScrollbar, &scrollbar,
  NULL);
XtVaSetValues(scrollbar,
  XtVaTypedArg, XmNbackground,
  XmRString, "red", strlen("red") + 1,
  NULL);
```

スクロールバーのリソースは、リソース・ファイルでセットすることもできます。スクロールバーの名前は、"vertScroll"と"horizScroll"です。例えば、アプリケーションの全てのテーブルの横向きのスクロールバーを指定するには、"*.XtXrtTable.horizScroll"としてください。リソース・ファイルでのリソースの指定法は、Appendix Gに解説されています。

XmScrollBarウィジェットのプログラミングに関する詳細は、Motif関連の資料を参照してください。

5.4.2 スクロールのプログラミング

インタラクティブなスクロールを不可に設定

スクロールは、縦横どちらの方向にも不可に設定することができます。マウスとキーボードのスクロールは、それぞれ別に設定してください。

- **マウス**: XmNxrtTblDisplayHorizScrollBarと(または)XmNxrtTblDisplayVertScrollBarをXRTTBL_DISPSB_NEVERにセットして、スクロールバーを画面から削除する。
- **キーボード**: 次のコードのように、スクロールのトランスレーションが*NoAction()*ルーチンを呼ぶように置き換える。

```
*.XtXrtTable.translations: #override \n\
  <Key> osfPageDown:    NoAction()
  <Key> osfPageUp:      NoAction()
```

スクロールを完全に不可にするには、アプリケーションでもユーザが表示領域外のセルを選択したり、トラバースができないようにしてください。

スクロールの強制

アプリケーションは、次の方法でテーブルのスクロールを強制することができます。

- 特定の行を表示の一番上にスクロールさせるには、XmNxrtTblTopRowをその行にセットします。また、特定の列を表示の一番左側にスクロールさせるには、XmNxrtTblLeftColumnをその列にセットします。
- セル/ラベルが表示されているかを知るには、*XrtTblIsCellVisible()*を呼び出す。

■ 特定のセルを表示するようにするには、XrtTblMakeCellVisible()を呼び出す。

5.4.3 スクロール・コールバック

アプリケーションは、エンドユーザまたはアプリケーションによって行われたスクロールの通知を受けることができます。XmNxrtTblScrollCallbackリソースを使用して、テーブルがスクロールした時に呼び出される手続きのリストを定義することができます。リストの各コールバックには、次の構造体が渡されます。

```
typedef struct {
  int               reason;       // Read-only
  XEvent            *event;      // Read-only
  unsigned char     direction;
  int               new value;
  String            preview_text; //Read-Write
  XmScrollBarCallbackStruct *sb_cbs; // Read-only
} XmNxrtTblScrollCallback;
```

Scrollコールバックは、スクロールバーのイベントが起こる前と後に呼び出されます。(ユーザがスクロールバーのつまみをドラッグしている時は、このイベントは繰り返して起こることがあります。) *reason*メンバは、XRTTBL_REASON_SCROLL_BEGIN(セルが再描画される前)または、XRTTBL_REASON_SCROLL_END(セルが再描画された後)のどちらかです。*new_value*はXmNxrtTblTopRowまたはXmNxrtTblLeftColumnの新しい値、*preview_text*はつまみが放された時にテーブルに表示されるテキストです。値を置き換える前に、*XtFree()*は古い値を解放し、新しい値に対するメモリを割り当てます。XRT/tableは、コールバック構造体のこのエレメントによって使用されたメモリを解放します。

*sb_cbs*メンバは、XmScrollBarウィジェット自身に渡されるコールバック構造体を持っています。これには、スライダの新しい位置、選択が起こった場所が含まれています。

5.5 行と列のリサイズ

*ResizeCell()*アクションは、ユーザのインタラクティブな行/列のリサイズを可能にします。(XmNxrtTblAllowCellResizeで許可されている場合) このアクション・ルーチンは、行がリサイズされる時はXmNxrtTblPixelHeightを変更し、列がリサイズされる時はXmNxrtTblPixelWidthを変更します。ユーザは、行または列を5 pixel以下にはリサイズできません。

セルのリサイズの不許可

XmNxrtTblAllowCellResizeを用いて、テーブル全体の行/列のインタラクティブなリサイズをコントロールできます。XRTTBL_RESIZE_NONEに設定されている時は、行/列のリサイズはできません。XRTTBL_RESIZE_HORIZONTALとセットされている場合は列だけ、XRTTBL_RESIZE_VERTICALとセットされている場合は行だけリサイズできます。

XmNxrtTblPixelHeightContextまたはXmNxrtTblPixelWidthContextをセットすることにより、テーブルのリサイズに従って行/列もリサイズするように指定できます。詳細は3.7.4を参照してください。

リサイズのコントロール

ResizeCellコールバック(XmNxrtTblResizeCellCallbackで指定)を用いて、インタラクティブな行/列のリサイズをコントロールできます。各コールバックには、次の構造体のポインタが渡されます。

```
typedef struct {
  XrtTblReason     reason;      // Read-only
  XEvent           *event;      // Read-only
  String           *params;     // Read-only
  Cardinal         num_params;  // Read-only
  Int              row,column;  // Read-only
  int          current_row_height, // Read-only
  int          current_column_width, // Read-only
  int          new_row_height,
  int          new_column_width,
  Boolean      doit;
} XrtTblResizeCellCallbackStruct;
```

ResizeCellコールバックは、リサイズの開始前(*reason*はXRTTBL_REASON_RESIZE_BEGIN)、ユーザが新しい行の高さ/列幅を選択した時(*reason*はXRTTBL_REASON_RESIZE_END)、テーブルが行/列をリサイズした後(*reason*はXRTTBL_REASON_RESIZE_DONE）に呼び出されます。次の表は、各段階でコールバックが変更できるメンバです。

Reason	変更できるメンバ	コールバックでの使用
XRTTBL_REASON_RESIZE_BEGIN	doit	Falseに変更し、ユーザによるリサイズを不許可にする
XRTTBL_REASON_RESIZE_END	new_row_height new_column_width	新しい値に変更し、リサイズをコントロールする
XRTTBL_REASON_RESIZE_DONE	None - 通知だけ	

*ResizeCell()*アクションは、これらのメンバを使用します。例えば、行5のみリサイズを不許可にするには、*reason*がXRTTBL_REASON_RESIZE_BEGINで*row*が5の時、*doit*をFalseにします。次の例は、リサイズされた列の幅を10 pixelのインクレメント(倍数)にセットしています。

```
static void
cellresizeCB(Widget table, XtPointer data,
  XrtTblResizeCellCallbackStruct *cbs)
{
  switch (cbs->reason) {
  case XRTTBL_REASON_RESIZE_BEGIN:
    break;
  case XRTTBL_REASON_RESIZE_END:
    cbs->new_column_width = \
      cbs->new_column_width / 10 * 10;
    break;
  case XRTTBL_REASON_RESIZE_DONE:
    break;
  }
}
```

上のコールバック・ルーチンを登録するには、次のコールを使用してください。

```
XtAddCallback(table, XmNxrtTblResizeCellCallback,
  cellresizeCB, NULL);
```

5.6 セルのトラバース

トラバースとは、カレント・セルをある位置から別の場所へ移動することです。*TraverseToCell()*は、インタラクティブなセルのトラバースを実行します。トラバースは、編集されたカレント・セルの検証、新しいカレント・セルの位置の決定、そのセルへ移動、という三段階を通ります。図51はこれらの段階と呼び出されるコールバックを示しています。

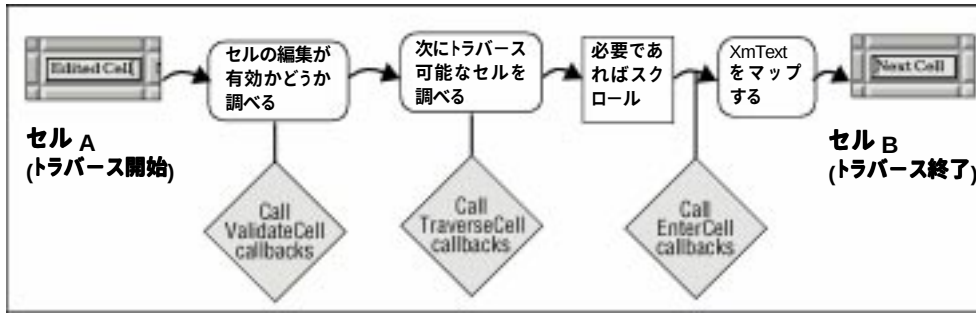


図51　セルのトラバースとコールバック

セルの検証とセルへのエントリの段階は、それぞれ別々に解説されています。セルの検証は6.2、エントリは5.7に解説があります。

(XmNxrtTblSelectionPolicyによって)ユーザがセルの選択を許可されている時、トラバースは、内部でSelectコールバックを呼び出します。

トラバースの禁止

デフォルトでは、全てのセルでトラバースができます[1]。ユーザによるセルのトラバースを防ぐには、XmNxrtTblTraversableContextをFalseにセットします。セルをトラバース不可にすることにより、(*XrtTblTraverseToCell()*またはTraverseCellコールバックを使用した)プログラミングによるトラバースも不可になります。

トラバースを不可にすると、XmNxrtTblEditableContextの設定に関らず、セルの編集も不可になります。

次の例は、行3の全てのセルをトラバース不可にセットします。

```
XtVaSetValues(table,
  XmNxrtTblContext,
    XrtTblSetContext(3, XRTTBL_ALLCELLS),
  XmNxrtTblTraversableContext, False,
  NULL);
```

[1] 3 pixel以下のセルには、トラバースの許可の設定に関らずトラバースはできません。

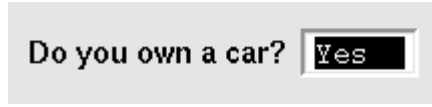
セルの表示のコントロール

デフォルトで、ユーザが現在表示されていないセルにトラバースすると、XRT/tableはそのセル全体を表示するようにスクロールします。XmNxrtTblMinCellVisibilityを用いて、セルがスクロールされてどのくらい表示されるかをコントロールできます。

テキストの選択

XmNxrtTblTextInitSelectをTrueにセットし、トラバースされた時に、そのセルのテキストが既に選択されている状態に指定できます。デフォルトでは、セルのテキストは、選択されていません

セルのテキストが前もって選択されていると、ユーザは前の値を削除せずに、そのまま新しい値を入力することができます。



ユーザが文字をタイプすると、"YES"は削除され、入力された文字に置き換わります。ユーザがセルの外へトラバースした場合は、"YES"はセルに残ります。

図52　前もって選択されているテキストを表示するセル

トラバースの強制

アプリケーションは、*XrtTblTraverseToCell()*を呼び出すことにより、カレント・セルが特定のセルにトラバースするよう強制できます。(XmNxrtTblTraversableContextによって)そのセルがトラバース不可であると、この手続きはFalseを返します。*XrtTblTraverseToCell()*は、Appendix Bに解説されていますので参照してください。

インタラクティブなトラバースのコントロール

(XmNxrtTblTraverseCellCallbackで指定される)TraverseCellコールバックを使用して、インタラクティブなトラバースをコントロールできます。各コールバックには次の構造体のポインタが返されます。

```
typedef struct {
  XrtTblReason     reason; // Read-only
  XEvent           *event; // Read-only
  String           *params; // Read-only
  Cardinal         num_params; // Read-only
  int          row, column; // Read-only
  int          next_row;
  int          next_column;
} XrtTblTraverseCellCallbackStruct;
```

TraverseCellコールバックは、*next_row*と*next_column*メンバを変更できます。*TraverToCell()*は、これらのメンバで指定されているセルにトラバースします。*next_row*と*next_column*がトラバース不可のセルに変更された場合は、トラバースは成功しません。次のコールバックは、ユーザが列0の外へトラバースしないようにしています。

```
static void
traverseCB(Widget table, XtPointer data,
  XrtTblTraverseCellCallbackStruct *cbs)
{
  if (cbs->next_column > 0) {
      cbs->next_row = cbs->row + 1;
      cbs->next_column = 0;
  }
}
```

テキスト･カーソルの位置

ユーザがセルを選択した時、通常、カーソルはそのセルのテキストの先頭に表示されます。マウスがクリックされたポイントにカーソルが来るようにするには、XmNxrtTblSetTextCursorPosリソースをTrueにします。

5.7 セルの編集

XRT/tableは、一つのXmTextウィジェットを作成、コントロールし、エンドユーザによるセル値の編集を可能にしています。デフォルトでは、ユーザはどのセルも編集することができます。ユーザがあるセルにトラバースすると、XmTextウィジェットはそのセルにマップされ、(XrtTblCellValues構造体からコピーされた)セル値を持ちます。ユーザによる編集は、ユーザが他のセルへトラバースする時に、XrtTblCellValues構造体へ書かれます。

アプリケーションは、セル毎に、編集の可･不可、大文字･小文字、複数行のサポート、combo/spinボックスのサポート、テキストの最大の長さ等をコントロールすることができます。また、テーブルのXmTextウィジェットをプログラミングすることにより、カレント･セルの属性や動きをコントロールすることもできます。

編集の禁止

XmNxrtTblEditableContextをFalseに設定すると、セルが編集不可になります。(XmNxrtTblTraversableContextによって)トラバースが不可にされていない限り、ユーザは編集不可のセルにトラバースし、標準のXコピーのメカニズムを利用して値をコピーすることができます。

注意: セルを編集可能にするには、トラバースも可能に設定してください。

次の例は、行5を編集不可(トラバースは可)にセットしています。

```
XtVaSetValues(table,
  XmNxrtTblContext,
      XrtTblSetContext(5, XRTTBL_ALLCELLS),
  XmNxrtTblEditableContext,          False,
  XmNxrtTblTraversableContext,  True,
  NULL);
```

大文字･小文字の変換

デフォルトでは、入力されたテキストのケース(大文字、小文字)は変更されません。ユーザによってトラバースまたは編集されたセルのテキストは、XmNxrtTblCaseContextがXRTTBL_CASE_UPPERであると全て大文字、XRTTBL_CASE_LOWERであると全て小文字になります。このリソースは、設定された時点から適用され、それ以前に入力されているセル値は変換されません。

複数行のセル

デフォルトでは、セルにはテキストが一行しか入力できません。複数行のテキストの入力を可能にするには、XmNxrtTblMultilineContextをTrueにします。このリソースは、セルがトラバースする時に、対応するXmNeditMode(XmTextリソース)をセットします。セルが既に複数行の値を持っている時は、このリソースは無視されます。

XmNxrtTblMultilineContext=False



XmNxrtTblMultilineContext=True



図53　1行と複数行のセル

Combo BoxとSpin Boxを使用する

...PickList...とXmNxrtTblDataTypeContextを使用して、セルの編集にCombo BoxやSpin Boxを使用することができるようになります。(但し、XRT/fieldという別のXRT製品のライセンスが必要です。) 詳細は、"XRT/fieldの使用"をご覧ください。

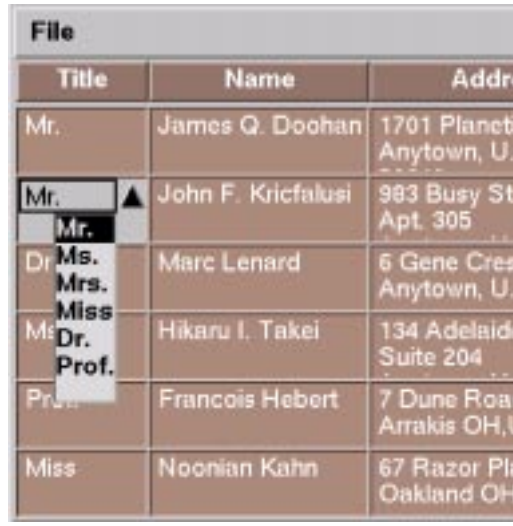


図54　セル内のCombo Box

テキストの長さ

ユーザがセルに入力できる文字の数は、XmNxrtTblMaxLengthContextの値を設定することにより制限できます。このリソースは、セルがトラバースされた時に、対応するXmNmaxLength(XmTextリソース)をセットします。ここで設定される最大文字数には、複数行のセルの*newline*文字も含まれます。

XmTextウィジェットの属性

カレント・セルのXmTextリソースは、設定したり取り出すことができます。XmNxrtTblTextはカレント・セルのウィジェットIDを指定します。次の例は、カレント・セルの背景色をセットしています。

```
Widget current_cell;
XtVaGetValues(table,
  XmNxrtTblText, &current_cell,
  NULL);
XtVaSetValues(current_cell,
  XtVaTypedArg, XmNbackground,
  XmRString, "grey80", strlen("grey80") + 1,
  NULL);
```

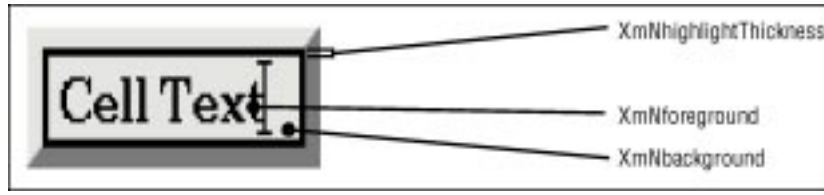


図55　カレント・セルのXmTextの属性

XmTextには、".text"を使用してリソースをセットすることができます。(Appendix Gには、リソース・ファイルでのリソースの指定法が解説されています。また、XmTextウィジェットのプログラミングに関する詳細は、Motif関連資料を参照してください。)

ランタイムの編集のコントロール

(XmNxrtTblEnterCellCallbackで指定される)EnterCellコールバックを用いて、ランタイムにセルの編集をコントロールできます。EnterCellコールバックは、*TraverseToCell()*によってトラバースの最後に呼び出されます。(図51参照) 各コールバックには、次の構造体のポインタが渡されます。

```
typedef struct {
  XrtTblReason     reason;      // Read-only
  XEvent           *event;      // Read-only
  String           *params;     // Read-only
  Cardinal         num_params;  // Read-only
  int row,         column;      // Read-only
  Boolean      doit;
} XrtTblEnterCellCallbackStruct;
```

EnterCellコールバックは、セルへのトラバースが完了する前(*reason*がXRTTBL_REASON_ENTER_CELL_BEGIN)と、後(*reason*がXRTTBL_REASON_ENTER_CELL_END)に呼び出されます。

次の表は、各段階でコールバックが変更できるメンバです。

Reason	変更できるメンバ	コールバックでの使用
XRTTBL_REASON_ENTER_CELL_BEGIN	doit	Falseに変更し、ユーザによるセルの編集を不可にする
XRTTBL_REASON_ENTER_CELL_END	None - 通知だけ	

値によって特定のセルを編集不可に指定するには、*reason* がXRTTBL_REASON_ENTER_CELL_BEGINの時、コールバックがセル値を取得し、*doit*をFalseにセットするようにします。次のコールバック・ルーチンは、ユーザがトラバースした時にセル値を現在の時刻にセットしています。

```
static void
enterCB(Widget table, XtPointer data,
  XrtTblEnterCellCallbackStruct *cbs)
{
  time_t now;

  if (cbs->reason != XRTTBL_REASON_ENTER_CELL_BEGIN)
    return;
  now = time(0);
  XtVaSetValues(table,
    XmNxrtTblContext,
      XrtTblSetContext(cbs->row, cbs->column),
    XmNxrtTblCellValueContext, ctime(&now),
    NULL);
}
```

5.8 セルの選択

ユーザは、インタラクティブに複数のセル範囲を選択することができ、アプリケーションは、各範囲を取り出してその中のセルを操作することができます。また、アプリケーションはユーザによる選択の通知を受け、ユーザがどのセルをどのように選択するかをコントロールすることもできます。

選択モード

XmNxrtTblSelectionPolicyは、エンドユーザとアプリケーションによるセルの選択の範囲を指定します。デフォルトでは、選択は許可されていません。この選択モードを変更すると、次回からの選択に影響しますが、現在の選択には影響しません。
図56は、このリソースの有効な値と、許可される選択の範囲です。

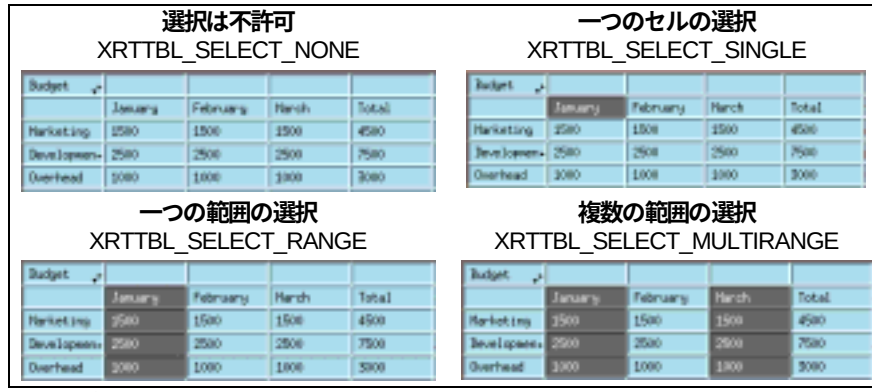


図56　選択のモード

選択されたセルのリスト

XmNxrtTblSelectedCellListは、現在テーブルで選択されている全ての範囲のリスト(NULLで終了)を指定します。このリソースは、ユーザがセルを選択するとダイナミックにアップデートされます。また、アプリケーションによってセルが選択または選択解除された時にも、アップデートされます。ラベルは、選択範囲の一部にすることはできません。[1]

選択されているセルのリストの各範囲は、XrtTblRange構造体です。

```
typedef struct {
  int      start_row, start_column;
  int      end_row, end_column;
} XrtTblRange;
```

この構造体の全てのメンバは、行と列インデックスです。*end_row*と*end_column*はXRTTBL_MAXINTとして、行/列の全てのセルと指定することもできます。ユーザはどの方向にも選択を開始することができるので、開始ポイントは、範囲の左上角である必要はありません。

次の例は、二つの選択された範囲をセットします。

```
static XrtTblRange selection1 = {0, 0, 4, 2};
static XrtTblRange selection2 = {7, 1, 7, 4};
static XrtTblRange *all_selections[] = {
  {&selection1, &selection2, NULL};
XtVaSetValues(table,
  XmNxrtTblSelectionPolicy, XRTTBL_SELECT_MULTIRANGE,
  XmNxrtTblSelectedCellList, all_selections,
  NULL);
```

[1] ラベルをクリックすると、その行または列の全てのセルが選択されますがラベルは含まれません。

選択の色

XmNxrtTblSelectedBackgroundとXmNxrtTblSelectedForegroundリソースで、選択されたセルを示す色をカスタマイズできます。詳細は、3.9を参照してください。

選択された範囲の操作

選択された範囲を取得するには、XrtTblRange構造体を割り当て、*XrtTblGetSelectedRange()*を呼び出します。この手続きは、次のプロトタイプを持っています。

```
Boolean
XrtTblGetSelectedRange(
  Widget       table,
  int      position
  XrtTblRange *range    /* Returned */
  )
```

次の例は、選択された各範囲を取得します。(選択モードはXRTTBL_SELECT_MULTIRANGEと仮定)

```
XrtTblRange range;
int i = 0;
while (XrtTblGetSelectedRange(table, i, &range)) {
  /* Process this range */
...
  i++;
}
```

特定のセルが選択されているかどうかを調べ、選択されている場合にその範囲を取り出すには、*XrtTblIsCellSelected()*を呼び出します。この手続きは、次のプロトタイプを持っています。

```
Boolean
XrtTblIsCellSelected(
  Widget           table,
  int          row, column
  int          *range_index,     /* Returned */
  XrtTblRange  *range            /* Returned */
  )
```

選択の強制

アプリケーションは、選択されているセルのリストに新たに選択を追加することができます。範囲を定義し、*XrtTblRangeListAdd()*を呼び出して、その範囲を現在選択されている範囲のコピーに追加してください。次にXmNxrtTblSelectedCellListを新しい範囲のリストにセットします。

```
XrtTblRange **all_selections;
static XrtTblRange myrange = {0, 0, 4, 2};

XtVaGetValues(table, XmNxrtTblSelectedCellList,
     &all_selections, NULL);
all_selections = XrtTblRangeListCopy(all_selections);
all_selections = XrtTblRangeListAdd(all_selections,
     XRTTBL_MAXINT, &myrange);

XtVaSetValues(table,
  XmNxrtTblSelectionPolicy, XRTTBL_SELECT_MULTIRANGE
  XmNxrtTblSelectedCellList, all_selections,
  NULL);
XrtTblRangeListFree(all_selections);
```

選択の削除

現在のリストから範囲を削除するには、選択されているセルのリストのコピーで*XrtTblRangeListDelete()*を使用し、XmNxrtTblSelectedCellListを新しい範囲のリストにセットしてください。

テーブルから全ての選択を削除するには、XmNxrtTblSelectedCellListをNULLにします。

範囲リスト内で特定の行/列を探すには*XrtTblRangeListFind()*、範囲リスト内で特定の範囲を探すには*XrtTblRangeListFindRange()*を使用してください。どちらの場合も、該当すす行や列、範囲がない場合は、XRTTBL_NOTFOUNDが返されます。

ランタイムの選択のコントロール

(XmNxrtTblSelectCallbackで指定される)Selectコールバックを使用して、インタラクティブなセルの選択を各段階でコントロールすることができます。各コールバックには、次の構造体のポインタが渡されます。

```
typedef struct {
  XrtTblReason     reason;      // Read-only
  XEvent           *event;      // Read-only
  String           *params;     // Read-only
  Cardinal         num_params;  // Read-only
  int          range_index; // Read-only
  XrtTblRange  *range;      // Read-only
  int          row, column;
  int          click_count;
  Boolean      doit;
} XrtTblSelectCallbackStruct;
```

Selectコールバックは、選択が開始される前(*reason*がXRTTBL_REASON_SELECT_BEGIN)、選択されているセルが選択解除される前(*reason*がXRTTBL_REASON_UNSELECT)、各セルが選択される時(*reason*がXRTTBL_REASON_SELECT_EXTEND)、選択が終了した後(*reason*がXRTTBL_REASON_SELECT_END)に呼び出されます。次の表は、各段階でコールバックが変更できるメンバを示しています。

Reason	変更できるメンバ	コールバックでの使用
XRTTBL_REASON_SELECT_BEGIN	row、column、doit	row/columnを変更し、選択を新しい位置から開始する。doitをFalseに変更し、選択を不可にする
XRTTBL_REASON_UNSELECT	doit	Falseに変更し、ユーザによる選択解除を不可にする
XRTTBL_REASON_SELECT_EXTEND	row、column、doit	row/columnを変更し、選択を制限する。doitをFalseに変更し、選択を不可にする。
XRTTBL_REASON_SELECT_END	None - 通知だけ	

*Select()*は、インタラクティブな選択の最中にこれらのメンバを使用します。例えば、特定なセルの選択を不許可にするには、*reason*メンバがXRTTBL_REASON_SELECT_BEGINまたはXRTTBL_REASON_SELECT_EXTENDの時に、*doit*メンバをFalseにセットします。次のコールバック・ルーチンは、選択を(選択が)開始された列に制限しています。

```
static void
selectCB(Widget w, XtPointer data,
  XrtTblSelectCallbackStruct *cbs)
{
  static int column;
  if (cbs->reason == XRTTBL_REASON_SELECT_BEGIN) {
    column = cbs->column;
  }
  else if (cbs->reason == XRTTBL_REASON_SELECT_EXTEND) {
    cbs->column = column;
  }
}
```

注意: ユーザがセルを選択できる時、セルのトラバース(*cell traversal*)は内部でSelectコールバックを呼び出します。指定したコールバックがユーザ・イベントのみで呼ばれるようにするには、Selectコールバック構造体の*event*メンバを調べてください。(*event*がNULLの時は、コールバックは内部でテーブルによって開始されています。)

Selectコールバックの他の使用法

このコールバックには、以下を始め幾つかの使用法があります。

- Row/Column全体の選択: *reason*がXRTTBL_REASON_SELECT_BEGINの時、*row/column*メンバを*row/column*ラベルに変更すると、ユーザはクリックすることにより行または列全体が選択できます。
- 選択解除の不許可: *reason*がXRTTBL_REASON_SELECT_BEGINの時、*doit*をFalseに変更し、ユーザによるセルの選択解除を不許可にします。
- ダブルクリック: XmNmultiClickTimeで指定されている時間内に、セル/ラベル上で行われたクリックの回数を*click_count*メンバで調べます。詳細は、5.2を参照してください。

セルのセンシティビティ

XmNxrtTblSensitiveContextを使用して、セルをcontextに対してセンシティブにすることができます。センシティブではないセルは、選択したりトラバースしたりすることができますが、灰色で表示されます。

- XmNxrtTblSensitiveSeriesは、その他のseriesリソースと同じ構文を使用してリソース・ファイルに指定できます。
- このリソースは、ContextValueCallbackで使用できます。詳細は、”Contextをプログラムによって管理する”(P139)の章を参照してください。
- このリソースは、セル内のウィジェットに(クローン化されているかどうかに関わらず)適用できます。詳細は、”テーブル内のウィジェット”(P110)の章を参照してください。
- テーブル全体がインセンシティブにセットされている場合は、このリソースは何もしません。
- インセンシティブなセルにもドラッグ&ドロップを行うことができます。

5.9 マウス・ポインタ

XRT/tableは、ウィジェット上のユーザの移動やインタラクションをマウス・ポインタで示します。ポインタがセルまたはラベル上にある時、ポインタは形を変更し、その位置での可能なアクションをユーザに示します。次の表は、XRT/tableのデフォルトのポインタです。

マウス・ポインタ	可能なアクション	リソース
I-ビームのポインタ XC_xterm	■ ポインタの下のセルは編集可能	XmNxrtTblCursorTraversable
十字のポインタ XC_plus	■ ポインタの下のセルまたはラベルはトラバース不可*	XmNxrtTblCursorNontraversable
列のリサイズのポインタ XC_right_side	■ ポインタの左にある列はリサイズ可能	XmNxrtTblCursorVertical
行のリサイズのポインタ XC_bottom_side	■ ポインタの上にある行はリサイズ可能	XmNxrtTblCursorHorizontal
行/列のリサイズのポインタ XC_bottom_right_corner	■ ポインタの上にある行と左にある列はリサイズ可能	XmNxrtTblCursorCorner

*ラベルはユーザによって編集できないので、このポインタは常にラベル上に現れます。

XRT/tableは、マウス・ポインタをトラッキングしている時は、XmNxrtTblTraversableContextとXmNxrtTblAllowCellResizeの設定を考慮します。例えば、行のリサイズが不可の時、↓と ↘ のポインタは表示されません。

デフォルトのポインタの変更

前述の表にあるリソースを使用して、アプリケーションはデフォルトのポインタを変更することができます。Xlibの手続き*XCreateFontCursor()*は、標準のカーソルのフォントを用いてマウス・ポインタを作成します。返されたカーソルは、XRT/tableのカーソルのリソースの値として使用できます。カーソルの作成については、Xlibの資料を参照してください。

XRT/tableのカーソルを特定なカーソルにセットするには、*XrtTblSetAllCursors()*を使用してください。デフォルトに戻す場合は、*XrtTblRestoreAllCursors()*が使用できます。

ポインタのトラッキングを不可にする

ウィジェット全体で、アプリケーションが定義したマウス・ポインタを使用するには、`XmNxrtTblTrackCursor`を`False`にセットしてください。XRT/tableは、ウィジェット上のマウスの位置をトラッキングしません。

TrackCursorコールバック

アプリケーションは、(`XmNxrtTblTrackCursorCallback`で指定される)TrackCursorコールバックを使用して、ユーザがマウスをテーブル上で移動した時に通知を受けることができます。各コールバックには、次の構造体のポインタが渡されます。

```
typedef struct {
  XrtTblReason         reason;          // Read-only
  XEvent               *event;          // Read-only
  int              row, column; // Read-only
  XrtTblBorderSides    sides;           // Read-only
  Boolean          enter_cell;      // Read-only
  Cursor               old_cursor;      // Read-only
  Cursor               new_cursor;
} XrtTblTrackCursorCallbackStruct;
```

TrackCursorコールバックは、ポインタが、セル/ラベルを放れる時(*reason*が`XRTTBL_REASON_CURSOR_LEAVE`)、セル/ラベルの罫線上にある時(*reason*が`XRTTBL_REASON_CURSOR_EDGE`)、セル/ラベル内にある時(*reason*が`XRTTBL_REASON_CURSOR_INSIDE`)に呼び出されます。

コールバックを使用して、*new_cursor*メンバを変更し、ランタイムに異なったマウス・ポインタを指定できます。これは、ResizeCellコールバックによりリサイズが不可になっている行/列で、リサイズのポインタを表示できないようにする場合に便利です。(次の例参照)

```
/* Disallow resize of row 3 */
static void
resizeCB(Widget table, XtPointer data, XrtTblResizeCellCallbackStruct *cbs)
{
    if (cbs->row == 2)
         cbs->doit = False;
}
/* Change mouse pointer when over bottom of row 3 or top of row 4 */
static void
trackCB(Widget table, XtPointer data, XrtTblTrackCursorCallbackStruct *cbs)
{
    Cursor cursor;

    if ((cbs->row == 2
              && cbs->reason == XRTTBL_REASON_CURSOR_EDGE
              && (cbs->sides & XRTTBL_BORDERSIDE_BOTTOM) != 0)
         || (cbs->row == 3
              && cbs->reason == XRTTBL_REASON_CURSOR_EDGE
              && (cbs->sides & XRTTBL_BORDERSIDE_TOP) != 0)) {
         XtVaGetValues(table, XmNxrtTblCursorTraversable, &cursor, NULL);
         cbs->new_cursor = cursor;
    }
}

...
XtAddCallback(table, XmNxrtTblTrackCursorCallback, trackCB, NULL);
XtAddCallback(table, XmNxrtTblResizeCellCallback, resizeCB, NULL);
```

5.10 ドラッグ&ドロップ

XRT/tableは、Motif 1.2のドラッグ&ドロップのメカニズムをサポートしています。ドラッグ&ドロップを利用すると、アプリケーション内またはアプリケーション間でテキストをインタラクティブに移動することができます。

デフォルトでは、セルのテキストのみにドラッグ&ドロップを行うことができます。行や列のラベルは、行や列全体が移動されたり、移動するように明示的に指定されていない限り、通常は移動しません。

ドラッグ&ドロップの機能は、アプリケーションで変更したり拡張したりすることができます。詳細は、6.9を参照してください。

選択方法の指定

デフォルトでは、一つのセルまたは1行/1列全体のテキストがドラッグ&ドロップできます。XmNxrtTblSelectionPolicy が XRTTBL_SELECT_RANGE または XRTTBL_SELECT_MULTIRANGEにセットされている時は、セル範囲と複数の行/列がドラッグ&ドロップできます。図57は、ドラッグ&ドロップによるセル範囲のコピーと移動を示しています。

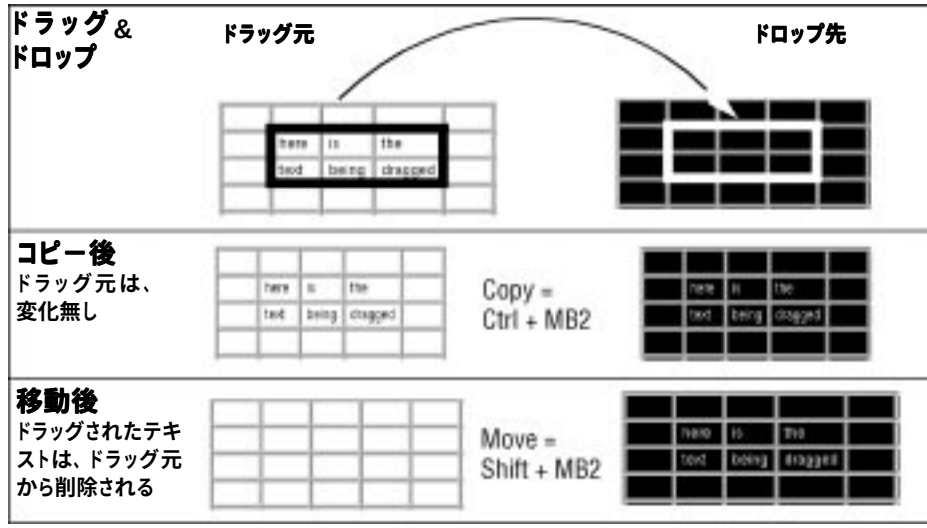


図57　セル範囲のドラッグ&ドロップ

CtrlまたはShiftを押さずにマウスの中央ボタンを押すと、次の動作が起こります。

- セルとセル範囲がコピーされる
- 行と列全体が移動する

ドラッグ&ドロップを不可にする

テーブル上の全てのドラッグ&ドロップを不可にするには、このデフォルトのトランスレーションが*NoAction()*ルーチンを呼び出すようにします。

```
static char trans[] = { "Shift <Btn2Down>:   NoAction()",
                        "Ctrl <Btn2Down>:         NoAction()",
                        "<Btn2Down>:              NoAction()" };
...
XtOverrideTranslations(table,
                        XtParseTranslationTable(trans));
for (i=0; child[i]; i++)
  XtOverrideTranslations(child[i],
                        XtParseTranslationTable(trans));
```

注意: これらのトランスレーションはリソースファイルではセットできません。

ドロップを不可にするには、XmNxrtTblImportTragetsをNULLにセットします。テーブルからのドラッグのみを禁止するには、XmNdragInitiatorProtocolStyleリソースをXmDRAG_NONEにセットします。テーブルへのドロップのみを禁止するには、XmNdragReceiverProtocolStyleをXmDRAG_NONEにセットしてください。

一つのトラバース可能なセルをドラッグ&ドロップする時は、XmTextウィジェットのドラッグ&ドロップのメカニズムが使用されます。従って、一つのトラバース可能なセルのドラッグ&ドロップ動作を変更するには、XmTextのドラッグ&ドロップのトランスレーションを置き換えてください。[1]

行/列ラベルのドラッグ

[1] デフォルトでは、XmTextのドラッグ&ドロップの動作はXRT/tableと同様です。

デフォルトでは、行または列全体がドラッグされる時、ラベルもドラッグされます。行/列だけをドラッグするには、XmNxrtTblImportTargetsで指定されるリストからXRTTBL_TGT_ROWLABELSまたはXRTTBL_TGT_COLUMNLABELSを削除してください。次の例では、行と列ラベルのドラッグ&ドロップを不可に設定しています。

```
Widget dragtable, droptable;
XrtTblImportExportStruct **implist, **explist;

/* disable row label dragging */
explist =
  XrtTblImportExportListCopy(defaultExportTargets);
explist = XrtTblImportExportListDelete(explist,
  XRTTBL_TGT_ROWLABELS);
XtVaSetValues(dragtable,
  XmNxrtTblExportTargets, explist,
  NULL);
XrtTblImportExportListDestroy(explist);

/* disable row label dropping */
implist =
  XrtTblCopyImportExportList(defaultImportTargets);
implist = XrtTblImportExportListDelete(implist,
  XRTTBL_TGT_ROWLABELS);
XtVaSetValues(droptable,
  XmNxrtTblImportTargets, implist,
  NULL);
XrtTblImportExportListDestroy(implist);
```

ドラッグ&ドロップの禁止

指定されたセルへのドロップを禁止するには、ドロップを不可にするセルまたは範囲のXmNxrtTblEditableContextをFalseにセットしてください。

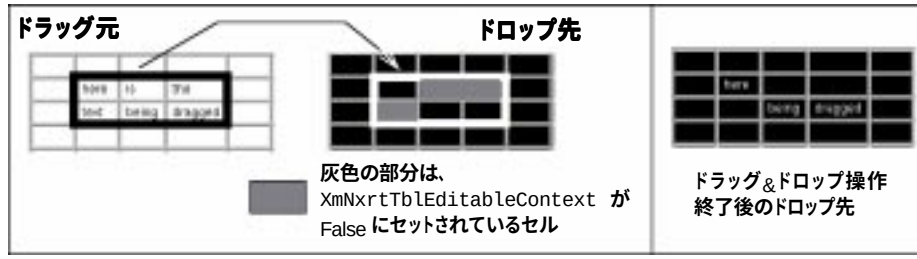


図58　指定したセルへのドロップを禁止する

行または列全体のドラッグ&ドロップを禁止するには、XmNxrtTblEditableContextをFalseにセットします。contextは、行ラベルの場合は(ALL, LABEL)、列ラベルの場合は(LABEL, ALL)にします。この設定は、テーブルに行または列ラベルが定義されていなくても有効です。

行または列全体を移動する時は、ドロップ先の行または列ラベルが編集可能な場合にのみ、編集不可のセルを移動できます。

6

高度なプログラミング

セル/ラベル内のウィジェット ■ セル/ラベル値の検証 ■ XRT/fieldを一緒に使用する
テーブルのサイズとリサイズ ■ 複合ストリング ■ Seriesリソースのプログラミング
共通のセル値を持つ複数のテーブル ■ XRT/graphとXRT/3dのデータ
ドラッグ&ドロップのプログラミング ■ XRT/tableの国際化 ■ パフォーマンスについて
警告とエラー・メッセージのカスタマイズ ■ C++の使用
UILの使用 ■ リソース・ファイル

ここでは、XRT/tableの基本操作を習得した方へ、更に進んだ使用法を紹介します。

6.1 セル/ラベル内のウィジェット

セルまたはラベルは、テキストやpixmapの他にMotifのウィジェットも持つことができます。XRT/tableは、Primitive、Composite、サード・パーティのウィジェットを表示します。セル/ラベル内のウィジェットは、その属性をプログラミングしたり、ユーザ・インタラクションを処理したりすることができます。テーブルはウィジェットのサイズ、位置、その他の属性(色等)を管理します。



図59 セル内のウィジェット

セル/ラベル内のウィジェットは、テーブル内でメニューやボタンを提供する時に便利です。例えば、*entry*という例(*$XRTHOME/src/table/examples*)では、一つの列にOptionMenusを使用し、ユーザはリストから値を選択することができます。

6.1.1 テーブル内のウィジェット

ウィジェットの追加

テーブルのXmNxrtTblWidgetContextリソースの設定または、ウィジェットのXmNxrtTblWidgetLocation constraintリソースの設定により、ウィジェットをセル/ラベルのcontextに配置することができます。

下の例は、XmNxrtTblWidgetLocationを使用してウィジェットをセル内に配置しています。

```
button = XtVaCreateWidget("button",
  xmArrowButtonWidgetClass,     table,
  XmNarrowDirection,            XmARROW_UP,
  XmNxrtTblWidgetLocation,      XrtTblSetContext(1, 5),
  NULL);
```

次の例は、XmNxrtTblWidgetContextをセットしています。

```
title = XtVaCreateWidget("title",
  xtXrtLabelWidgetClass,    table,
  XmNlabelString,       "Employee Name",
  NULL);
XtVaSetValues(button,
  XmNxrtTblContext, XrtTblSetContext(XRTTBL_LABEL, 0),
  XmNxrtTblWidgetContext, title,
  NULL);
```

ウィジェットはテーブルでセットされた時、テーブルがその管理をコントロールするので、ウィジェットはunmanaged(管理しない)で作成してください。

ウィジェットを行と列ラベルが交差する領域に配置する場合は、XmNxrtTblWidgetContextをセットする前に、contextを(XRTTBL_LABEL, XRTTBL_LABEL)にセットします。

ウィジェットのクローン

XRT/tableは、次の場合、セルまたはラベル内のウィジェット・セットと同一のコピー(*clones*)を作成します。

- (XmNxrtTblWidgetContextを使用して)ウィジェットが追加された時、カレントcontextが二つ以上のセル/ラベル(列全体等)の場合。

クローンは、その位置がXmNxrtTblWidgetLocation(またはXmNxrtTblWidgetContext)によって指定される場合に作成されます。従って、コールバックも含む全てのクローンのリソースは、その前に設定されていなければなりません。次の例では、一つの列全体に充分な数のクローンを作成しています。

```
button = XtVaCreateWidget("button",
  xmArrowButtonWidgetClass,     table,
  XmNarrowDirection,            XmARROW_UP,
  NULL);
XtAddCallback(table, XmNxrtTblDisplayWidgetCallback,
  buttonCB, NULL);
XtVaSetValues(button,
  XmNxrtTblWidgetLocation,
    XrtTblSetContext(XRTTBL_ALLCELLS, 5),
  NULL);
```

この例では、*buttonCB()*コールバックがクローン化されたarrowボタンのリソースを変更しています。

下記のウィジェット・クラスがclone化できます。

- Primitiveウィジェット - XmArrowButton, XmDrawnButton, XmLabel, XmList, XmPushButton, XmScrollBar, XmSeparator, XmText, XmTextField, XmToggleButton, XmXrtField(とそのサブクラス), XtXrtLabel, XtXrtGraph, XtXrt3d
- Compositeウィジェット - XmBulletinBoard, XmCommand, XmDrawingArea, XmFileSelectionBox, XmForm, XmFrame, XmMainWindow, XmMessageBox, XmPanedWindow, XmRowColumn(OptionMenuは例外), XmScrolledWindow, XmSelectionBox, XmXrtComboBox, XmXrtSpinBox, XrtTable

テーブルは、表示されているセルに必要な数のクローンをダイナミックに作成します。マスター・ウィジェット(クローン化されるウィジェット)は、セルにはマップされません。このクローン化されたウィジェットは、ユーザがテーブルをスクロールすると、作成と破棄のサイクルを繰り返します。表示される行/列の数が変化すると、テーブルは必要に応じてクローンを作成、破棄します。

`XmNxrtTblArgListContext`リソースと(`XmNxrtTblDisplayWidgetCallback`で指定される)DisplayWidgetコールバックを使用して、クローン化されているウィジェットを個々にプログラミングできます。但し、クローン化されたウィジェットのリソースは、ウィジェットの位置が`XmNxrtTblWidgetLocation`または`XmNxrtTblWidgetContext`で指定された後はセットできません。(クローンは、位置が指定される時に作成されます。) この点は、以前のXRT/tableのバージョンと異なりますので注意してください。

注意: ウィジェットは、`XrtTblRangeContext`を使用してはクローン化できません。

ウィジェットのリソース設定

クローン化されたウィジェットは、セル/ラベルのリソースが適用されることを除けば、元となったウィジェットと全く同一です。ウィジェットがセルまたはラベルにセットされた時、XRT/tableのリソースは、そのウィジェット・クラスの対応するリソースに適用されます。これらのウィジェットのリソースをセットするには、ウィジェットを含むセルでXRT/tableのリソースをセットしてください。次ページの表は、セル/ラベル内に位置するウィジェットに適用されるXRT/tableのリソースです。

	Table Resources													
Widgets	XmNxrtTblAlignmentContext	XmNxrtTblBackgroundContextt	XmNxrtTblCellValueContext	XmNxrtTblCellValueContext	XmNxrtTblEditableContext	XmNxrtTblFontListContext	XmNxrtTblForegroundContext	XmNxrtTblLabelContext	XmNxrtTblMaxLengthContext	XmNxrtTblMultilineContext	XmNxrtTblPixelHeightContext	XmNxrtTblPixelWidthContext	XmNxrtTblPixmapContext[a]	XmNxrtTblTraversableContext
XmLabel	✔	✔	✔			✔	✔	✔			✔	✔	✔	✔
XmPushButton	✔	✔	✔			✔	✔				✔	✔	✔	✔
XmText		✔		✔	✔	✔	✔		✔	✔	✔	✔	✔	✔
XmTextField		✔			✔	✔	✔		✔		✔	✔	✔	✔
XmXrtField (and subclasses)		✔		✔	✔	✔	✔		✔	✔	✔	✔	✔	✔
XtXrtLabel	✔	✔	✔			✔	✔	✔			✔	✔	✔	✔
XtXrtTable		✔					✔				✔	✔	✔	✔
Any other widget		✔					✔				✔	✔	✔	✔
	XmNalignment	XmNbackground	XmNlabelString	XmNvalue	XmNeditable	XmNfontList	XmNforeground	XmNlabelString	XmNmaxLength	XmNeditMode	XmNheight	XmNwidth	XmNbackgroundPixmap	XmNtraversalOn
	Widget Resources													

[a] pixmapは、XmNxrtTblPixmapLayoutContextの設定に関らずタイルのパターンで適用されます。

ウィジェットを持つセル/ラベルにXRT/tableのリソースがセットされると、XRT/tableはそのウィジェットの対応するリソースをセットします。

可変の行/列

サイズが可変の行または列に、ウィジェットがセットされた場合、XRT/tableはウィジェットに適するサイズ(*XtQueryGeometry()*で調べられます)になるよう、行/列をリサイズします。

ウィジェットの非管理

ウィジェットを管理しないよう(unmanage)にする(テーブルから一時的にウィジェットを削除する)には、ウィジェットに対応するXmNxrtTblWidgetContextをNULLにセットするか、またはウィジェットのXmNxrtTblWidgetLocationをNULLにセットしてください。ウィジェットが非管理になる時、ウィジェットは破棄されませんが、全てのクローンは破棄されます。(*XrtTblDeleteRows()*等を使用して)ウィジェットを持つテーブルのセルを削除した場合も同様に、ウィジェットがアンマップされ、クローンが破棄されます。但し、ウィジェットをクローン化する場合とは異なり、ウィジェットはXrtTblRangeContextで削除できます。

ウィジェットの破棄

テーブルの子であるウィジェットを破棄するには、*XtDestroyWidget()*を呼び出します。クローン化されたウィジェットが破棄されると、作成されていた他のクローンも一緒に破棄されます。破棄されたウィジェットがXmNxrtTblWidgetSeriesリストに含まれる場合は、seriesよりそのエレメントが削除されます。

ウィジェットの位置

アプリケーションが、ウィジェットを持つセル/ラベルの位置を知る必要がある時は、*XrtTblGetRowColByWidget()*を使用してください。これは、同じコールバック手続きが、複数のクローン化されているウィジェットから呼び出された時に便利です。

ArgListの使用

XmNxrtTblArgListContextを使用して、セル毎のウィジェットのリソース設定を記憶することができます。これは、クローン化されたウィジェットの属性を個別にプログラミングする時に便利です。このリソースは、他のリソースとは異なり、内部的にコピーされないので、XmNxrtTblArgListContextをセットした後にArgListを解放しないでください。

CreateWidgetコールバック

XRT/tableがウィジェットのクラス・ポインタ(Motif OptionMenus等)を知らない時や、ウィジェットをクローン化できない時は、(XmNxrtTblCreateWidgetCallbackで指定される)CreateWidgetコールバックを使用してウィジェットを作成またはクローン化してください。このコールバックは、ウィジェットが作成またはクローン化される時に、各ウィジェットの属性をセットする場合にも使用できます。各コールバックには、次の構造体のポインタが渡されます。

```
typedef struct {
  XrtTblReason     reason;      // Read-only
  XEvent       *event;      // Read-only
  int      row, column; // Read-only
  Widget       parent;      // Read-only
  WidgetClass  widget_class;    // Read-only
  String       class_name, name; // Read-only
  Widget       clone_source;    // Read-only
  Widget       new_widget;
} XrtTblCreateWidgetCallbackStruct;
```

CreateWidgetコールバックは、*reason*がXRTTBL_REASON_CREATE_WIDGET_BEGINの時、*new_widget*メンバを変更することができます。XRT/tableが認識しない

ウィジェット(*widget_class* が NULL)を作成するには、*reason* が XRTTBL_REASON_CREATE_WIDGET_BEGINの時、このコールバックでウィジェットを作成し、*new_widget*をそのウィジェットにセットします。

CreateWidgetコールバックは、テーブルが作成される時に追加してください。(この時に、リソース・ファイルで指定されているリソースがテーブルに適用されます。) *XtAddCallback()*は、テーブルが作成された後にコールバックを追加します。次の例は、テーブルを作成する時にどのようにコールバックを追加するかを示しています。

```
static XtCallbackRec createWidgetCBlist[] = {
  {(XtCallbackProc) createWidgetCB, NULL},
  {(XtCallbackProc) NULL, NULL}};
...
table = XtVaCreateManagedWidget("table",
  xtXrtTableWidgetClass,        parent,
  XmNxrtTblCreateWidgetCallback,    createWidgetCBlist,
  NULL);
```

DisplayWidgetコールバック

(XmNxrtTblDisplayWidgetCallbackで指定される)DisplayWidgetコールバックを使用して、ウィジェットが表示される前に属性をセットできます。DisplayWidgetコールバックは、(XmNxrtTblArgListContextで指定される)ArgList値を適用した後に呼び出されます。各コールバックには、次の構造体のポインタが渡されます。

```
typedef struct {
  XrtTblReason     reason;      // Read-only
  XEvent       *event;      // Read-only
  int     row, column; // Read-only
  Widget       clone_source;    // Read-only
  Widget       widget;
} XrtTblDisplayWidgetCallbackStruct;
```

DisplayWidgetコールバックは、*widget*メンバを異なったウィジェットにセットすることができます。

ウィジェットがクローン化される場合、ウィジェットは位置が指定された時点でクローン化されるので、ウィジェットの位置が(XmNxrtTblWidgetLocationまたはXmNxrtTblWidgetCallbackによって)指定された後はXmNxrtTblDisplayWidgetCallbackをセットできません。

リソース・ファイルで指定されているウィジェットにDisplayWidgetコールバックを使用するには、このコールバックをテーブルが作成される時に追加してください。

```
static XtCallbackRec displayWidgetCBlist[] = {
  {(XtCallbackProc) displayWidgetCB, NULL},
  {NULL, NULL}};
...
table = XtVaCreateManagedWidget("table",
  xtXrtTableWidgetClass,            parent,
  XmNxrtTblDisplayWidgetCallback,   displayWidgetCBlist,
  NULL);
```

WidgetSetValueコールバック

XmNxrtTblWidgetSetValueCallbackを使用して、セル/ラベル内のウィジェットのリソースがセットされる時に呼び出されるコールバックのリストを作成できます。これを利用して、アプリケーションはリソース値を不可にしたり修正したりする

ことができます。リストの各コールバックには、次の構造体が渡されます。

```
typedef struct {
  int reason;      // Read-only
  XEvent  *event;      // Read-only
  int row, column; // Read-only
  Widget  widget;      // Read-only
  Arg *arglist;        // Read-only
  int num_arglist; // Read-only
  Boolean doit;
} XrtTblWidgetSetValueCallbackStruct;
```

各コールバックは、リソース値がセットされる前と後に呼び出されます。*reason*は、現在、どちらの状態かを示し、XRTTBL_REASON_SETVALUE_BEGIN(リソースがセットされる前)か、XRTTBL_REASON_SETVALUE_END(リソースがセットされた後)のどちらかです。

*doit*メンバは、リソース設定を置き換えることができます。*doit*がFalseであれば、*arglist*で指定されているリソースはアップデートされません。デフォルトでは、*widget*で指定されたウィジェットがOSFウィジェットであれば*doit*はTrue、それ以外のウィジェットであればFalseです。

影とウィジェット

ウィジェットがセルまたはラベルに表示される時、セル/ラベルの影を描画するかどうか指定できます。描画する場合は、XmNxrtTblWidgetShadowContextリソースをTrueにセットします。

```
XtVaSetValues(table,
  XmNxrtTblContext,XrtTblSetContext(1,1),
  XmNxrtTblWidgetShadowContext, True);
```

デフォルトでは、XmNxrtTblWidgetShadowCotnextはFalseで、セル/ラベルの影は描画されません。影の幅は、XmNxrtTblShadowThicknessリソースでコントロールします。

例

セル内でウィジェットを使用する例が、サンプル・プログラムの*entry*(*$XRTHOME/src/table/examples*)にあります。より複雑なウィジェットを使用したデモは、*$XRTHOME/src/table/demos/widgets*にあります。

6.1.2 リソース・ファイルのウィジェット

XRT/tableは、(XmNxrtTblWidgetClassListで指定される)ウィジェットのクラス・ポインタを知っている時は、リソース・ファイルで指定されるウィジェットを作成します。以下にあるリソース・ファイルの一部は、ある列の全てのセルにXmPushButtonウィジェットを作成しクローン化しています。

```
*.XtXrtTable.xrtTblWidgetSeries: \
  (allcells 4 XmPushButton.mybutton)
```

ウィジェット・クラスのリスト

XmNxrtTblWidgetClassListは、XRT/tableがリソース・ファイルから作成できるウィジェットのクラス・ポインタとウィジェット・クラスを指定します。このリストは、次のウィジェットで構成されます。

- Primitiveウィジェット - XmArrowButton, XmDrawnButton, XmLabel, XmList, XmPushButton, XmScrollBar, XmSeparator, XmText, XmTextField, XmToggleButton, XmXrtfield(とそのサブクラス), XtXrtLabel
- Compositeウィジェット - XmBulletinBoard, XmCommand, XmDrawingArea, XmFileSelectionBox, XmForm, XmFrame, XmMainWindow, XmMessageBox, XmPanedWindow, XmRowColumn, XmScale, XmScrolledWindow, XmSelectionBox, XmXrtComboBox, XmXrtSpinbox, XtXrtTable

リソース・ファイルでサード・パーティのウィジェットを指定するには、テーブルを作成する時に、そのウィジェットのクラス名とクラスポインタを`XmNxrtTblWidgetClassList`に追加してください。[1] 次のコードは、XRT/graphをウィジェットのクラス・リストに追加しています。

```
static XrtTblWidgetClass class_list[] = {
  {"XtXrtGraph", &xtXrtGraphWidgetClass},
  {NULL, NULL}};
...
table = XtVaCreateManagedWidget("table",
  xtXrtTableWidgetClass,        parent,
  XmNxrtTblWidgetClassList,     class_list,
  ...
  NULL);
```

6.2 セル/ラベル値の検証

概要

XRT/tableでは、セルやラベル値がテーブルのセル/ラベル値の構造体に書かれる前に、値を検証するように設定できます。編集されたテキストは、セル/ラベルに指定した“データ型”に照らし合わせて検証されます。ここで言う“データ型”とは、テキストのフォーマットや条件のことで、C言語のデータ型のことではありません。セルとラベル値は、内部では常にXmStringとして処理されます。

XRT/tableは、独自のValidateCellコールバックを使用してセル/ラベル値を検証します。このコールバックは、セルのデータ型に対応する検証ルーチンを呼び出します。検証ルーチンは編集されたセル/ラベル値を検査し、値がデータ型に従わない場合は、ValidateCellコールバック構造体の*doit*を`False`にセットします。

XRT/tableでは、データ入力フィールド・ウィジェットのXRT/fieldを使用して値の編集、検証をすることもできます。詳細は、6.3を参照してください。

セル/ラベルのデータ型

セルとラベルのデータ型の設定には、`XmNxrtTblDataTypeContext`を使用します。既に組み込まれている型にセットされている場合は[2]、XRT/tableは、(`XmNxrtTblValidatePolicy`で指定される)検証のポリシーに従い自動的に、

[1] `XmNxrtTblWidgetClassList`は、必ずウィジェットの作成時にセットしてください。(この時に、リソースファイルで指定されているリソースがテーブルに適用されます。)

[2] `XmNxrtTblDataTypeContext`は、XRT/fieldウィジェットの型にもセットすることができます。(6.3参照)

(行われた)編集を検証します。デフォルトのデータ型であるStringでは、検証は実施されません。(どれでも有効なStringです。) 次の表は、組み込まれているデータ型と検証ルーチンのリストです。

リソース値	データ型	検査ルーチン
XRTTBL_TYPE_STRING	テキスト・ストリング	ルーチンは呼ばれません
XRTTBL_TYPE_DOUBLE	倍精度実数値	*XrtTblCheckIsDouble()*
XRTTBL_TYPE_INT	整数値	*XrtTblCheckIsInt()*
XRTTBL_TYPE_FLOAT	浮動小数点値	*XrtTblCheckIsFloat()*
XRTTBL_TYPE_BOOLEAN	Boolean値	*XrtTblCheckIsBoolean()*

検査のポリシー

XmNxrtTblValidatePolicyリソースは、いつ編集を検証するか決定します。デフォルトでは、ユーザによって編集されたセルのみが検証されます。XRTTBL_VALIDATE_ALWAYSにセットされると、ユーザまたはアプリケーションによって行われたセルまたはラベルの変更が検証されます。XRTTBL_VALIDATE_NEVERの場合は、検証は実施されません。XRTTBL_VALIDATE_SETであれば、アプリケーションによって行われた変更のみが検証されます。

データ型の拡張

独自のデータ型と検証ルーチンをXmNxrtTblDataTypeListで指定されるリストに追加することにより、組み込まれているデータ型を拡張することができます。一度リストに加えられると、(XmNxrtTblDataTypeContextを使用して)セル/ラベルはそのデータ型にセットでき、定義された検証ルーチンが変更を検査するために呼び出されます。次の例は、正の整数のデータ型を定義し、データ型のリストに追加しています。

```
#define TYPE_INT_POSITIVE (XRTTBL_TYPE_LAST + 1)
static XrtTblDataTypeRoutine datatype_list[] = {
  {TYPE_INT_POSITIVE, (XtCallbackProc) checkIsPositive},
  {0, NULL}
};
...
table = XtVaCreateManagedWidget("table",
  xtXrtTableWidgetClass, toplevel,
  XmNxrtTblDataTypeList, datatype_list,
  ...
  NULL);
```

今後のXRT/tableのリリースで追加されるデータ型と、独自に定義したデータ型が対立しないように、追加するデータ型は、XRTTBL_TYPE_LASTからインデックスを付けてください。

独自に定義する検証ルーチンには、次のプロトタイプが必要です。

```
void
my_validation_method(
  Widget       table,
  XtPointer    call_data,
  XtPointer    cbs
  )
```

*cbs*はXrtTblValidateCellCallbackStruct構造体のポインタです。この構造体は、編集された値(*value*)、変換された値(*converted_value*)、編集された値が有効かどうか(*doit*)、ターミナルのベルを鳴らすかどうか(*bell*)のメンバを持ちます。検証ルーチンはこれらの値を読み取り、変更し、編集された値が有効かどうかを示します。

次の検証ルーチンは、値が正の整数であるかどうかを調べます。

```
void
checkIsPositive(Widget table, XtPointer call_data,
  XtPointer client_data)
{
  XrtTblValidateCellCallbackStruct *cbs
    = (XrtTblValidateCellCallbackStruct *) client_data;
  /* Check whether a valid integer */
  XrtTblCheckIsInt(table, call_data, cbs);
  if (cbs->doit && cbs->converted_value < 0) {
    cbs->doit = False;
    cbs->bell = True;
  }
}
```

全ての検証ルーチンのソースコードは、*$XRTHOME/src/table/util*にあります。これらを利用して、独自のルーチンを作成できます。

ValidateCellコールバック

ValidateCellコールバックは、主としてデータ検証ルーチンによって使用されます。アプリケーションは、XmNxrtTblValidateCellCallbackを使用して、検証ルーチンの後に呼び出されるValidateCellコールバックを追加することができます。これは、失敗したデータ型検証を取得し、ユーザから詳しい情報を得ようとする時に使用できます。各コールバックには、次の構造体のポインタが渡されます。

```
typedef struct {
  XrtTblReason          reason;      // Read-only
  XEvent           *event;      // Read-only
  int          row, column; // Read-only
  Boolean          value_changed;   // Read-only
  XrtTblDataType   datatype;        // Read-only
  XtArgVal             *converted_value; // Read-only
  XtPointer        user_data;   // Read-only
  String           value;
  Boolean          doit;
  Boolean          bell;
} XrtTblValidateCellCallbackStruct;
```

デモプログラム

*spreadsheet*デモ(*$XRTHOME/src/table/demos/spreadsheet*)には、組み込まれているデータ型の使用や新しいデータ型の定義の例がありますので、参照してください。

6.3 XRT/fieldを一緒に使用する

XRT/tableは、デフォルトでは、XmTextウィジェットをテキストの編集や検証に使用します。その他に、XRTファミリのデータ入力フィールド・ウィジェットのXRT/fieldを使用してテキストの検証、編集、表示を行うこともできます。XRT/fieldを使用するには、次の4つのステップに従ってください。

1 コンパイルされたプログラムをリンクする際、ライブラリのリストでは、必ず、XRT/fieldライブラリ(`libxrtfield.a`)がXRT/tableライブラリ(`libxrttable.a`)の前に来るようにしてください。

2 テーブルを作成する前に、*XrtTblEnableXrtField()*を呼び出してください。

```
XrtTblEnableXrtField();
...
table = XtVaCreateManagedWidget("table",
  xtXrtTableWidgetClass, toplevel,
  NULL);
```

3 `XmNxrtTblDataTypeContext`リソースで、使用するXRT/fieldウィジェットの型を指定します。次の型が使用できます。

`XRTTBL_TYPE_FIELD_CURRENCY`	currency(通貨)フィールド
`XRTTBL_TYPE_FIELD_DATE`	date/time(日付/時間フィールド
`XRTTBL_TYPE_FIELD_FLOAT`	floating-point(浮動小数点)フィールド
`XRTTBL_TYPE_FIELD_INT`	integer(整数)フィールド
`XRTTBL_TYPE_FIELD_NONE`	
`XRTTBL_TYPE_FIELD_STRING`	string(文字列)フィールド
`XRTTBL_TYPE_FIELD_COMBO_CURRENCY`	currency(通貨) combo box
`XRTTBL_TYPE_FIELD_COMBO_DATE`	date (日付)combo box
`XRTTBL_TYPE_FIELD_COMBO_FLOAT`	floating-point(浮動小数点) combo box
`XRTTBL_TYPE_FIELD_COMBO_INT`	integer(整数) combo box
`XRTTBL_TYPE_FIELD_COMBO_STRING`	string(文字列) combo box
`XRTTBL_TYPE_FIELD_SPIN_CURRENCY`	currency (通貨)spin box
`XRTTBL_TYPE_FIELD_SPIN_DATE`	date(日付) spin box
`XRTTBL_TYPE_FIELD_SPIN_FLOAT`	floating-point(浮動小数点) spin box
`XRTTBL_TYPE_FIELD_SPIN_INT`	integer (整数)spin box
`XRTTBL_TYPE_FIELD_SPIN_STRING`	string(文字列) spin box

指定した型のXRT/fieldウィジェットは、必要時に自動的に作成されます。一つの型につき一つのフィールドが作成され、ユーザのトラバースによってこれが再利用されます。

4 XRT/field ウィジェットのリソースをセットするには、XmNxrtTblFieldArgListContextリソースを使用します。値は、Argsの静的リストで、このリストの最後のエレメントはNULLにしてください。

```
static Arg args[3];

XtSetArg(args[0], XmNxrtFldMask, "###-####");
XtSetArg(args[1], XmNxrtFldAllowNull, False);
XtSetArg(args[2], NULL, NULL);
XtVaSetValues(table,
  XmNxrtTblContext, XrtTblSetContext(5, 5),
  XmNxrtTblFieldArgListContext, args,
  NULL);
```

編集が終了したら、編集された値はXrtTblCellValues構造体に記憶されます。

*table_entry*デモ(*$XRTHOME/src/common_demos/table_entry*)には、XRT/tableでXRT/fieldを使用する例がありますので参照してください。

注意: XmNxrtTblFieldArgListは内部的にはコピーされないので、メモリ管理はユーザ側で行ってください。

リソース・ファイルのフィールド

作成されたXRT/fieldウィジェットの名前は、その型を反映しています。

ウィジェットの型	field名	子field名	子arrow名	子arrow名
XRTTBL_TYPE_FIELD_CURRENCY	“currency_field”			
XRTTBL_TYPE_FIELD_DATE	“date_field”			
XRTTBL_TYPE_FIELD_FLOAT	“float_field”			
XRTTBL_TYPE_FIELD_INT	“int_field”			
XRTTBL_TYPE_FIELD_STRING	“string_field”			
XRTTBL_TYPE_FIELD_COMBO_CURRENCY	“combo_currency_field”	“field”	“arrow”	
XRTTBL_TYPE_FIELD_COMBO_DATE	“combo_date_field”	“field”	“arrow”	
XRTTBL_TYPE_FIELD_COMBO_FLOAT	“combo_float_field”	“field”	“arrow”	
XRTTBL_TYPE_FIELD_COMBO_INT	“combo_int_field”	“field”	“arrow”	
XRTTBL_TYPE_FIELD_COMBO_STRING	“combo_string_field”	“field”	“arrow”	
XRTTBL_TYPE_FIELD_SPIN_CURRENCY	“spin_currency_field”	“field”	“decr_arrow”	“incr_arrow”
XRTTBL_TYPE_FIELD_SPIN_DATE	“spin_date_field”	“field”	“decr_arrow”	“incr_arrow”
XRTTBL_TYPE_FIELD_SPIN_FLOAT	“spin_float_field”	“field”	“decr_arrow”	“incr_arrow”
XRTTBL_TYPE_FIELD_SPIN_INT	“spin_int_field”	“field”	“decr_arrow”	“incr_arrow”
XRTTBL_TYPE_FIELD_SPIN_STRING	“spin_string_field”	“field”	“decr_arrow”	“incr_arrow”

このフィールドのリソースは、リソース・ファイルでセットできます。

フィールドの取得

特定な型のXRT/fieldウィジェットを取得するには、*XrtTblGetXrtFieldByType()*を呼び出します。

```
field = XrtTblGetXrtFieldByType(table,
  XRTTBL_TYPE_FIELD_CURRENCY):
```

もし、その型のフィールドが存在しない場合は、それが(unmanagedで)作成されます。フィールドを一度取得したら、セル値のフォーマットやアップデートに使用することができます。例えば、次のコードは、整数値を16進数フォーマットに変換しセルに入れています。

```
field = XrtTblGetXrtFieldByType(table,
  XRTTBL_TYPE_FIELD_INT);
XtVaSetValues(field,
  XmNxrtFldDisplayFormat, XRTFLD_FORMAT_HEX,
  XmNxrtFldInternalValue, 53,
  NULL);
value = XrtFldGetDisplayValue(field);
XtVaSetValues(table,
  XmNxrtTblContext,XrtTblSetContext(4, 3),
  XmNxrtTblCellValueContext,value,
  NULL);
XtFree(value);
```

XRT/fieldの*XrtFldGetDisplayValue()*は、フィールドが表示するストリングを取得します。このストリングは、`XmNxrtTblCellValueContext`リソースをセットすることにより、セルに記憶されます。但し、次のXRT/fieldリソースは、リソースファイルや*XtSetValues()*の呼び出しではセットできません。

```
XmNxrtFldMask
XmNxrtFldPick list
XmNxrtFldMatchPick list
XmNxrtFldPick listIndex
```

これらのリソースは、ユーザがセル間を移動する際にテーブルによって内部的にリセットされますので、これらのリソースは`XmNxrtTblFieldArgListContext`を使用してセットしてください。新しいセルでこの設定を有効にするには、`XmNxrtFldArgListContext`からリロードしてください。但し、`XmNxrtFldPick list`にはこの方法は使用できません。テーブルは(`XmNxrtTblStringPick listContext`等の)独自のpick listリソースを使用し、ユーザが新しいセルに移動した時に、そのpick listリソースがフィールドに適用されます。セルにXRT/tableのpick listが与えられていない場合は、`XmNxrtFldArgListContext`のpick listが使用されます。pick listがテーブルと`XmNxrtTblFieldArgListContext`の両方に存在する場合は、テーブルのpick listが優先されます。 pick listの指定には、`XmNxrtTblFieldArgListContext`を使用せずに、(`XmNxrtTbl...Pick list`等を使用して)できるだけテーブルにpick listを管理させるようにしてください。もし、`XmNxrtTblFieldArgListContext`を使用する場合、これらのpick listに必要なメモリの割り当て、管理、割り当て解除は、XRT/tableでは行われません。

XRT/fieldとその機能の詳細に関しては、XRT/fieldのマニュアルをご覧ください。

6.4 テーブルのサイズとリサイズ

テーブルは、Motif Compositeウィジェットの子として作成できます。通常、親のCompositeウィジェットが、テーブル・ウィジェットの適切なサイズを決定し、テーブル・ウィジェットはそれに従います。

テーブルは、次の要因を考慮しながら、各構成部分のサイズを決定します。

- ウィンドウの現在の幅と高さ。
- 各タイトルのサイズと位置。これは、各タイトルに使用されているウィジェットのサイズ、テーブルでの位置とそのオフセットに依存します。

テーブルのセルとラベルは、残りの矩形の領域にあてはまるように配置されます。全ての行または列の表示に充分なスペースがない時は、スクロールバーが現れます。ユーザはテーブルをスクロールして、全ての行や列を見ることができます。

リサイズのアプローチ

行/列のサイズに影響するリソース(図60参照)の値を変更すると、テーブルは`XmNxrtTblAllowResize`の値によって、リサイズします。このリソースは、表示されるテーブルの大きさまたは、ウィジェットのサイズをコントロールする場合に設定します。

`XRTTBL_RESIZE_NONE`の場合は、図60にあるリソースが変化してもリサイズは行われず、セル/ラベルのビューポート(viewport)のサイズは変わりません。[1]

`XRTTBL_RESIZE_ALL`であると、図60のリソースの変更によりテーブルはリサイズし、セル/ラベルのビューポートが、変更前と同じ量(同じ数のセル)のテーブルを表示します。

`XRTTBL_RESIZE_HORIZONTAL`または`XRTTBL_RESIZE_VERTICAL`に設定すると、図60のリソースの変更により、縦か横どちらかの方向にリサイズします。

[1] `XmNxrtTblVisibleColumns`と`XmNxrtTblVisibleRows`は、常にリサイズの原因となるので例外です。

リサイズの原因となるリソース

XmNxrtTblCellValueContext	XmNxrtTblCharHeight[CS]	XmNxrtTblCharWidth[CS]
XmNxrtTblColumnLabels	XmNxrtTblFontList[CS]	XmNxrtTblLabelContextM
XmNxrtTblMarginHeight	XmNxrtTblMarginWidth	XmNxrtTblPixelHeight[CS]
XmNxrtTblPixelWidth[CS]	XmNxrtTblPixmap[CS]	XmNxrtTblRowLabels
XmNxrtTblShadowThickness	XmNxrtTblWidget[CS]	

(**最後に**[CS]**が付くリソースは、**Context**と**Series**の両方のリソースを指す。**
例:XmNxrtTblFontListContext**と**XmNxrtTblFontListSeries.)

XmNxrtTblVisibleColumns**または**XmNxrtTblVisibleRows**は、常にテーブルをリサイズする。**

図60　テーブルをリサイズさせるリソース

注意:　テーブルが実際にリサイズできるかどうかは、親のウィジェットがリサイズを許可するかどうかに依存します。(テーブルがリサイズするには)Shellウィジェットの XmNallowShellResizeもTrueでなければなりません。テーブルのリサイズのリクエストが親ウィジェットによって拒否されると、テーブルは代わりにスクロールバーを表示またはアップデートします。

ExposeとResizeのコールバック

XmNxrtTblExposeCallbackまたはXmNxrtTblResizeCallbackリソースを使用してコールバックを指定することにより、アプリケーションは、テーブル・ウィジェットがいつエクスポーズまたはリサイズしたかを知ることができます。Xlibを用いてテーブルのイメージ上に描画を行う時にExposeコールバックが利用できます。Resizeコールバックは、テーブルのサイズに応じてウィジェットのリソースを調節する時に使用できます。次のResizeコールバックは、リサイズされたテーブルの幅に合うよう全ての列をリサイズします。

```
static void
resizeCB(
  Widget table,
  XtPointer data,
  XrtTblResizeCallbackStruct *cbs)
{
  int ncols, colwidth, firstwidth;
  XrtTblSeries *series;

  if (cbs->reason != XRTTBL_REASON_RESIZE_END)
    return;
  XtVaGetValues(table,
    XmNxrtTblNumColumns, &ncols, NULL);
  ncols++; /* Include row labels */
  colwidth = cbs->width/ncols;
  firstwidth = cbs->width - (ncols - 1) * colwidth;
```

```
    series = XrtTblSeriesCreate(XRTTBL_VALUE_INT);
    XrtTblSeriesAdd(series,
      XrtTblSetContext(XRTTBL_ALL, XRTTBL_ALL),
      colwidth);
    XrtTblSeriesAdd(series,
      XrtTblSetContext(XRTTBL_ALL, XRTTBL_LABEL),
      firstwidth);
    XtVaSetValues(table,
      XmNxrtTblDisplayVertScrollBar, XRTTBL_DISPSB_NEVER,
      XmNxrtTblPixelWidthSeries, series,
      XmNxrtTblAllowResize, XRTTBL_RESIZE_HORIZONTAL,
      NULL);
    XtVaSetValues(table,
      XmNxrtTblAllowResize, XRTTBL_RESIZE_NONE,
      NULL);
    XrtTblSeriesDestroy(series);
  }
```

上記のコールバック・ルーチンを登録するには、次のコールを使用します。

```
XtAddCallback(mytable, XmNxrtTblResizeCallback,
  resizeCB, NULL);
```

6.5 複合ストリング

XRT/tableは、Motifの複合ストリング(Compound String)をサポートしています。複合ストリングを使用すると、図61にあるようなマルチ・フォントのストリングを指定することができます。セル値を設定する場合に、複合ストリングを使用する必要はありません。(但し、`XmNxrtTblXmStringSetMode`がセットされていない場合。3.5.1参照)

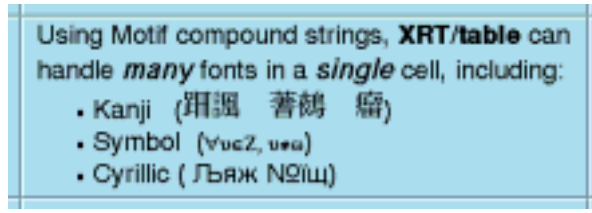


図61 マルチ・フォントを表示する複合ストリング

XmTextウィジェットは複合ストリングを表示できないので、カレント・セルは、セルのフォント・リストにある一番始めのフォントを使用して、複合ストリング(Compound String)をストリング(String)に変換します。セルは、ユーザが他のセルにトラバースしたら元のフォントに戻ります。

マルチ・フォントの複合ストリングは、リソース・ファイルには入力できません。これらは、タグ付のフォント・リストと*XmStringCreate()*を使用して、アプリケーションによって作成される必要があります。複合ストリングのプログラミングに関する詳細は、Motif関連資料をご覧ください。

次のコードは、複合ストリングを作成しセル値にセットする例です。

```
static char *fallback[] = {
    "Simple*.fontList: -adobe-helvetica-medium-r-normal--
        25-180-100-100-p-130-iso8859-1=tag1, 9x15=b", NULL
};

int
main(int argc, char *argv[])

{
    Widget toplevel, table, label;
    XtAppContext app;
    XmString s1, s2, s3;

    XmFontListEntry ent1, ent2;
    XmFontList list;

    /* Initialize Intrinsics */
    toplevel = XtVaAppInitialize(&app, "Simple", NULL, 0, &argc, argv,
        fallback, NULL);
    ent1 = XmFontListEntryLoad(XtDisplay(toplevel),
        "-adobe-helvetica-medium-r-normal--25-180-100-100-p-130-iso8859-1",
        XmFONT_IS_FONT, "tag1");
    Ent2 = XmFontListEntryLoad(XtDisplay(toplevel), "9x15",
        XmFONT_IS_FONT, "b");
    List = XmFontListAppendEntry(NULL, ent1);
    List = XmFontListAppendEntry(list, ent2);

    XrtTblSetXmStringConverter();

    s1 = XmStringCreate ("Italic", "tag1");
    s2 = XmStringCreate ("Regular", "b");
    s3 = XmStringConcat (s1,s2);

    /* Create the table */
    table = XtVaCreateManagedWidget("table",
        xtXrtTableWidgetClass, toplevel,
        XmNxrtTblNumRows,      7,
        XmNxrtTblNumColumns,   5,
        XmNxrtTblCellValueContext,s3,
        XmNxrtTblDebug,        True,
        XmNxrtTblFontListContext,list,
        NULL);

    Label = XtVaCreateManagedWidget ("label",
        XmXrtLabelWidgetClass, table,
        XmNlabelString,        s3,
        XmNxrtTblLocation,     XRTTBL_LOCATION_BOTTOM,
        NULL);

    XmStringFree (s1);
    XmStringFree (s2);
    XmStringFree (s3);

    XtRealizeWidget(toplevel);
    XtAppMainLoop(app);
}
```

6.6 Seriesリソースのプログラミング

Contextリソースをセットすると、対応するSeriesリソースもアップデートされます。Seriesリソースは、テーブルの生存中、全てのContextリソースの設定を記憶します。ここでは、Seriesリソースを直接プログラミングする方法を解説します。通常、Contextリソースを使用するプログラミングの方が簡単ですが、Seriesリソースを使用した方がよい場合もあります。

Seriesの構成

Seriesリソースの値は、seriesメンバのスタックで構成されています。全てのSeriesリソースは、次のようなXrtTblSeriesのデータ型を持っています。

```
typedef struct {
  XrtTblValueType type; /* value type */
  String       name; /* resource name */
  int      last_index; /*index of last item in list*/
  int      length; /* # elements in list */
  XrtTblAnyValue **list; /* list */
} XrtTblSeries;
```

XrtTblValueTypeとXrtTblAnyValue構造体によって、XrtTblSeries構造体はseriesのどの型(色、Boolean値、整数値等）にも使用できます。seriesの各型は、それ自身の構造体で定義されます。例えば、XrtTblIntValue構造体は、次のように定義されます。

```
typedef struct {
  XrtTblContext    context;
  int              value;
} XrtTblIntValue;
```

簡易手続き

XRT/tableには、XrtTblSeries構造体を操作する手続きが幾つかあります。これらの手続きは、Appendix Bに詳細が解説されています。

XrtTblSeriesCreate()	**新しい**XrtTblSeries**構造体を作成する。**series**は、メンバがセットまたは追加される前に作成される必要がある。**
XrtTblSeriesCopy()	**新しい**XrtTblSeries**を作成し、既存の**series**のメンバをロードする。**
XrtTblSeriesAdd()	**指定された**context**のメンバを**series**に追加する。既存の**context**は置き換えられる。**
XrtTblSeriesAddValue()	series**リストのある位置にメンバを追加する。その位置にある既存の値を置き換える。**
XrtTblSeriesSetValue()	**指定された位置の**series**メンバをセットする。**
XrtTblSeriesDelete()	series**からメンバを削除する。**
XrtTblSeriesFind()	**指定されたセルまたはラベルに使用されている値を持つ**sereis**メンバの位置を捜す。**
XrtTblSeriesIterate()	**各メンバの内容を返しながら、**series**メンバを反復(**Iterate**)していく。**

XrtTblSeriesDestroy()	XrtTblSeries**構造体を破棄し、割り当てられていたメモリを解放する。**

次からのセクションは、Seriesリソースを使用した方が便利な場合を解説しています。

複数のcontextの属性

アプリケーションがテーブルの幾つかの部分のリソースを変更する場合、全ての変更を一つのseriesに加えてからseriesリソースをそれにセットすると便利です。(下参照)

```
XtSetValues:

XrtTblSeries *series;
XtVaGetValues(table, XmNxrtTblPixelHeightSeries, &series, NULL);
Series = XrtTblSeriesCopy(series);
XrtTblSeriesAdd(series, XrtTblSetContext(5, XRTTBL_ALL), 10);
XrtTblSeriesAdd(series, XrtTblSetContext(15, XRTTBL_ALL), 20);
XtVaSetValues(table, XmNxrtTblPixelHeightSeries, series, NULL);
XrtTblSeriesDestroy(series);
```

Fallbackリソース

Seriesリソースによって、テーブルの異なった部分にFallbackリソース値を定義することができます。(下参照)

```
Static char *fallback[] = {
    "My_App*.XtXrtTable.background: #DBAFA7",
    "My_App*.XtXrtTable.xrtTblBackgroundSeries: \
        (ALL ALL Light Blue) (ALL LABEL #C29B93) \
        (LABEL ALL #C29B93)",
    "My_App*.XtXrtTable.xrtTblForegroundSeries: \
        (ALL ALL Black) (ALL LABEL White) (LABEL ALL White)",
    NULL,
};
toplevel = XtAppInitialize(&app_context, "My_App", NULL, 0,
    &argc, argv, fallback, NULL, 0);
```

パフォーマンスについて

リソースがテーブルの多くの異なった部分にセットされている場合、contextをセットして*XtGetValues()*を使用するより、直接にseriesリソース値を取り出す方が速い場合があります。次の例は、*XrtTblSeriesIterate()*を使用して、戻り値を適切なseries型にcastしながら、背景色のseriesの各メンバを繰り返(iterate)しています。

```
int row, column, position = 0;
XtArgVal value;
XrtTblSeries *bg_series;
Pixel bg_color, top_shadow;
XtVaGetValues(table, XmNxrtTblBackgroundSeries, &bg_series, NULL);
While(XrtTblSeriesIterate(bg_series,&position,&row,&column,&value))
{
    bg_color = (Pixel) value;
    ...
}
```

次の例は*XrtTblSeriesFind()*を使用して、指定されたセルまたはラベルに対応するメンバの位置を捜します。その後メンバは直接読み込むことができます。

```
int I;
char name[100];
if ((i = XrtTblSeriesFind(bg_series, 45, 34)) != XRTTBL_NOVALUE)
{
    bg_color = (Pixel) bg_series->list[i]->color.value.pixel;
    top_shadow = (Pixel) bg_series->list[i]->color.value.top_shadow;
    strcpy(name, bg_series->list[I]->color.value.name);
}
```

6.7 共通のセル値を持つ複数のテーブル

アプリケーションは、同じXrtTblCellValues構造体を共有する幾つかのテーブル・ウィジェットを作成することができます。この構造体は、これらの全てのテーブルに対して、中心になるデータとして機能します。

複数のテーブルが共通のセル値を持つようにするには、一つのテーブルがXrtTblCellValues構造体を表示し、他のテーブルはCellValueコールバックを使用して、一つめのテーブルのセル値構造体にある値を表示するようにします。

ユーザがどのテーブルのセル値でも編集できるようにするには、ValidateCellコールバックを用いて、セルの編集を取得し、XrtTblCellValues構造体と他のテーブルで表示されている値を両方アップデートしてください。

次のページは、上記の方法の例を示しています。

```
static Widget table1, table2;

/* Cell value callback for table2 */
static void
cellvalueCB(Widget table,XtPointer data,XrtTblCellValueCallbackStruct *cbs)
{
    XmString value;
    XtVaSetValues(table1,
        XmNxrtTblContext, XrtTblSetContext(cbs->row, cbs->column),
        NULL);
    XtVaGetValues(table1, XmNxrtTblCellValueContext, &value, NULL);
    cbs->string = XmStringCopy(value);
}

/* Capture cell edit and set other table's value */
static void
validateCB(Widget table,XtPointer data,XrtTblValidateCellCallbackStruct *cbs)
{
    static Boolean entered = False;
    Widget other_table = (Widget) data;

    if (!cbs->value_changed)
        return;

    /* Prevent recursion after committing cell edit */
    if (entered)
        return;
    entered = True;
    /*Must commit cell edit, so cellvalueCB will get table1's cell value*/
    if (table == table1)
        XrtTblCommitEdit(table, False);
    XtVaSetValues(other_table,
        XmNxrtTblContext, XrtTblSetContext(cbs->row, cbs->column),
        XmNxrtTblCellValueContext, cbs->value,
        NULL);
    Entered = False;
}
...
    table1 = XtVaCreateManagedWidget("table1",
        xtXrtTableWidgetClass, parent,
        XmNxrtTblCellValues, values,
        NULL);
    Table2 = XtVaCreateManagedWidget("table2",
        xtXrtTableWidgetClass, parent,
        NULL);

    /* Register ValidateCell CB, using other table as call_data */
    XtAddCallback(table1, XmNxrtTblValidateCellCallback,
        validateCB, (XtPointer) table2);
    XtAddCallback(table2, XmNxrtTblValidateCellCallback,
        validateCB, (XtPointer) table1);
    /* Register CellValue CB for second table */
    XtAddCallback(table2, XmNxrtTblCellValueCallback,
        cellvalueCB, (XtPointer) table2);
...
```

6.8 XRT/graphとXRT/3dのデータ

XRT/tableと一緒に、XRT/graphとXRT/3dを使用することができます。*XrtTblGetGraphData()*はXRT/graphの配列データセットまたはXRT/graphのデータ・オブジェクト・ハンドルを`XrtTblCellValues`構造体へ変換し、*XrtTblGet3dData()*はXRT/3dのデータセット(規則的または不規則的なグリッド)を`XrtTblCellValues`構造体へ変換します。(但し、この方法は、散布図のデータ型には使用できません。)

次の例は、XRT/graphのデータセットを変換しXRT/tableで使用しています。

```
XrtData *graph_data = XrtMakeDataFromFile(my_file);
XrtTblCellValues *cells = XrtTblGetGraphData(graph_data);
Graph = XtVaCreateManagedWidget("graph",
    xtXrtGraphWidgetClass,    parent,
    XtNxrtData,           graph_data,
    NULL);
Table = XtVaCreateWidget("table",
    xtXrtTableWidgetClass,    parent,
    XmNxrtTblCellValues,      cells,
    NULL);
/* table copies values, so destroy them.
* Don't destroy graph data, since it doesn't copy them */
XrtTblDestroyCellValues(cells);
```

XRT/tableには、XRT/graphとXRT/3dを使用する二つのデモとそのソースコードがあります。*graph_editor* デモは*$XRTHOME/src/common_demos/graph_editor* に、*surface_editor* デモは、*$XRTHOME/src/common_demos/surface_editor*にあります。

6.9 ドラッグ&ドロップのプログラミング

XRT/tableでは、プログラミングによってドラッグ&ドロップの機能を変更したり拡張したりすることができます。この章では、次のプログラミングについて解説します。

- テーブル以外のウィジェットからのドラッグや、これらのウィジェットへのドロップを不許可にする。
- 普通のテキストをテーブルにドラッグする際、またはテーブルから普通のテキストをエキスポートする際に使用する区切り文字を指定する。
- ExportまたはImportコールバックを使用する。
- 独自のドラッグ&ドロップのターゲットとそれに関連する変換手続きを定義する。

6.9.1 ドラッグ&ドロップのターゲット

`XmNxrtTblExportTargets`と`XmNxrtTblImportTargets`リソースを使用して、テーブルで許可されるドラッグ&ドロップの型をコントロールできます。

`XmNxrtTblExportTargets`は、テーブルがドラッグ元として使用されている時に、そのテーブルの動作をコントロールします。このリソースの値は、データ・アイテムのポインタの配列で、それぞれドラッグ操作の一つの型を指定します。このリストの各データ・アイテムは*ターゲット(target)*と呼ばれます。

XmNxrtTblImportTargetsは、テーブルがドロップ先として使用されている時に、そのテーブルの動作をコントロールするターゲットのリストを指定します。

これらのリソースには、XrtTblImportExportStruct構造体のポインタの配列が渡されます。

```
typedef struct {
  String       target;
  XrtTblImportExportConvertProc converter;
} XrtTblImportExportStruct;
```

*target*は、この構造体のメンバで定義されているターゲットの型の名前です。*converter*は、この型のドラッグ&ドロップを実行する時に使用される変換手続きを指定します。変換手続きは、独自のドラッグ&ドロップ・ターゲットを定義している場合や既存の動作を置き換えたい場合に使用します。(変換手続き及びアプリケーションで定義されるドラッグ&ドロップは、6.9.4に解説されています。)

ターゲットには、データ・フォーマットを示すターゲットと情報ターゲットの二つの種類があります。

データ・フォーマットのターゲット

データ・フォーマットのターゲットは、テキストがXRT/tableウィジェットまたはXmText/XmTextFieldウィジェットからドラッグまたは、それらへドロップすることを示します。変換手続きが定義されるのは、これらのターゲットだけです。各ドラッグ&ドロップの操作は、次のターゲットの内一つを定義します。

XRTTBL_TGT_CELLS	**ドラッグ&ドロップの操作は、**XRT/table**のデータ転送フォーマットを使用する。**
XRTTBL_TGT_COMPOUND_TEXT XRTTBL_TGT_TEXT XRTTBL_TGT_STRING	XmText**または**XmTextField**ウィジェット(または、これらのターゲットを認識できる他のウィジェット)からのドラッグとこれらのウィジェットへのドロップ。**

デフォルトでは、これら全てのターゲットの型が、XmNxrtTblImportTargetsとXmNxrtTblExportTargetsリストに含まれています。

情報ターゲット

情報ターゲットは、ドラッグまたはドロップされるデータに関する情報を示します。下記のターゲットが定義されます。

XRTTBL_TGT_RANGE	データはセル範囲。
XRTTBL_TGT_ROWS	データは、一行または複数行(ラベルが含まれることもある)。
XRTTBL_TGT_COLUMNS	データは、一列または複数列(ラベルが含まれることもある)。

XRTTBL_TGT_COLUMNLABELS	データには列ラベルが含まれる。
XRTTBL_TGT_ROWLABELS	データには行ラベルが含まれる。
XRTTBL_TGT_INTERNALMOVE	ドラッグとドロップ操作は、同一テーブル上で行われる。

テーブルからデータがエキスポートされる時、ユーザの選択により、XRTTBL_TGT_RANGE、XRTTBL_TGT_ROWS、XRTTBL_TGT_COLUMNSの内一つが、XmNxrtTblExportTargetsで指定されるリストに、自動的に追加されます。

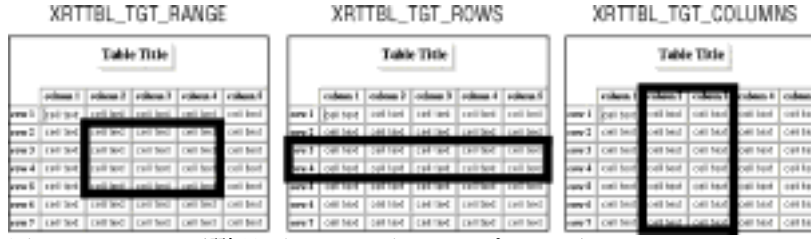


図62　セル/ラベルの選択を示すドラッグ&ドロップのターゲット

デフォルトでは、XRTTBL_TGT_RANGE、XRTTBL_TGT_ROWS、XRTTBL_TGT_COLUMNSは、XmNxrtTblImportTargetsとXmNxrtTblExportTargetsの両方のリストに含まれます。XRTTBL_TGT_INTERNALMOVEは、ドラッグ&ドロップが同じテーブル内で行われる場合は自動的に、XmNxrtTblExportTargetsリストに追加されます。また、XRTTBL_TGT_INTERNALMOVEは、デフォルトでXmNxrtTblImportTargetsリストに含まれます。[1]

ターゲット・リストの手続き

下記は、ドラッグ&ドロップのターゲット・リストを操作する手続きです。

- *XrtTblImportExportListCopy()*は、ターゲット・リストのコピーを作成する。
- *XrtTblImportExportListAdd()*は、リストにターゲットを追加する。
- *XrtTblImportExportListDestroy()*は、ターゲット・リストを破棄する。
- *XrtTblImportExportListDelete()*は、ターゲット・リストからターゲットを削除する。

6.9.2　区切り文字の指定

普通のテキストがテーブルへインポートされる時、XmNxrtTblImportDelimiterリソースが、それぞれのセルに入るテキストを区切る文字列を定義します。デフォルトでは、この文字列は、”\t”(一つのタブ(tab))です。

[1] XRTTBL_TGT_INTERNALMOVEが、XmNxrtTblImportTargetsで指定されるターゲット・リストより削除されていると、同一テーブル内でのドラッグ&ドロップは正しく行われないことがあります。



図63 セルにドロップされた、区切りのあるテキスト

このリソースをXRTTBL_DELIMIT_BY_WHITESPACEにセットすると、スペースとタブの組み合わせが区切り文字列として処理されます。区切り文字が挿入されないようにするには、このリソースをXRTTBL_DELIMIT_NONEにセットします。

XmNxrtTblExportDelimiterリソースは、普通のテキストがテーブルからエキスポートされる時に使用される区切り文字列を定義します。デフォルトでは、"\t"です。区切り文字を指定しない場合は、このリソースをXRTTBL_DELIMIT_NONEにセットします。XmNxrtTblImportDelimiterまたはXmNxrtTblExportDelimiterの値をデフォルトに戻すには、XRTTBL_DELIMIT_DEFAULTまたはNULLにセットしてください。

6.9.3 ExportとImportコールバック

Exportコールバック

テーブルからのドラッグを行っている時に開始されるコールバックを定義することができます。XmNxrtTblExportCallbackで定義されているコールバック・リストにコールバックを追加します。このリストの各コールバックには、次の構造体が渡されます。

```
typedef struct {
  XrtTblReason     reason;
  XEvent       *event;
  int      row, column;
  String       target;
  Boolean      internal;
  XrtTblRange  range;
  unsigned char    operation;
  Boolean      doit;
  Boolean      bell;
} XrtTblExportCallbackStruct;
```

これらのコールバックは、次の場合に開始されます。

- ドラッグ&ドロップを開始するためにマウスのボタンが押された時 (*reason*はXRTTBL_REASON_EXPORT_DRAG_BEGIN)
- ドロップが開始されようとする時 (*reason*はXRTTBL_REASON_EXPORT_TARGET)
- ドロップが完了した時 (*reason*はXRTTBL_REASON_EXPORT_DRAG_END)

これらのコールバックを使用して、次の事が実行できます。

- *operation*メンバ(値はXmCOPYまたはXmMOVEのどちらか)を変更することにより、ドラッグ&ドロップの操作をコピーから移動へ変更する。

- *doit*メンバをFalseにセットして、ドラッグ&ドロップの操作を中止する。(但し、*reason*がXRTTBL_REASON_EXPORT_DRAG_BEGINまたはXRTTBL_REASON_EXPORT_TARGETの時)
- *bell*をTrueにセットして、警告音を鳴らす。

Importコールバック

XmNxrtTblImportCallbackリソースで定義されるコールバックのリストに新たなコールバックを追加することにより、テーブルへドロップする時に開始されるコールバックを定義することができます。各コールバックには、次の構造体が渡されます。

```
typedef struct {
  int     reason;      // Read-only
  XEvent      *event;      // Read-only
  int     row, column; // Read-only
  String      target;      // Read-only
  Boolean     internal;        // Read-only
  unsigned char    operation;
  unsigned char    dropSiteStatus;
  Boolean     insert;
  Boolean     doit;
  Boolean     bell;
} XrtTblImportCallbackStruct;
```

Importコールバックは、次の場合に開始されます。

- ドラッグが行われている時 (*reason*はXRTTBL_REASON_IMPORT_DRAG_MOTION)
- マウス・ボタンが離された時 (*reason*はXRTTBL_REASON_IMPORT_BEGIN)
- ドロップの変換が開始されようとする時 (*reason*はXRTTBL_REASON_IMPORT_TARGET)

operation、*doit*、*bell*、*row*、*columns*、*internal*は、XrtTblExportCallbackStruct構造体のメンバと同じ役割を持ちます。*dropSiteStatus*は、XmVALID_DROP_SITE(ドロップは有効、デフォルトの設定)、またはXmINVALID_DROP_SITE(ドロップは無効)です。*insert*は、普通のテキスト(他のウィジェットからのテキスト、または一つのセルからのテキスト)がセルにドロップされた時に使用されます。これは、True(挿入する、デフォルトの設定)またはFalse(既存データに上書きする)のどちらかにセットできます。

6.9.4 独自のターゲットを定義する

XRT/tableでは、次の手順により、独自のドラッグ&ドロップのターゲットを定義することができます。[1]

1 独自のターゲット名を選ぶ(例: “MY_TARGET”)

2 そのターゲット・フォーマットでデータを読み込む変換手続きを定義する

3 ターゲット名と変換手続きを使用し、XrtTblImportExportStruct型のエレメントを作成する。

```
XrtTblImportExportStruct mytarget =
   (“MYTARGET”, MyTragetProc);
```

[1] ドラッグ&ドロップを変更する場合、*Motif Programming Manual*の第18章の内容を理解している必要があります。

4 *XrtTblImportExportListAdd()*を用いて、importターゲット・リストまたはexportターゲット・リストのどちらかに、新しいターゲットを追加する。

5 XmNxrtTblImportTargetsまたはXmNxrtTblExportTargetsを新しいターゲット・リストにセットする。

作成したexport変換手続きには、次の引数が渡されるようにしてください。

```
Boolean
ExportConversionProcedure(
  Widget       table,
  String       target,  /* Name of target type */
  XrtTblRange  *range,  /* Region of operation */
  String       *processed, /* Targets already processed */
  XtPointer    *data,   /* Returned, converted data */
  unsigned long    *data_len    /* Returned, length of data */
)
```

*table*はデータをエキスポートするテーブル、*target*は前述のターゲットの内の一つ(または独自に指定したexportターゲット)です。*range*は、エキスポートするデータのセル/ラベル範囲、*processed*は既に処理されているターゲットのリストです。*data*はエキスポートされた、返される(変換される)データで、*data_len*はそのデータの長さです。*data*は、*XtMalloc()*を使用して割り当てられ、自動的に解放されるようにしてください。

作成したimport変換手続きには、次の引数が渡されるようにしてください。

```
Boolean
ImportConversionProcedure(
  Widget       table,
  String       target,  /* Name of target type */
  XrtTblContext    *context,    /* Region of operation */
  String       *processed, /* Targets already processed */
  Boolean      insert,  /* Insert data if possible */
  XtPointer    data,    /* Pointer to data to import */
  unsigned long    data_len     /* Length of data to import */
)
```

*table*はデータをインポートするテーブル、*target*は前述のターゲットの内の一つ(または独自に指定したimportターゲット)です。*context*は、データがインポートされるテーブルのcontextです。*processed*は既に処理されているターゲットのリストです。*insert*は、普通のテキスト(他のウィジェットからのテキスト、または一つのセルからのテキスト)がセルにドロップされた時に使用されます。これは、True(挿入する、デフォルトの設定)またはFalse(既存データに上書きする)のどちらかにセットできます。*data*はインポートされたデータで、*data_len*はそのデータの長さです。

変換手続きは、組み込まれているターゲットに指定する必要はありません。これらのターゲットには、必要時にデフォルトの変換手続きが使用されます。

6.9.5 データ転送フォーマット

下記は、XRT/tableが他のテーブル・ウィジェットにドラッグ&ドロップを行う時に使用するデータ転送フォーマットです。

```
DIM (rows) (columns)
CLB (collabel) (collabel) ...
RLB (rowlabel) (rowlabel) ...
XMS (celldata) (celldata) ...
...
```

このファイルの各行のフォーマットは、(一つまたは複数の)括弧に囲まれた値が後に続く3文字のキーワードです。各アイテム(キーワードまたは値)は、一つのスペース(空白)で区切られています。

DIMで始まる一行目は、転送される行と列の数を示しています。列ラベルがドラッグされる場合、CLBがそれを示します。CLBで指定される*collabel*値の数は、DIMで指定される*columns*の数と同じです。同様にして、RLBは行ラベルを示し、*rowlabel*値の数は*rows*の数と同じです。

XMSはセル・データ自身を示します。各XMS行は、転送される各行に対応します。各行の*celldata*エントリの数は、転送される列数と同じです。

例えば、簡単なテーブルとそのデータ転送フォーマットは、下記のようになります。

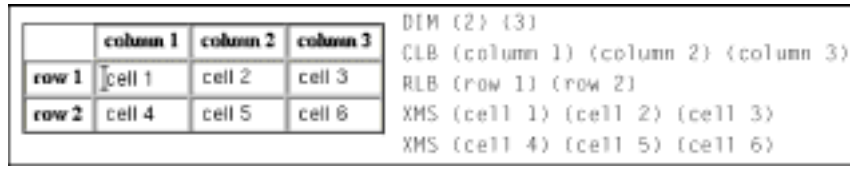


図64　テーブルとそのデータ転送フォーマット

また、このフォーマットを使用して、既存のターゲットの変換手続きを定義することもできます。使用しているターゲット・リストから既存のターゲット定義を削除し、新しい変換手続きを指定するターゲット定義を追加してください。

```
XrtTblImportExportStruct mytextconv =
  {XRTTBL_TGT_COMPOUND_TEXT, myTextRoutine};

XrtTblImportExportListDelete(implist, 0,
  XRTTBL_TGT_COMPOUND_TEXT);
XrtTblImportExportListAdd(implist, 0, &mytextconv);
XtVaSetValues(droptable,
  XmNxrtTblImportTargets, implist,
  NULL);
```

通常は、既存のターゲット型には、変換手続きを定義する必要はありません。

6.10 XRT/tableの国際化

ISO Latin 1文字セット

使用しているXサーバでサポートされているX Windowフォントならどれでもセルやラベルのテキストとして使用できます。ISO Latin1は、北米と西欧の言語のISO標準文字セットです。ISO Latin1は、ASCIIのスーパーセットですので、ISO Latin1フォントを指定してもASCIIアプリケーションの動作は変更されません。名前の最後にiso8859-1がくるX Windowフォントは、ISO Latin1フォントです。

2 byteフォント

漢字等の2 byteフォントもセルやラベルに表示できます。2 byteフォントは、XmStringsでのみサポートされています。

印刷

*XrtTblVaDrawPS()*と*XrtTblDrawPS()*は、ISO Latin1文字セットを使用するPostScriptを出力します。これらの手続きは、日本語以外の2 byteフォントのテキストをプリントできません。

6.11 パフォーマンスについて

大きなデータを扱うテーブルや、属性が頻繁に変更されるテーブルでは、パフォーマンスが落ちることがあります。ここでは、そのような場合に、どのようにパフォーマンスを向上させるかを解説します。

*$XRTHOME/src/table/demos/stocks*ディレクトリにある*stocks*デモには、様々なXRT/tableのタスクを行う際にパフォーマンスを上げる方法の例がありますので参照してください。

6.11.1 一般的な場合のパフォーマンス

- ほとんどのcontextリソースには、対応するseriesリソースがあります。多くの場合、seriesリソースを直接セットした方が、良いパフォーマンスが得られます。seriesリソースを使用するプログラムの例が*$XRTHOME/src/table/demos/form/view.c*にありますので参照してください。
- リソース設定にcontextを定義する際、なるべく大きくcontextを設定し、定義するcontextの数を最小限にしてください。最初にデフォルトのcontext(ALL, ALL)を定義し、その後に各contextを設定すると良い結果が得られます。

  例えば、ほとんどのセルの背景色が青でその他のセルが赤や緑の場合、最初にデフォルトのcontextの背景色を青にセットします。

```
XtVaSetValues(table,
  XmNxrtTblContext,
    XrtTblSetContext(XRTTBL_ALL, XRTTBL_ALL),
  XmNxrtTblBackgroundContext, "blue",
  NULL);
```

  その後、赤や緑のcontextをセットしていきます。

- 可能であれば、一回の*XtVaSetValues()*コールで全てのcontextリソースをセットしてください。
- XmNxrtTblShadowThicknessの値を小さくすると再描画が速くなります。
- セットするcontextリソースの数が多い(異なるリソース設定のセルやラベルが多い)場合、seriesリソースを使用する代りに、独自のcontextリストを管理するようにしてください。6.11.5に解説されている指示に従ってXmNxrtTblContextValueCallbackリソースをセットしてください。

6.11.2 インタラクティブ機能に関するパフォーマンス

- 大きなテーブルでトラバース可能なセルが少ない時、XmNxrtTblTraversableContextをFalseにセットし、XmNxrtTblTraverseCellCallbackを使用してトラバースをカスタマイズすることにより、トラバースのパフォーマンスが向上します。
- XmNxrtTblTrackCursorをFalseにすると、XRT/tableから発生するネットワーク・トラフィックの量が減少します。但し、マウス・ポインタの追跡がオフに設定されますので、ユーザがテーブル上でマウスを動かしてもポインタの形は変更されません。
- クローン化されたウィジェットを多く含むテーブルをスクロールする時は、XRT/tableのプレビュー・スクロールを使用してください。通常のスクロールでは、XRT/tableは多くのウィジェットを作成し位置決めをしていかなければなりませんのでパフォーマンスがかなり落ちてしまいます。一方、プレビュー・スクロールでは、スクロールが完了するまで画面をアップデートしません。

6.11.3 表示に関するパフォーマンス

- デフォルトでは、リソースの一つがセットされる度にXRT/tableは再計算、再描画を行います。再計算、再描画を行う必要がないリソースがある時は、XmNxrtTblRepaintをXRTTBL_REPAINT_NOCALCにセットしてください。(例えば、テーブルに行を追加する時は、XmNxrtTblRepaintをXRTTBL_REPAINT_NOCALCにセットし、*XrtTblAddRows()*を呼び出して行を追加します。その後XmNxrtTblRepaintをXRTTBL_REPAINT_ONに戻します。)
- 可変の高さや幅を持つ行や列を使用する時は、XmNxrtTblPixelHeightContext/XmNxrtTblPixelWidthContextをXRTTBL_VARIABLE以外の値にセットし、セルへ必要な変更を加えてから、XmNxrtTblPixelHeightContext/XmNxrtTblPixelWidthContextをXRTTBL_VARIABLEに戻します。

ダブル・バッファリング

ダブル・バッファリングは、テーブルが変更されたりスクロールされた時に起こる画面のちらつきを減少させる機能です。この機能を利用して、表示へのアップデートは、オフ・スクリーンで行い、その後画面上にコピーすることができます。

XmNxrtTblDoubleBufferがXRTTBL_DOUBLEBUFFER_CELL(デフォルト)にセットされている時は、テーブルはセルまたはラベル毎にダブル・バッファリングが行われます。XRTTBL_DOUBLEBUFFER_WINDOWにセットされている時は、ウィンドウ全体がダブル・バッファリングの対象となります。

XmNxrtTblDoubleBufferをXRTTBL_DOUBLEBUFFER_NONEにセットすると、この機能はオフになります。パフォーマンスは上がりますが、テーブルが変更されたりスクロールされたりしたときは、ちらつきが起こります。

6.11.4 セル値の処理

- *XtVaSetValues()*をXmNxrtTblContextリソースと一緒に使用すると、*XtVaSetValues()*は指定されたcontext内の各セルに対して呼び出されますので、パフォーマンスが落ちます。テーブルの全てのXmStringを記憶するXrtTblCellValues構造体を直接構築してみてください。(*XrtTblReadAscii()*もこの方法の一つです。*simple.c*というプログラムにその例がありますので参照してください。)
- セル値のセットには、XrtTblCellValues構造体の代りに独自のコールバックを使用してください。XmNxrtTblCellValueCallbackは、4.5に解説されています。
- XmNxrtTblXmStringSetModeがXRTTBL_SET_STRINGにセットされていると、XrtTblCellValues構造体のセル値は、内部的にStringとして記憶されます。この方法は、計算や描画にかかる時間をかなり短くすることができます。
- XmNxrtTblPixelHeightContext/XmNxrtTblPixelWidthContextを明示的にpixel数で指定します。または、
- XmNxrtTblCharHeightContext/XmNxrtTblCharWidthContextを明示的に行数または文字数で指定します。
- ひとつのセルを素早くアップデートするには、*XrtTblSetCell()*を使用してください。このルーチンは、XmNxrtTblPixelWidthContextがXRTTBL_VARIABLEでも新しいデータに合うようセルをリサイズしません。

6.11.5 contextを管理する

ContextまたはSeriesリソースを使用する代りに、アプリケーションはcontextを独自に管理することができます。異なる設定を持つセルの数が多い場合等には、パフォーマンスがよくなります。

実行時にcontext値をコントロールするには、コールバックを定義し、XmNxrtTblContextValueCallbackで指定されるリストに追加してください。このリストの各コールバックは、セルやラベルのcontext値が必要な時(テーブルの大きさの計算や再描画時)に呼び出されます。

各XmNxrtTblContextValueCallbackルーチンには、次の構造体が渡されます。

```
typedef struct {
  int          reason;      // Read-only
  int          row, column; // Read-only
  XEvent           *event;      // Read-only
  XrtTblAction         action;      // Read-only
  XrtTblContextReason  context_reason;  // Read-only
  XrtTblContextValue   value;
} XrtTblContextValueCallbackStruct;
```

*action*メンバは、XRTTBL_ACTION_CALCULATE(コールバックは、サイズの計算やテーブルの初期化時に呼び出される)、またはXRTTBL_ACTION_DRAW(コールバックはテーブルの再描画時に呼び出される)のどちらかです。

*row*と*column*メンバは、現在チェックされているセルまたはラベルを指します。

*context_reason*は、どのcontext値が現在セルに必要であるか示します。各contextのreasonは、XRT/tableのcontextリソースと対応します。*value*は、このcontextのreasonに対する現在の値で、XrtTblContextValue型です。コールバックは、*value*を変更することにより、セルに対するcontext値を変更することができます。(*value*は使用後解放されず、必ず静的ストーレッジを指すようにしておいてください。)

XrtTblContextValueはunionとして定義されます。各contextのreasonは、unionのメンバに対応します。このunionメンバの型は、そのcontextのreasonに対応するcontextリソースの値と同じ型です。

下の表は、*context_reason*の値、各contextのreasonに対応するXRT/tableのリソース、そのcontextのreasonに関連するXrtTblContextValue unionのメンバ名の一覧です。

Context Reason	対応するXRT/table Resource	union menber名
XRTTBL_CONTEXT_ALIGNMENT	XmNxrtTblAlignmentContext	alignment
XRTTBL_CONTEXT_ARGLIST	XmNxrtTblArgListContext	arglist
XRTTBL_CONTEXT_BACKGROUND	XmNxrtTblBackgroundPixelContext	background
XRTTBL_CONTEXT_BORDERSIDES	XmNxrtTblBorderSidesContext	border_sides
XRTTBL_CONTEXT_BORDERTYPE	XmNxrtTblBorderTypeContext	border_type
XRTTBL_CONTEXT_BOTTOMSHADOW	XmNxrtTblBottomShadowContext	bottom_shadow
XRTTBL_CONTEXT_CASE	XmNxrtTblCaseContext	string_case
XRTTBL_CONTEXT_CHAR_HEIGHT	XmNxrtTblCharHeightContext	char_height
XRTTBL_CONTEXT_CHAR_WIDTH	XmNxrtTblCharWidthContext	char_width
XRTTBL_CONTEXT_CLIP_PIXMAP	XmNxrtTblClipPixmapContext	clip_pixmap
XRTTBL_CONTEXT_CURRENCYPICK LIST	XmNxrtTblCurrencyPick listContext	pick list
XRTTBL_CONTEXT_DATATYPE	XmNxrtTblDataTypeContext	datatype
XRTTBL_CONTEXT_DATEPICK LIST	XmNxrtTblDatePick listContext	pick list
XRTTBL_CONTEXT_EDITABLE	XmNxrtTblEditableContext	editable
XRTTBL_CONTEXT_FIELD_ARGLIST	XmNxrtTblFieldArgListContext	field_arglist
XRTTBL_CONTEXT_FLOATPICK LIST	XmNxrtTblFloatPick listContext	pick list
XRTTBL_CONTEXT_FOREGROUND	XmNxrtTblForegroundPixelContext	foreground
XRTTBL_CONTEXT_INTPICK LIST	XmNxrtTblIntPick listContext	pick list

Context Reason	対応するXRT/table Resource	union menber名
XRTTBL_CONTEXT_MAXLENGTH	XmNxrtTblMaxLengthContext	maxlength
XRTTBL_CONTEXT_MULTILINE	XmNxrtTblMultilineContext	multiline
XRTTBL_CONTEXT_PIXEL_HEIGHT	XmNxrtTblPixelHeightContext	pixel_height
XRTTBL_CONTEXT_PIXEL_WIDTH	XmNxrtTblPixelWidthContext	pixel_width
XRTTBL_CONTEXT_PIXMAP	XmNxrtTblPixmapContext	pixmap
XRTTBL_CONTEXT_PIXMAP_LAYOUT	XmNxrtTblPixmapLayoutContext	pixmap_layout
XRTTBL_CONTEXT_PIXMAP_SHADOW	XmNxrtTblPixmapShadowContext	pixmap_shadow
XRTTBL_CONTEXT_PS_FONTLIST	XmNxrtTblPSFontListContext	ps_fontlist
XRTTBL_CONTEXT_SENSITIVE	XmNxrtTblSensitiveContext	sensitive
XRTTBL_CONTEXT_STRINGPICK LIST	XmNxrtTblStringPick listContext	pick list
XRTTBL_CONTEXT_TOP_SHADOW	XmNxrtTblTopShadowContext	top_shadow
XRTTBL_CONTEXT_USERDATA	XmNxrtTblUserDataContext	user_data
XRTTBL_CONTEXT_WIDGET_SHADOW	XmNxrtTblWidgetShadowContext	widget_shadow
XRTTBL_CONTEXT_WORDWRAP	XmNxrtTblWordWrapContext	wordwrap

例えば、次のContextValueコールバックは、テーブルの3行目の高さを10文字にセットしています。

```
void
ContextValueCB(Widget table, XtPointer data,
  XrtTblContextValueCallbackStruct *cs);
{
  if (cs->context_reason == XRTTBL_CONTEXT_CHAR_HEIGHT &&
      cs->row == 2) {
      cs->value.char_height = 10;
  }
}
```

*char_height*メンバの型は、`XmNxrtTblCharHeightContext`リソースのデータ型(`int`)と同じです。

context値によっては、`XmNxrtTblContextValueCallback`ルーチンの中からセットできないものもあります。下記は、ContextValueコールバックでセットできないcontextリソースです。(対応するcontextのreasonがありません。)

```
XmNxrtTblCellValueContext
XmNxrtTblFontListContext
XmNxrtTblOverflowContext
XmNxrtTblTraversableContext
```

セル値は、`XmNxrtTblCellValueCallback`を利用してセットすることができます。(4.5参照) 上記のその他のcontext値は、contextまたはseriesリソースを使用してセットしてください。

6.12 警告とエラー・メッセージのカスタマイズ

XRT/tableは、警告とエラー・メッセージの出力に*XtAppWarningMsg()*を使用しています。Xtエラー・データベースでは、アプリケーションとは無関係に、エラーや警告のテキストをカスタマイズすることができます。XRT/tableのメッセージは、*$XRTHOME/include/Xm/XrtTable.err*

にあります。

警告とエラー・メッセージを変更するには、このファイルのコピーを修正し、次のようにしてください。

- */usr/lib/X11/XtErrorDB*にあるシステムのメッセージ・ファイルに、修正したメッセージを追加する。または、
- Xlib手続きの*XrmMergeDatabases()*を呼び出して、修正したファイルとシステムのファイルをマージする。

6.13 C++の使用

XRT/tableは、C++インターフェースを提供しています。C++を使用するには、C++プログラムに、ヘッダー・ファイル*$XRTHOME/include/Xm/XrtTable.hxx*をインクルードしてください。このファイルは、以下の方法でXrtTableクラスを定義します。

- *XmCreateXrtTable()*を呼び出すconstructor。
- *XtDestroyWidget()*を呼び出すdestructor。
- XRT/tableウィジェットを始めの引数として取る各Cの手続きに対応する方法 (例: *XrtTblAddRows()*の方法は*addRows()*)
- 各設定可能なリソースをセットする方法。(例: `XmNxrtTblBackgroundContext`をセットする方法は、*setBackgroundContext()*)
- 各リソースを取得する方法。(例: `XmNxrtTblBackgroundContext`を取得する方法は、*getBackgroundContext()*)
- 一般的なXtのファンクションを実行する方法(*addCallback()*、 *destroy()*、 *getDisplay()*、*getName()*、*getParent()*、*isManaged()*、*isRealized()*、*manageChild()*)

C++を使用する例が、*$XRTHOME/src/examples/simple_cpp.cxx*にあります。以下は、XRT/tableでのC++の使用例を示しています。

```
XrtTable *table = new XrtTable(toplevel, "table", NULL, 0);
Table->setNumColumns(5);
Table->setBackgroundContext("red");
Table->addCallback(XmNxrtTblEnterCellCallback, enterCB, (XtPointer)
table);
table->setContext(3, 4);
printf("name: %s, color: %s\n",
        table->getName(), table->getBackgroundContext());
```

6.14 UILの使用

XRT/tableでは、OSFのUser Interface Language(UIL)を使用してテーブル・ウィジェットを作成することができます。UILは標準のMotifウィジェットに使用されるデータ型のみ

をサポートしていますので、XRT/tableの幾つかのリソースはUILで設定できませんが、MrmNcreateCallbackを用いてアプリケーションで設定することができます。UILの引数とデータ型についてはAppendix Hをご覧ください。

UILファイルでセットされるContextリソースは、テーブル全体に適用されます。UILでテーブルを作成するには、次のようにしてください。

- アプリケーションに、XRT/tableのUIL ヘッダーファイル*XrtTable.uil*をインクルードする。
- *MrmInitialize()*のコールの直後に、*XrtTblMrmInitialize()*を呼び出す。

UILのサンプル・プログラム、*styles_uil.c*と*styles_uil.uil*が、*$XRTHOME/src/table/examples*にあります。

6.15 リソース・ファイル

XRT/tableには、テーブルのリソースとその現在の値を保管したり読んだりする機能が提供されています。

リソース値を書く

*XrtTblWriteResources()*は、テーブルのリソースをファイルに書きます。

```
Widget table;
FILE *fp;

if (!(fp = fopen("mytable.res", "w"))) {
  perror("mytable.res");
  exit(0);
}
XrtTblWriteResources(table, fp, "*.table", False);
fclose(fp);
```

*XrtTblWriteResources()*には4つの引数が必要です。最初の二つはテーブル・ウィジェットとファイルのポインタです。

3番めの引数は、各リソース名の前に書かれるストリングです。これを利用して、複数のテーブルのリソースを一つのファイルに保管することができます。この引数で指定されるストリングは、各リソース名の前に書かれます。例えば、"*.table"が指定される場合、*XrtTblWriteResources()*はXmNxrtTblNumColumnsの値を次のように書きます。

```
*.table.xrtTblNumColumns:  6
```

ストリングが指定されていない(3番めの引数がNULL)の場合は、"*."が各リソース名の前に書かれます。

4番めの引数は、継承されたリソース(XmNwidth等)を書くかどうかを指定します。

リソース値を読む

*XrtTblWriteResources()*で書かれたリソース設定を読むには、*XrtTblReadResources()*を使用します。

```
XrtTblReadResources(table, “mytable.res”);
```

この手続きは、全てのリソースをアプリケーション内でマッチするテーブルに適用します。

Part II

Reference Appendices

A

リソースの一覧

ウィジェットの概要 ■ *XRT/tableのリソース*
Constraint リソース ■ *継承されたリソース*

XRT/tableの全てのリソースのアルファベット順の一覧です。リソース名の後は、データ型、デフォルト値、そしてリソースと共に使用される手続きです。**C**は、そのリソースがcreate(例: *XtCreateWidget()*)で定義でき、**S**はset(例: *XtSetValues()*)でき、**G**はget(例: *XtGetValues()*)で使用できることを意味します。

A.1 ウィジェットの概要

インクルード・ファイル	*<Xm/XrtTable.h>*
クラス名	XtXrtTable
クラス階層	Core→Composite→Constraint→XmManager→XtXrtTable
クラス・ポインタ	xtXrtTableWidgetClass
インスタンスの生成	`widget = XtCreateWidget(name, xtXrtTableWidgetClass ...)`
関数/マクロ	*XmCreateXrtTable(), XtIsXrtTable()*

A.2 XRT/tableのリソース

XmNxrtTblAlignmentContext XrtTblAlignment XRTTBL_ALIGNMENT_TOPBEGIN CSG

現在のcontextの、セルとラベルのテキストの行揃えを指定します。有効な値は次の通りです。[1]

XRTTBL_ALIGNMENT_TOPBEGIN	XRTTBL_ALIGNMENT_TOPCENTER	XRTTBL_ALIGNMENT_TOPEND
XRTTBL_ALIGNMENT_MIDDLEBEGIN	XRTTBL_ALIGNMENT_MIDDLECENTER	XRTTBL_ALIGNMENT_MIDDLEEND
XRTTBL_ALIGNMENT_BOTTOMBEGIN	XRTTBL_ALIGNMENT_BOTTOMCENTER	XRTTBL_ALIGNMENT_BOTTOMEND

継承されたXmNstringDirectionリソースは、セルのどちら側からテキストが開始されるかを指定します。(右から左へ読むロケールに使用されます。)

XmNxrtTblAlignmentSeries XrtTblSeries* NULL CSG

XrtTblIntValue構造体のseriesとして、テキストの行揃えを指定します。

XmNxrtTblAllowCellResize XrtTblResize XRTTBL_RESIZE_ALL CSG

ユーザによるインタラクティブな行や列のリサイズを指定します。有効な値は、XRTTBL_RESIZE_ALL 、 XRTTBL_RESIZE_NONE 、 XRTTBL_RESIZE_HORIZONTAL 、 XRTTBL_RESIZE_VERTICALです。

XmNxrtTblAllowResize XrtTblResize XRTTBL_RESIZE_NONE CSG

セル/ラベルのサイズに影響するリソースが変更された時、どのようにテーブル・ウィジェットがリサイズするかを指定します。XRTTBL_RESIZE_NONEのデフォルトになっている時は、アプリケーションがXmNxrtTblVisibleRowsまたはXmNxrtTblVisibleColumnsをセットした場合にのみ、テーブルはリサイズします。XRTTBL_RESIZE_HORIZONTAL、XRTTBL_RESIZE_VERTICAL、XRTTBL_RESIZE_ALLにセットされている時は、セル/ラベルのサイズを変更するリソース(XmNxrtTblMarginWidth等)の変更によってテーブルは横、縦または両方向にリサイズします。このリソースは、テーブルがrealizeされた後にのみ使用できます。図60に、テーブルのリサイズの原因となるリソースのリストがあります。

XmNxrtTblArgListContext ArgList NULL CSG

現在のcontextの、ウィジェットのArgListを指定します。このArgListの最後のエレメントはNULLです。ArgList値は、DisplayWidgetコールバックを呼び出す前に、セルまたはラベル内のウィジェット・セットに適用されます。このリソース値がポインタの時は、必ず静的または割り当てられたメモリを参照するようにします。

XmNxrtTblArgListSeries XrtTblSeries* NULL CSG

ウィジェットのArgListをXrtTblArgListValue構造体のseriesとして指定します。他のseriesとは異なり、これらは内部的にコピーされません。

[1] これらの位置決めは、XmNstringDirectionがXmSTRING_DIRECTION_L_TO_Rであると仮定した場合です。

XmNxrtTblBackgroundContext String "Inherited" CSG

現在のcontextの、セルとラベルの背景色を指定します。値は、Xカラー・データベースにある色またはRGBの16進数です。"Inherited"にセットされている場合、CoreのXmNBackgroundから色が継承されます。

XRTTBL_REPEAT_ROWまたはXRTTBL_REPEAT_COLUMNにセットされている時は、色は、XmNxrtTblRepeatBackgroundColorsで指定されるリストから、順番に繰り返して適用されます。

モノクロのサーバで、このリソースを黒以外の色に指定すると白に表示されます。

XmNxrtTblBackgroundContextがセットされると、XmNxrtTblBackgroundPixelContextは自動的に変更されます。

XmNxrtTblBackgroundPixelContext Pixel XRTTBL_NOVALUE CSG

現在のcontextの、セルの背景色をpixelで指定します。値がXRTTBL_NOVALUEの時は、このリソースは無視されます。このリソースがウィジェット作成時にセットされると、影の上部と下部の色が計算されます。ウィジェット作成後にセットされた場合は、影の色は変更されません。[2]

XRTTBL_REPEAT_ROW_PIXELまたはXRTTBL_REPEAT_COLUMN_PIXELにセットされると、色は、XmNxrtTblRepeatBackgroundColorsで指定されるリストから、順番に繰り返して適用されます。

XmNxrtTblBackgroundPixelContextとXmNxrtTblBackgroundContextが同じ*XtVaSetValues()*のコールでセットされた場合、後者は無視されます。

XmNxrtTblBackgroundPixelContextを変更すると、自動的にXmNxrtTblBackgroundContextが変更されます。(もし、新しく指定された色が色名に対応しない場合は、XmNxrtTblBackgroundContextは、16進数のRGB値になります。)

XmNxrtTblBackgroundSeries XrtTblSeries* NULL CSG

XrtTblColorValue構造体のseriesとして、セルとラベルの背景色を指定します。

XmNxrtTblBorderSidesContext XrtTblBorderSides XRTTBL_BORDERSIDE_ALL CSG

現在のcontextの、セルまたはラベルの罫線が描かれる場所を指定します。値は、以下のbitwise ORです。

```
XRTTBL_BORDERSIDE_NONE
XRTTBL_BORDERSIDE_ALL
XRTTBL_BORDERSIDE_LEFT
XRTTBL_BORDERSIDE_RIGHT
XRTTBL_BORDERSIDE_TOP
XRTTBL_BORDERSIDE_BOTTOM
```

罫線の幅は、XmNxrtTblShadowThicknessで指定できます。

XmNxrtTblBorderSidesSeries XrtTblSeries* NULL CSG

XrtTblIntValue構造体のseriesとして、罫線の描かれる場所を指定します。リソース・ファイルでは、下の例のように、シンボルの間に"+"を挿入してください。

```
*.xrtTblBorderSidesSeries: (ALL ALL BORDERSIDE_LEFT+BORDERSIDE_TOP)
```

[2] 罫線の影の色は、XmNxrtTblBottomShadowContextとXmNxrtTblTopShadowContextで指定されます。

XmNxrtTblBorderTypeContext XrtTblBorderType XRTTBL_BORDER_IN CSG

現在のcontextの、セルまたはラベルの罫線のスタイルを指定します。有効な値は次の通りです。

XRTTBL_BORDER_ETCHED_IN	セルが沈んでいるように見える二重線
XRTTBL_BORDER_ETCHED_OUT	セルが浮き上がっているように見える二重線
XRTTBL_BORDER_FRAME_IN	セルの周囲の1 pixel幅の凹枠
XRTTBL_BORDER_FRAME_OUT	セルの周囲の1 pixel幅の凸枠
XRTTBL_BORDER_IN	セルが沈んでいるように見える線
XRTTBL_BORDER_OUT	セルが浮き上がっているように見える線
XRTTBL_BORDER_PLAIN	前景色で描かれる装飾無しの罫線
XRTTBL_BORDER_NONE	罫線なし

罫線の幅は、XmNxrtTblShadowThicknessで指定できます。

XmNxrtTblBorderTypeSeries XrtTblSeries* NULL CSG

XrtTblIntValue構造体のseriesとして、罫線のスタイルを指定します。

XmNxrtTblBottomShadowContext Pixel XRTTBL_NOVALUE CSG

現在のcontextの、下部の影の色を指定します。XRTTBL_NOVALUEにセットされている時は、XRT/tableは、XmNxrtTblBackgroundContextより適当な色を自動的に計算します。このリソースは、必ずMotif Pixel色値にセットしてください。

このリソースには対応するSeriesリソースがありません。(XRT/tableは、影の色をXmNxrtTblBackgroundSeriesに記憶します。)

XmNxrtTblCaseContext XrtTblCase XRTTBL_CASE_AS_IS CSG

現在のcontextの、ユーザによって編集またはトラバースされたセルの値の大文字・小文字を指定します。既存のセル値はこのリソースによって変更されません。XRT/tableはXmNmodifyVerifyCallbackとANSIマクロのtoupperとtolowerを使用して、入力された文字を変換します。(これらのマクロの働きは、ロケールによって異なります。)このリソースは、セルのデータ型がストリング(XmNxrtTblDataTypeContextがXRTTBL_TYPE_STRING)の時にのみ使用されます。有効な値は以下の通りです。

XRTTBL_CASE_LOWER	小文字に変換
XRTTBL_CASE_AS_IS	変換しない
XRTTBL_CASE_UPPER	大文字に変換

XmNxrtTblCaseSeries XrtTblSeries* NULL CSG

XrtTblIntValue構造体のseriesとして、大文字・小文字変換を指定します。

XmNxrtTblCellValueCallback　XtCallbackList　NULL　CSG

テーブルが空白(XrtTblCellValues構造体に値がない)セルを表示する時に呼び出されるコールバックのリストを指定します。コールバックは、ウィジェットの初期表示、スクロール、エクスポーズの際に呼び出されます。各コールバックには、XrtTblCellValueCallbackStructのポインタが渡されます。

```
typedef struct {
  XrtTblReason     reason;          // Read-only
  XEvent           *event;          // Read-only
  int          row, column; // Read-only
  XmString         string;
  Boolean      store;           //initially False
  Boolean      free_string; //initially False
XrtTblCellValueCallbackStruct;
```

*reason*はXRTTBL_REASON_CELLVALUEにセットされます。*row*と*column*はセルの位置を指定、*string*はコールバック・ルーチンによって返されたセル値です。*store*は、ストリングがテーブルに記憶される場合はTrue(この場合、このセルに対してはコールバックはもう呼び出されません)。*free_string*がTrueであれば、*string*は表示後に解放されます。

デフォルトでは、*string*はXmStringとStringの両方の型の値を認識できます。(但し、Motif 2.xにリンクしていない場合) XmNxrtTblXmStringSetModeによって、*string*がString型かXmString型かのどちらかに制限できます。

このリソースがセットされている場合、XmNxrtTblCellValueFuncは無視されます。

XmNxrtTblCellValueContext　XmString　NULL　CSG

現在のcontextのセル値を指定します。このデータ型はXmStringですが、Stringも指定できます。(Motif 1.2にリンクしている場合) XRT/table は *XmStringCreateLtoR()* とXmSTRING_DEFAULT_CHARSETを使用して内部でXmStringに変換します。また、XmNxrtTblXmStringSetModeの指定によっては、Stringを指定することもできます。

このリソースを設定する時、セル値はcontextの各セルにコピーされます。このリソースを取得する時は、カレントcontextは一つのセルにしてください。

XmNxrtTblCellValueFunc　XrtTblXmStringProc　NULL　CSG

XrtTblCellValues構造体に値がないセルを表示する時に呼ばれるルーチンを指定します。ルーチンは、ウィジェットの初期表示、スクロール、エクスポーズの時に呼び出されます。このルーチンは、次のプロトタイプを持っています。

```
XmString cell_value_func(Widget table, int row, int column)
```

*table*は、XRT/tableウィジェットです。戻り値は、XmNxrtTblXmStringSetModeの値に依り、StringまたはXmStringのどちらかです。返されたストリングは、表示後、ウィジェットによって解放されます。この関数の登録には、*XtSetValues()*を使用し、*XtAddCallback()*は使用しないでください。

XmNxrtTblCellValueCallbackがセットされていると、このリソースによって指定されているルーチンは無視されます。

注意: XmNxrtTblCellValueFuncは、次からのバージョンではサポートされない可能性がありますので、できるだけ、XmNxrtTblCellValueCallbackを使用するようにしてください。

XmNxrtTblCellValues XrtTblCellValues* NULL CSG

セル値を持つXrtTblCellValues構造体を指します。構造体がStringを含む場合は、内部でXmStringに変換されます。セル範囲の値は、*XrtTblSetCellValues()*を使用してセットできます。XrtTblCellValuesは最適化されるので、値を取り出す時は、CELL_VALUEマクロを使用し、インデックスを使用しないでください。

XmNxrtTblCharHeightContext int 1 CSG

現在のcontextの、行に表示されるテキストの行数を指定します。セルまたはラベルに表示しきれないテキストは途中で切り取られます。このリソースは、XmNxrtTblPixelHeightContextがXRTTBL_NOVALUEの時に、セルの高さをコントロールします。このリソースを設定する時は、contextは行全体(x, XRTTBL_ALL)にしてください。このリソースは、固定幅のフォントを使用する時に効果があります。

ユーザがインタラクティブに行をリサイズすると、このリソースはpixelの高さに置き換えられ、XmNxrtTblPixelHeightContextが、そのpixelの高さにアップデートします。

XmNxrtTblCharHeightSeries XrtTblSeries* NULL CSG

XrtTblIntValue構造体のseriesとして、行に表示されるテキストの行数を指定します。

XmNxrtTblCharWidthContext int 10 CSG

現在のcontextの、列に表示される文字数を指定します。セルまたはラベルに表示しきれないテキストは切り取られます。このリソースは、XmNxrtTblPixelWidthContextがXRTTBL_NOVALUEの時にセル幅をコントロールします。このリソースを設定する時は、contextは列全体(XRTTBL_ALL, x)にしてください。このリソースは、固定幅のフォントを使用する時に効果があります。

ユーザがインタラクティブに列をリサイズすると、このリソースはpixel幅に置き換えられ、XmNxrtTblPixelWidthContextが、そのpixel幅にアップデートします。

XmNxrtTblCharWidthSeries XrtTblSeries* NULL CSG

XrtTblIntValue構造体のseriesとして、列に表示される文字数を指定します。

XmNxrtTblClipChildren Widget* NULL G

セルやラベルのビュー領域を切り取る子ウィジェットのNULL終端リストを指定します。トランスレーションやイベント・ハンドラは、これらのウィジェットにセットしてください。

XmNxrtTblClipChildrenPersistent Boolean False CSG

Clip子ウィジェットを永久に作成しておくかどうか指定します。Falseの場合は、必要に応じて作成されたり破棄されたりします。XRT/tableが破棄される前にポップダウンする可能性があるポップアップ・シェルにXRT/tableを配置する場合は、XmNxrtTblClipChildrenPersistentをTrueにしておいてください。

XmNxrtTblClipPixmapContext Pixmap XmUNSPECIFIED_PIXMAP CSG

現在のcontextで、pixmapを描画する時に使用するclipマスクを指定します。(pixmap自身は`XmNxrtTblPixmapContext`によって指定され、clipマスクは、このpixmapをどれだけ表示するか指定します。)

XmNxrtTblClipPixmapSeries XrtTblSeries* NULL CSG

clipマスクを`XrtTblPixmapValue`構造体のseriesとして指定します。

XmNxrtTblColumnLabelOffset int 0 CSG

列ラベルとセル領域の間のスペースをpixelで指定します。負の値が指定されると、ラベルはセル領域に重なります。

XmNxrtTblColumnLabelPlacement XrtTblRowPlacement XRTTBL_PLACE_TOP CSG

スクリーン上の列ラベルの位置を指定します。有効な値は`XRTTBL_PLACE_TOP`と`XRTTBL_PLACE_BOTTOM`です。

XmNxrtTblColumnLabels XrtTblXmStringList* NULL CSG

全ての列ラベルを指定します。データ型がStringなら内部でXmStringに変換されます。`XmNxrtTblXmStringSetMode`の設定に依っては、Stringを指定することもできます。

XmNxrtTblContext XrtTblContext* ALL ALL CSG

`XmNxrtTblRangeContext`がセットされていなければ、現在のcontextを指定します。後に続くContextリソースの設定は、このcontextに適用されます。[1] このリソースが有効なcontextに設定されると、`XmNxrtTblRangeContext`は、副作用で`NULL`にセットされます。

contextは、*row*と*column*インデックスで指定されます。下記は、特別な行/列の値です。

`XRTTBL_ALL`	行または列の全てのセルとラベル
`XRTTBL_LABEL`	行または列のラベル
`XRTTBL_ALLCELLS`	行または列の全てのセル(ラベルは含まれない)

*row*と*column*の両方が`XRTTBL_ALL`であれば、contextはテーブル全体にセットされます。以下は、contextの例です。

`1 2`	セル(1, 2)
`XRTTBL_ALL 1`	列1の全てのセルとラベル
`2 XRTTBL_ALL`	行2の全てのセルとラベル
`XRTTBL_ALL XRTTBL_LABEL`	全ての行ラベル

[1] contextとContextリソースに関する詳細は、2.4を参照してください。

contextは、XmNxrtTblRangeContextを用いて範囲にセットすることもできます。

このリソースをセットする時は*XrtTblSetContext()*を使用してください。XmNxrtTblContextはリソース・ファイルではセットできません。

XmNxrtTblContextValueCallback　XtCallbackList　NULL　CSG

テーブルのサイズの計算や描画の際に、セルやラベルのcontext値が必要な時に呼び出されるコールバックのリストを指定します。各ルーチンには、XrtTblContextValueCallbackStructのポインタが渡されます。

```
typedef struct {
  int               reason;           // Read-only
  int               row, column;  // Read-only
  XEvent                *event;           // Read-only
  XrtTblAction      action;           // Read-only
  XrtTblContextReason    context_reason;    // Read-only
  XrtTblContextValue     value;
} XrtTblContextValueCallbackStruct;

typedef union {
  XrtTblAlignment       alignment;
  ArgList               arglist;
  Pixel                 background;
  XrtTblBorderSides border_sides;
  XrtTblBorderType      border_type;
  Pixel                 bottom_shadow;
  int               char_height;
  int               char_width;
  Pixmap                clip_pixmap;
  XrtTblDataType        datatype;
  Boolean               editable;
  ArgList               field_arglist;
  Pixel                 foreground;
  int               maxlength;
  Boolean               multiline;
  XtPointer             picklist
  int               pixel_height;
  int               pixel_width;
  Pixmap                pixmap;
  XrtTblPixmapLayout    pixmap_layout;
  Boolean               pixmap_shadow;
  String                ps_fontlist;
  Boolean               sensitive
  Pixel                 top_shadow;
  XtPointer             userdata;
  Boolean               widget_shadow;
  Boolean               wordwrap;
} XrtTblContextValue;
```

*reason*は常にXRTTBL_REASON_CONTEXTVALUE、*event*はNULL、*action*はXRTTBL_ACTION_CALCULATEまたはXRTTBL_ACTION_DRAWです。*context_reason*は、どの値が必要とされているか指定します。*value*は対応するcontextに使用される値で、seriesから、またはCellValue/LabelCreateコールバックから取得します。値は使用後に解放されません。また、値は静的ストーレッジを指すようにしておいてください。

*context_reason*は、次の内の一つです。

XRTTBL_CONTEXT_ALIGNMENT	XRTTBL_CONTEXT_FOREGROUND
XRTTBL_CONTEXT_ARGLIST	XRTTBL_CONTEXT_MAXLENGTH
XRTTBL_CONTEXT_BACKGROUND	XRTTBL_CONTEXT_MULTILINE
XRTTBL_CONTEXT_BORDERSIDES	XRTTBL_CONTEXT_PIXEL_HEIGHT
XRTTBL_CONTEXT_BORDERTYPE	XRTTBL_CONTEXT_PIXEL_WIDTH
XRTTBL_CONTEXT_BOTTOMSHADOW	XRTTBL_CONTEXT_PIXMAP
XRTTBL_CONTEXT_CASE	XRTTBL_CONTEXT_PIXMAP_LAYOUT
XRTTBL_CONTEXT_CHAR_HEIGHT	XRTTBL_CONTEXT_PIXMAP_SHADOW
XRTTBL_CONTEXT_CHAR_WIDTH	XRTTBL_CONTEXT_PS_FONTLIST
XRTTBL_CONTEXT_CLIP_PIXMAP	XRTTBL_CONTEXT_SENSITIVE
XRTTBL_CONTEXT_CURRENCYPICKLIST	XRTTBL_CONTEXT_STRINGPICKLIST
XRTTBL_CONTEXT_DATATYPE	XRTTBL_CONTEXT_TOP_SHADOW
XRTTBL_CONTEXT_DATEPICKLIST	XRTTBL_CONTEXT_USERDATA
XRTTBL_CONTEXT_EDITABLE	XRTTBL_CONTEXT_WIDGET_SHADOW
XRTTBL_CONTEXT_FIELD_ARGLIST	XRTTBL_CONTEXT_WORDWRAP
XRTTBL_CONTEXT_FLOATPICKLIST	

XmNxrtTblCreateWidgetCallback XtCallbackList NULL CSG

セル/ラベル内のウィジェットを作成/クローン化する前後に、アプリケーションに通知するために呼び出されるコールバックのリストを指定します。各コールバックには、XrtTblCreateWidgetCallbackStructのポインタが渡されます。このリソースの使用法は6.1.2に解説されています。以下はこのコールバックのフォーマットです。

```
void
CallbackProcedure(
    Widget       table,
    XtPointer    client_data,
    XtPointer    call_data    // points to an
    )                         // XrtTblCreateWidgetCallbackStruct
```

XrtTblCreateWidgetCallbackStructは次のように定義されます。

```
typedef struct {
  XrtTblReason    reason;          // Read-only
  XEvent          *event;          // Read-only
  int             row, column; // Read-only
  Widget          parent;          // Read-only
  WidgetClass     widget_class;    // Read-only
  String          class_name, name; // Read-only
  Widget          clone_source;    // Read-only
  Widget          new_widget;
} XrtTblCreateWidgetCallbackStruct;
```

reason は XRTTBL_REASON_CREATE_WIDGET_BEGIN または XRTTBL_REASON_CREATE_WIDGET_ENDのどちらかです。*row*、*column*はウィジェットを持つセルまたはラベルの位置を指定。*parent*は親ウィジェット。*widget_class*はウィジェットのクラス・ポインタ(XRT/tableがウィジェットを認識しない時はNULL）。*class_name*と*name*は、ウィジェットのクラスとインスタンス名。*clone_source*はクローン化されるウィジェット。(ウィジェットがクローンでない時はNULL）。*new_widget*はセットされるウィジェット(*reason*がXRTTBL_REASON_CREATE_WIDGET_ENDの場合にのみ有効)。

XmNxrtTblCurrencyPickListContext double * NULL CSG

現在のcontextに対して、有効な値のリストを指定します。最初のエレメントの値は、必ずリストの残りのエレメントの数にしてください。

XmNxrtTblCurrencyPickListSeries XrtTblSeries * NULL CSG

XrtTblPickListValue構造体のseriesとしてpicklistを指定します。

XmNxrtTblCursorCorner Cursor XC_bottom_right_corner CSG

XmNxrtTblAllowCellResizeがXRTTBL_RESIZE_ALLの場合に、セル/ラベルの右下に表示されるマウスのカーソルを指定します。デフォルトのカーソルはXC_bottom_right_cornerシンボルから作成されます。

XmNxrtTblCursorHorizontal Cursor XC_right_side CSG

XmNxrtTblAllowCellResizeがXRTTBL_RESIZE_ALLまたはXRTTBL_RESIZE_HORIZONTALの場合に、セル/ラベルの右端に表示されるマウスのカーソルを指定します。デフォルトのカーソルは、XC_right_sideシンボルから作成されます。

XmNxrtTblCursorNontraversable Cursor XC_plus CSG

トラバース不可なセル(XmNxrtTblTraversableContextがFalse)の内側で表示されるマウスのカーソルを指定します。デフォルトのカーソルは、XC_plusシンボルから作成されます。

XmNxrtTblCursorTraversable Cursor XC_xterm CSG

トラバース可能なセル(XmNxrtTblTraversableContextがTrue)の内側で表示されるマウスのカーソルを指定します。デフォルトのカーソルは、XC_xtermシンボルから作成されます。

XmNxrtTblCursorVertical Cursor XC_bottom_side CSG

XmNxrtTblAllowCellResizeがXRTTBL_RESIZE_ALLまたはXRTTBL_RESIZE_VERTICALの場合に、セル/ラベルの下に表示されるマウスのカーソルを指定します。デフォルトのカーソルはXC_bottom_sideシンボルから作成されます。

XmNxrtTblDataTypeContext XrtTblDataType XRTTBL_TYPE_STRING CSG

現在のcontextの、セル/ラベル値のデータ型を指定します。セットされると、その時点からセル/ラベル値の変更に影響します。編集した値をセル/ラベル値の構造体に確定する前に[1]、XRT/tableはValidateCellコールバックを用いて、このリソースで指定されているデータ型のデータ検証ルーチンを呼び出します。このリソースは、XmNxrtTblDataTypeListリソースで指定されているデータ型にセットできます。

[1] XmNxrtTblValidatePolicyは、変更されたセル/ラベル値をいつ調べるか決定します。

XmTextがセル値の編集に使用されている場合、有効なデータ型は、XRTTBL_TYPE_STRING、XRTTBL_TYPE_DOUBLE 、 XRTTBL_TYPE_INT 、 XRTTBL_TYPE_FLOAT 、XRTTBL_TYPE_BOOLEANです。

XRT/fieldウィジェットがセル値の編集に使用されている場合の有効なデータ型は次の通りです。

XRTTBL_TYPE_FIELD_CURRENCY	XrtCurrencyField
XRTTBL_TYPE_FIELD_DATE	XrtDateField
XRTTBL_TYPE_FIELD_FLOAT	XrtFloatField
XRTTBL_TYPE_FIELD_INT	XrtIntField
XRTTBL_TYPE_FIELD_STRING	XrtStringField
XRTTBL_TYPE_FIELD_COMBO_CURRENCY	XrtComboBox
XRTTBL_TYPE_FIELD_COMBO_DATE	XrtComboBox
XRTTBL_TYPE_FIELD_COMBO_FLOAT	XrtComboBox
XRTTBL_TYPE_FIELD_COMBO_INT	XrtComboBox
XRTTBL_TYPE_FIELD_COMBO_STRING	XrtComboBox
XRTTBL_TYPE_FIELD_SPIN_CURRENCY	XrtSpinBox
XRTTBL_TYPE_FIELD_SPIN_DATE	XrtSpinBox
XRTTBL_TYPE_FIELD_SPIN_FLOAT	XrtSpinBox
XRTTBL_TYPE_FIELD_SPIN_INT	XrtSpinBox
XRTTBL_TYPE_FIELD_SPIN_STRING	XrtSpinBox

*XrtTblEnableXrtField()*が呼び出されていなければ、XmTextウィジェットが使用されます。

XmNxrtTblDataTypeList XrtTblDataTypeRoutine* **下参照** CSG

データ型と検証ルーチンのNULL終端リストを指定します。指定されるデータ型は、セル/ラベルにセットできる有効な型(XmNxrtTblDataTypeCotnextで指定)です。指定されるルーチンは、データ型を検証するようXRT/tableによって呼び出されます。

このリソースは、次のように定義されるXrtTblDataTypeRoutineを取ります。

```
typedef struct {
  XrtTblDataType   type;
  XtCallbackProc   routine;
} XrtTblDataTypeRoutine;
```

デフォルトのデータ型リストは次の通りです。

XRTTBL_TYPE_STRING	NULL (検査ルーチンなし)

XRTTBL_TYPE_DOUBLE	*XrtTblCheckIsDouble()*
XRTTBL_TYPE_INT	*XrtTblCheckIsInt()*
XRTTBL_TYPE_FLOAT	*XrtTblCheckIsFloat()*
XRTTBL_TYPE_BOOLEAN	*XrtTblCheckIsBoolean()*
XRTTBL_TYPE_FIELD_CURRENCY	XrtCurrencyField検証を使用
XRTTBL_TYPE_FIELD_DATE	XrtDateField検証を使用
XRTTBL_TYPE_FIELD_FLOAT	XrtFloatField検証を使用
XRTTBL_TYPE_FIELD_INT	XrtIntField検証を使用
XRTTBL_TYPE_FIELD_STRING	XrtStringField検証を使用
XRTTBL_TYPE_FIELD_COMBO_CURRENCY	XrtCurrencyField検証を使用
XRTTBL_TYPE_FIELD_COMBO_DATE	XrtDateField検証を使用
XRTTBL_TYPE_FIELD_COMBO_FLOAT	XrtFloatField検証を使用
XRTTBL_TYPE_FIELD_COMBO_INT	XrtIntField検証を使用
XRTTBL_TYPE_FIELD_COMBO_STRING	XrtStringField検証を使用
XRTTBL_TYPE_FIELD_SPIN_CURRENCY	XrtCurrencyField検証を使用
XRTTBL_TYPE_FIELD_SPIN_DATE	XrtDateField検証を使用
XRTTBL_TYPE_FIELD_SPIN_FLOAT	XrtFloatField検証を使用
XRTTBL_TYPE_FIELD_SPIN_INT	XrtIntField検証を使用
XRTTBL_TYPE_FIELD_SPIN_STRING	XrtStringField検証を使用
XRTTBL_TYPE_LAST	この列挙法から独自の検証ルーチンをインデックスする。XmNxrtTblDataTypeListを使用して、データ型と検証ルーチンを追加する。

このリソースをセットする時、新しいデータ型がリストに追加されます。(既に存在する場合は置き換えられます。)新しいデータ型は、XRTTBL_TYPE_LASTより大きい列挙値を持っていなければなりません。指定されたデータ型のルーチンがNULLであれば、その型においては検証は行われません。

XmNxrtTblDataTypeSeries　XrtTblSeries*　NULL　CSG

XrtTblIntValue構造体のseriesとして、各セルのデータ型を指定します。

XmNxrtTblDatePickListContext　struct tm **　NULL　CSG

現在のcontextに対して有効な値のリストを指定します。この値のリストは必ずNULLで終了させてください。テーブルは渡された値の内部コピーを作成します。

XmNxrtTblDatePickListSeries XrtTblSeries * NULL CSG

`XrtTblPickListValue`構造体のseriesとしてpicklistを指定します。

XmNxrtTblDebug Boolean False CSG

Xtエラー・データベースから警告またはエラー・メッセージを*stderr*に出力するかどうかを指定します。`False`であれば、リソース変換のエラーのみが出力されます。

XmNxrtTblDisplayClipArrows Boolean True CSG

セル/ラベル値が全て表示されていないことを示す矢印を表示するかどうかを指定します。clipping arrowは、セル/ラベルの右下角に現れます。

XmNxrtTblDisplayHorizScrollBar XrtTblDispScrollBar XRTTBL_DISPSB_AS_NEEDED CSG

横のスクロールバーをいつ表示するかを指定します。有効な値は次の通りです。

`XRTTBL_DISPSB_ALWAYS`	スクロールバーは常に表示される
`XRTTBL_DISPSB_NEVER`	スクロールバーは表示されない
`XRTTBL_DISPSB_AS_NEEDED`	スクロールバーは、表示できる列の数よりスクロール可能な領域の列の数が多い時に表示される

XmNxrtTblDisplayVertScrollBar XrtTblDispScrollBar XRTTBL_DISPSB_AS_NEEDED CSG

縦のスクロールバーをいつ表示するかを指定します。有効な値は次の通りです。

`XRTTBL_DISPSB_ALWAYS`	スクロールバーは常に表示される
`XRTTBL_DISPSB_NEVER`	スクロールバーは表示されない
`XRTTBL_DISPSB_AS_NEEDED`	スクロールバーは、表示できる行の数よりスクロール可能な領域の行の数が多い時に表示される

XmNxrtTblDisplayWidgetCallback XtCallbackList NULL CSG

セルまたはラベル内にウィジェットを描画する前に呼び出されるコールバックのリストを指定します。DisplayWidgetコールバックは、テーブルが最初に表示される時とそのセルがビューにスクロールされる時に呼び出されます。[1] `XrtTblDisplayWidgetCallbackStruct`のポインタが各コールバックに渡されます。*XtAddCallback()*を用いてリストにコールバックを追加してください。コールバックは次のように構成されます。

```
void
CallbackProcedure(
  Widget     table,
  XtPointer  client_data,
  XtPointer  call_data //points to an XrtTblDisplayWidgetCallbackStruct
```

[1] (`XmNxrtTblArgListContext`を使用して)contextにウィジェットのArgListが既に定義されている場合、DisplayWidgetコールバックを呼び出す前に、ArgListがウィジェットに適用されます。

XrtTblDisplayWidgetCallbackStructは次のように定義されます。

```
typedef struct {
  XrtTblReason    reason;          // Read-only
  XEvent          *event;          // Read-only
  int             row, column; // Read-only
  Widget          clone_source;    // Read-only
  Widget          widget;
} XrtTblDisplayWidgetCallbackStruct;
```

*reason*は常にXRTTBL_REASON_DISPLAY_WIDGETです。*row*、*column*はウィジェットを持つセルまたはラベルの位置を指定。*clone_source*はクローン化されるウィジェット(ウィジェットがクローン化されていなければNULL)。*widget*はセットすることができます。

ウィジェットがクローン化される時は、(XmNxrtTblWidgetLocationまたはXmNxrtTblWidgetContextによって)ウィジェットの位置が指定される前に、コールバックがXmNxrtTblDisplayWidgetCallbackリストに追加されていなければなりません。ウィジェットは、位置が指定される時にクローン化されます。(この点は、以前のXRT/tableのバージョンと異なります。)

XmNxrtTblDoubleBuffer　XrtTblDoubleBuffer　XRTTBL_DOUBLEBUFFER_CELL　CSG

テーブルを表示またはアップデートする際、どのダブル・バッファリング法を使用するか指定します。有効な値は次の通りです。

XRTTBL_DOUBLEBUFFER_CELL	セル/ラベル毎にダブル・バッファリングを行う
XRTTBL_DOUBLEBUFFER_WINDOW	ウィンドウ全体にダブル・バッファリングを行う
XRTTBL_DOUBLEBUFFER_NONE	ダブル・バッファリングは行わない

ダブル・バッファリングがオンの場合は、表示のアップデートは最初にオフ・スクリーンの領域に描かれ、次に画面にコピーされます。

XmNxrtTblEditableContext　Boolean　True　CSG

現在のcontextのセル値をエンドユーザが変更できるかどうかを指定します。編集不可のセルは、トラバースされた時にXmTextウィジェットがマップされますが、ユーザが値を変更しようとすると、ビープ音が鳴ります。編集可能にするには、XmNxrtTblTraversableContextもTrueにセットされていなければなりません。

行/列ラベルのcontextに対してこのリソースをTrueにすると、その位置に行/列をドロップすることができます。

XmNxrtTblEditableSeries　XrtTblSeries*　NULL　CSG

XrtTblBooleanValue構造体のseriesとして、セルを編集できるように指定します。

XmNxrtTblEnterCellCallback　XtCallbackList　NULL　CSG

セルからセルへのトラバースが完了する前と後に呼び出されるコールバックのリストを指定します。EnterCellコールバックは、*TraverseToCell()*のアクションによって呼び出されます。XrtTblEnterCellCallbackStructのポインタが、各コールバックに渡されます。*XtAddCallback()*を用いてリストにコールバックを追加してください。コールバックは次のように構成されてます。

```
void
CallbackProcedure(
  Widget     table,
  XtPointer  client_data,
  XtPointer  call_data//points to an XrtTblEnterCellCallbackStruct
  )
```

XrtTblEnterCellCallbackStructは次のように定義されます。

```
typedef struct {
  XrtTblReason   reason;      // Read-only
  XEvent         *event;      // Read-only
  String         *params;     // Read-only
  Cardinal       num_params;  // Read-only
  int            row, column; // Read-only
  Boolean        doit;
} XrtTblEnterCellCallbackStruct;
```

reason は XRTTBL_REASON_ENTER_CELL_BEGIN または XRTTBL_REASON_ENTER_CELL_ENDのどちらかです。*row*、*column*はトラバースされるセルを指定。*doit*はセルの編集を許可するかどうかを決定するフラグです。

XmNxrtTblExportCallback XtCallbackList NULL CSG

エクスポート(ドラッグ)操作が行われる前と後に呼び出されるコールバックのリストを指定します。各コールバックには、XrtTblExportCallbackStructのポインタが渡されます。*XtAddCallback()*を使用して、リストにコールバックを追加してください。コールバックは次のように構成されます。

```
void
CallbackProcedure(
  Widget     table,
  XtPointer  client_data,
  XtPointer  call_data // points to an XrtTblExportCallbackStruct
  )
```

XrtTblExportCallbackStructは次のように定義されます。

```
typedef struct {
  XrtTblReason   reason;          // Read-only
  XEvent         *event;          // Read-only
  int            row, column; // Read-only
  String         target;          // Read-only
  Boolean        internal;        // Read-only
  XrtTblRange    range;
  unsigned char  operation;
  Boolean        doit;
  Boolean        bell;
} XrtTblExportCallbackStruct;
```

*reason*はXRTTBL_REASON_EXPORT_DRAG_BEGIN(ドラッグの開始 - マウス・ボタンが押される)、XRTTBL_REASON_EXPORT_TARGET(ドロップが開始されようとする)、XRTTBL_REASON_EXPORT_DRAG_END(ドラッグ&ドロップの終了 - マウス・ボタンが上に上がる)のどれかです。*event*はNULL、*row*と*column*はカーソルがあるセルの位置です。*internal*は、ドラッグ&ドロップが同一テーブル上で行われる時はTrueです。

*target*は、エクスポートされるデータ(ドラッグのターゲット)の記述です。*reason*がXRTTBL_REASON_EXPORT_DRAG_BEGINまたはXRTTBL_REASON_EXPORT_DRAG_ENDであれば、*target*は次の内のどれかです。

XRTTBL_TGT_RANGE	ドラッグ・ターゲットは、セル範囲
XRTTBL_TGT_COLUMNS	ドラッグ・ターゲットは、一つか複数の列
XRTTBL_TGT_ROWS	ドラッグ・ターゲットは、一つか複数の行
XRTTBL_TGT_TEXT	ドラッグ・ターゲットは、テーブル以外からの文字列

*reason*がXRTTBL_REASON_EXPORT_TARGETの場合、*target*はXmNxrtTblExportTargetsで指定されるリストにあるターゲットです。(コールバックは、変換手続きが定義されている各ターゲットに対して呼び出されます。変換手続きが定義されていない場合は、デフォルトの手続きが使用され、コールバックは、この手続きが変換を行おうとする時に呼び出されます。)

*range*は、ドラッグされるセル/ラベルの範囲です。*operation*はXmDROP_COPY(選択された範囲をコピー)またはXmDROP_MOVE(選択された範囲を移動、既存のコピーを破棄)です。*doit*は、ドラッグ&ドロップ操作を行うかどうかをコントロールします。*bell*がTrueにセットされていれば、ピープ音が鳴ります。

XmNxrtTblExportDelimiter　String　"\t"　CSG

テーブルから普通のテキストをエクスポート(ドラッグ)する時に、セル間の区切り文字として使用される文字列を指定します。区切り文字が書かれないようにするには、このリソースをXRTTBL_DELIMIT_NONEにします。デフォルトを使用する場合は、XRTTBL_DELIMIT_DEFAULTにセットしてください。

XRTTBL_DELIMIT_BY_WHITESPACEは、スペースとタブ(tab)の組み合わせが区切り文字として処理されるようにします。

XmNxrtTblExportTargets　XrtTblImportExportStruct**　**下記参照**　CSG

テーブルからエクスポートできるアイテム(ターゲット)のリストを指定します。このリストの各エレメントは、XrtTblImportExportStruct型です。

```
typedef struct { String   target;  XrtTblImportExportConvertProc
    converter;} XrtTblImportExportStruct;
```

このリストの最後のエレメントはNULLです。

*target*は、エクスポートされるデータに関する情報を持ちます。ドラッグ操作が行われる時、XrtTblImportExportStructのエレメントは、次の*target*値で自動的に作成されます。

XRTTBL_TGT_RANGE	セル範囲
XRTTBL_TGT_COLUMNS	列全体
XRTTBL_TGT_ROWS	行全体

XmText/XmTextFieldウィジェットへエクスポートする場合は、次のターゲットをインクルードしてください。

```
XRTTBL_TGT_COMPOUND_TEXT
XRTTBL_TGT_TEXT
XRTTBL_TGT_STRING
```

テキストがXmText/XmTextFieldウィジェットにエクスポートされる時、XmNxrtTblExportDelimiterで指定されたストリングがセルのテキスト間に挿入されます。

次のターゲットを指定することもできます。

XRTTBL_TGT_CELLS	XRT/tableのデータ転送フォーマットでエクスポート
XRTTBL_TGT_ROWLABELS	ドラッグの際、行ラベルも含む
XRTTBL_TGT_COLUMNLABELS	ドラッグの際、列ラベルも含む

*converter*は、特定なデータ型をドラッグする時に使用される変換手続きを定義します。NULLの場合は、デフォルトの変換手続きが使用されます。変換手続きは、XRTTBL_TGT_RANGE、XRTTBL_TGT_COLUMNS、XRTTBL_TGT_ROWS、XRTTBL_TGT_ROWLABELS、XRTTBL_TGT_COLUMNLABELSには定義しないでください。

XRTTBL_TGT_ROWLABELSとXRTTBL_TGT_COLUMNLABELSは、XRTTBL_TGT_CELLSが指定されていないと、意味を持ちません。

XmNxrtTblExportTargetsのデフォルトは、次のターゲットのリストです。

```
XRTTBL_TGT_COMPOUND_TEXT
XRTTBL_TGT_TEXT
XRTTBL_TGT_STRING
XRTTBL_TGT_CELLS
```

XmNxrtTblExposeCallback　XtCallbackList　NULL　CSG

テーブルの部分を再描画する前と後に、アプリケーションに通知するために呼び出されるコールバックのリストを指定します。Exposeコールバックは、ウィジェットが再描画、またはスクリーン上のイメージの一部(行ラベル、列ラベル、フリーズされた行等)を変更した時に呼び出されます。XrtTblExposeCallbackStructのポインタが各コールバックに渡されます。*XtAddCallback()*を用いてリストにコールバックを追加してください。コールバックは次のように構成されます。

```
void
CallbackProcedure(
  Widget     table,
  XtPointer  client_data,
  XtPointer  call_data// points to an XrtTblExposeCallbackStruct
)
```

XrtTblExposeCallbackStructは次のように定義されます。

```
typedef struct {
XrtTblReason reason; // Read-only
XEvent           *event; // Read-only
XrtTblRange      range;   // Read-only
} XrtTblExposeCallbackStruct;
```

*reason*はXRTTBL_REASON_EXPOSE_BEGINまたはXRTTBL_REASON_EXPOSE_ENDのどちらかです。*event*は、イベントが内部で生成された時にはNULL。*range*は再描画によって影響された範囲を指定します。

XmNxrtTblFieldArgListContext ArgList NULL CSG

現在のcontextの、Argsのリストを指定します。このリストの最後のエレメントはNULLです。argsは、セルがXRT/fieldを含む時(XRT/fieldのXmNxrtTblDataTypeContextがセットされている場合)にのみ適用されます。このリソースは内部的にコピーされません。

XmNxrtTblFieldArgListSeries XrtTblSeries* NULL CSG

XrtTblArgListValue構造体のseriesとして、セルにXRT/fieldのリソースを指定します。このリソースは内部的にコピーされません。

XmNxrtTblFloatPickListContext double * NULL CSG

現在のcontextに対する有効な値のリストを指定します。最初のエレメントの値は、リストの残りのエレメントの数にしてください。テーブルは、渡された値の内部コピーを作成します。

XmNxrtTblFloatPickListSeries XrtTblSeries * NULL CSG

XrtTblPickListValue構造体のseriesとしてpicklistを指定します。

XmNxrtTblFontListContext XmFontList "fixed" CSG

現在のcontextの、セル/ラベル値の表示に使用されるフォント・リストを指定します。

XmNxrtTblFontListSeries XrtTblSeries* NULL CSG

XrtTblFontListValue構造体のseriesとして、セル/ラベルのフォント・リストを指定します。

XmNxrtTblForegroundContext String "Inherited" CSG

現在のcontextの、セルとラベルの前景色を指定します。値は、Xカラー・データベースの色か、RGBの16進数です。"Inherited"にセットされると、色はXmManagerのXmNForegroundから継承されます。

XRTTBL_REPEAT_ROWまたはXRTTBL_REPEAT_COLUMNにセットされると、色は、XmNxrtTblRepeatForegroundColorsで指定されるリストから、順番に繰り返して適用されます。

モノクロのサーバで、このリソースを白以外の色に指定すると黒に表示されます。

XmNxrtTblForegroundContextがセットされると、XmNxrtTblForegroundPixelContextは自動的に変化します。

XmNxrtTblForegroundPixelContext Pixel XRTTBL_NOVALUE CSG

現在のcontextのセル前景色をPixelで指定します。XRTTBL_NOVALUEであれば、このリソースは無視されます。このリソースがウィジェットの作成時にセットされると、上部と下部の影の色が計算されます。作成後にセットされた場合は影の色は変更されません。[1]

XRTTBL_REPEAT_ROW_PIXELまたはXRTTBL_REPEAT_COLUMN_PIXELにセットされると、色は、XmNxrtTblRepeatForegroundColorsで指定されるリストから、順番に繰り返して適用されます。

XmNxrtTblForegroundPixelContext と XmNxrtTblForegroundContext が、同じ*XtVaSetValues()*コールでセットされた場合は、後者は無視されます。

XmNxrtTblForegroundPixelContext を変更すると、自動的にXmNxrtTblForegroundContextが変更されます。(もし、新しく指定された色が色名に対応しない場合は、XmNxrtTblForegroundContextは、16進数のRGB値になります。)

XmNxrtTblForegroundSeries XrtTblSeries* NULL CSG

XrtTblColorValue構造体のseriesとして、セルとラベルの前景色を指定します。

XmNxrtTblFrameBorderType XrtTblBorderType XRTTBL_BORDER_NONE CSG

セルとラベルの領域を囲むフレームの種類を指定します。下記が有効な種類です。

XRTTBL_BORDER_NONE	罫線なし
XRTTBL_BORDER_ETCHED_IN	セルが沈んでいるように見える二重線
XRTTBL_BORDER_ETCHED_OUT	セルが浮き上がっているように見える二重線
XRTTBL_BORDER_IN	セルが沈んでいるように見える線
XRTTBL_BORDER_OUT	セルが浮き上がっているように見える線
XRTTBL_BORDER_PLAIN	前景色で描かれる装飾無しの罫線

XmNxrtTblFrameShadowThickness Dimension 0 CSG

セルとラベルの領域を囲むフレームの幅を指定します。

XmNxrtTblFrozenColumnPlacement XrtTblColumnPlacement XRTTBL_PLACE_LEFT CSG

ウィジェットが表示されている範囲で、フリーズされる全ての列の位置を指定します。フリーズされる列の位置を変更しても、XrtTblCellValues構造体での列の位置は変更されません。有効な値はXRTTBL_PLACE_LEFTまたはXRTTBL_PLACE_RIGHTです。

[1] 罫線の影の色は、XmNxrtTblBottomShadowContextまたはXmNxrtTblTopShadowContextで指定できます。

XmNxrtTblFrozenColumns int 0 CSG

スクロールできない列の数をテーブルの開始点から指定します。フリーズされる列の位置は、XmNxrtTblFrozenColumnPlacementによって指定されます。

XmNxrtTblFrozenRowPlacement XrtTblRowPlacement XRTTBL_PLACE_TOP CSG

ウィジェットが表示されている範囲で、フリーズされる全ての行の位置を指定します。フリーズされる行の位置を変更しても、XrtTblCellValues構造体での行の位置は変更されません。有効な値は、XRTTBL_PLACE_TOPまたはXRTTBL_PLACE_BOTTOMです。

XmNxrtTblFrozenRows int 0 CSG

スクロールできない行の数をテーブルの開始点から指定します。フリーズされる行の位置は、XmNxrtTblFrozenRowPlacementによって指定されます。

XmNxrtTblHorizontalScrollBarWidget NULL SG

横のスクロールバーのウィジェットIDを指定します。スクロールバーを非表示にする時は、XmNxrtTblDisplayHorizScrollBarを使用し、このリソースをNULLにセットしないでください。

XmNxrtTblHorizScrollBarLeftAttachment XrtTblAttachment XRTTBL_ATTACH_CELLS CSG

横のスクロールバーの左端をどのようにテーブルに接続するか指定します。XRTTBL_ATTACH_CELLSの場合、スクロールバーの左端は表示されているセルの左端の位置に来ます。XRTTBL_ATTACH_SIDEの場合は、ウィジェットの左端の位置に来ます。

XmNxrtTblHorizScrollBarPlacement XrtTblRowPlacement XRTTBL_PLACE_BOTTOM CSG

横のスクロールバーを配置する場所を指定します。XRTTBL_PLACE_BOTTOMの場合は、スクロールバーはテーブルの下、XRTTBL_PLACE_TOPの場合はテーブルの上に表示されます。

XmNxrtTblHorizScrollBarPosition XrtTblSBPosition XRTTBL_SBPOSITION_CELLS CSG

横のスクロールバーをどのようにテーブルに接続するか指定します。XRTTBL_SBPOSITION_CELLSの場合はセル/ラベルのviewportに、XRTTBL_SBPOSITION_SIDEの場合はウィジェットの端に接続されます。XmNxrtTblSpaceは、スクロールバーとテーブルの間の距離を指定します。

XmNxrtTblHorizScrollBarRightAttachment XrtTblAttachment XRTTBL_ATTACH_CELLS CSG

横のスクロールバーの右端をどのようにテーブルに接続するか指定します。XRTTBL_ATTACH_CELLSの場合、スクロールバーの右端は表示されているセルの右端の位置に来ます。XRTTBL_ATTACH_SIDEの場合は、ウィジェットの右端の位置に来ます。

XmNxrtTblImportCallback XtCallbackList NULL CSG

インポート(ドロップ)操作が行われる前と後に呼び出されるコールバックのリストを指定します。各コールバックには、XrtTblImportCallbackStructのポインタが渡されます。*XtAddCallback()*を使用して、リストにコールバックを追加してください。コールバックは次のように構成されます。

```
void
CallbackProcedure(
  Widget     table,
  XtPointer  client_data,
  XtPointer  call_data // points to an XrtTblImportCallbackStruct
)
```

XrtTblImportCallbackStructは次のように定義されます。

```
typedef struct {
  int            reason;      // Read-only
  XEvent         *event;      // Read-only
  int            row, column; // Read-only
  String         target;      // Read-only
  Boolean        internal;    // Read-only
  unsigned char  operation;
  unsigned char  dropSiteStatus;
  Boolean        insert;
  Boolean        doit;
  Boolean        bell;
} XrtTblImportCallbackStruct;
```

*reason*はXRTTBL_REASON_IMPORT_DRAG_MOTION(ドラッグが起こっていることを知らせる)、XRTTBL_REASON_IMPORT_BEGIN(マウス・ボタンが離される)、XRTTBL_REASON_IMPORT_TARGET(ドロップの際の変換が開始されようとする)、のどれかです。*event*はNULL、*row*と*column*はカーソルがあるセルの位置です。*internal*は、ドラッグ&ドロップが同一テーブル上で行われる時はTrueです。

*target*は、インポートされるデータ(ドロップのターゲット)の記述です。*reason*がXRTTBL_REASON_IMPORT_BEGINまたはXRTTBL_REASON_IMPORT_DROP_ENDであれば、*target*は次の内のどれかです。

XRTTBL_TGT_RANGE	ドロップ・ターゲットは、セル範囲
XRTTBL_TGT_COLUMNS	ドロップ・ターゲットは、一つか複数の列
XRTTBL_TGT_ROWS	ドロップ・ターゲットは、一つか複数の行
XRTTBL_TGT_TEXT	ドラッグ・ターゲットは、テーブル以外からの文字列

*target*は、*reason*がXRTTBL_REASON_IMPORT_BEGINの時に、XRTTBL_TGT_COLUMNSまたはXRTTBL_TGT_ROWSをXRTTBL_TARGET_RANGEに修正することができます。

*reason*がXRTTBL_REASON_IMPORT_TARGETの場合、*target*はXmNxrtTblImportTargetsで指定されるリストにあるターゲットです。(コールバックは、変換手続きが定義されている各ターゲットに対して呼び出されます。変換手続きが定義されていない場合は、デフォルトの手続きが使用され、コールバックは、この手続きが変換を行おうとする時に呼び出されます。)

*operation*はXmDROP_COPY(選択された範囲をコピー)またはXmDROP_MOVE(選択された範囲を移動、既存のコピーを破棄)です。*dropSiteStatus*は、XmVALID_DROP_SITE(この場所でドロップは有効、デフォルト)またはXmINVALID_DROP_SITE(ドロップは無効)のどちらかです。*insert*は、指定されている場所にドロップされたデータを挿入するかどうかを指定します。*insert*は、True(可能であれば挿入する、デフォルト)または、False(既存データに上書きする)で、変更することもできます。*doit*は、ドラッグ&ドロップ操作を行うかどうかをコントロールします。*bell*がTrueにセットされていれば、ピープ音が鳴ります。

XmNxrtTblImportDelimiter	String	“\t”	CSG

テーブルへ普通のテキストをインポート(ドロップ)する時に、セル間の区切り文字として使用される文字列を指定します。区切り文字が指定されないようにするには、このリソースを XRTTBL_DELIMIT_NONE にします。デフォルトを使用する場合は、XRTTBL_DELIMIT_DEFAULTにセットしてください。

XRTTBL_DELIMIT_BY_WHITESPACEは、スペースとタブ(tab)の組み合わせが区切り文字として処理されるようにします。

XmNxrtTblImportTargets　XrtTblImportExportStruct** **下記参照**　CSG

テーブルへインポートできるアイテム(ターゲット)のリストを指定します。このリストの各エレメントは、XrtTblImportExportStruct型です。

```
typedef struct { String  target; XrtTblImportExportConvertProc
    converter;} XrtTblImportExportStruct;
```

このリストの最後のエレメントはNULLです。

ターゲット・リストの各アイテムは、インポートできるデータ型を指定し、*target*は、このデータ型の指定に使用されます。他のテーブル(または同一テーブル)からインポートする時は、ターゲット・アイテムは、次の*target*値のどれでも、または全てを持つことができます。

XRTTBL_TGT_RANGE	セル範囲
XRTTBL_TGT_COLUMNS	列全体
XRTTBL_TGT_ROWS	行全体

次のターゲット・アイテムは、ラベルと(または)セルがドロップされるかどうか指定します。

XRTTBL_TGT_CELLS	テキストは、XRT/tableのデータ転送フォーマット
XRTTBL_TGT_ROWLABELS	ドロップの際、行ラベルも含む
XRTTBL_TGT_COLUMNLABELS	ドロップの際、列ラベルも含む

XRTTBL_TGT_CELLSが指定されていないと、XRTTBL_TGT_ROWLABELSとXRTTBL_TGT_COLUMNLABELSは指定できません。

テーブル内で行または列のドラッグ&ドロップ操作を安全かつ効率的に行うには、リストに次のターゲット・アイテムを加えてください。

XRTTBL_TGT_INTERNALMOVE	(同一テーブル内で)高速のドラッグ&ドロップを指定

このターゲットが追加された場合、オーバーラップするドラッグ&ドロップの操作が正しく行われます。内部的なドラッグ&ドロップでは、フォント・リストやpixmapも含む行または列の全てが移動します。

XmTextウィジェットからインポートする時は、次のターゲットのどれかまたは全てを定義してください。

```
XRTTBL_TGT_COMPOUND_TEXT
XRTTBL_TGT_TEXT
XRTTBL_TGT_STRING
```

実際に必要となるデータ型は、XmTextウィジェットからどのようにデータがドラッグされたかに依存します。上の三つのターゲット型を全て定義しておいた方が良い結果が得られます。XmText/XmTextFieldウィジェットからテキストがインポートされる時は、XmNxrtTblImportDelimiterで指定されるストリングがセルの区切りを示します。

*converter*は、特定なデータ型をドロップする時に使用される変換手続きを定義します。NULLの場合は、デフォルトの変換手続きが使用されます。変換手続きは、XRTTBL_TGT_RANGE、XRTTBL_TGT_COLUMNS、XRTTBL_TGT_ROWS、XRTTBL_TGT_ROWLABELS、XRTTBL_TGT_COLUMNLABELSには定義しないでください。

XmNxrtTblImportTargetsのデフォルトは、次のターゲットのリストです。

```
XRTTBL_TGT_ROWS
XRTTBL_TGT_COLUMNS
XRTTBL_TGT_RANGE
XRTTBL_TGT_INTERNALMOVE
XRTTBL_TGT_CELLS
XRTTBL_TGT_COMPOUND_TEXT
XRTTBL_TGT_TEXT
XRTTBL_TGT_STRING
```

XmNxrtTblIntPickListContext **long *** **NULL** **CSG**

現在のcontextに対する有効な値のリストを指定します。最初のエレメントの値は、リストの残りのエレメントの数にしてください。テーブルは、渡された値の内部コピーを作成します。

XmNxrtTblIntPickListSeries **XrtTblSeries *** **NULL** **CSG**

XrtTblPickListValue構造体のseriesとしてpicklistを指定します。

XmNxrtTblJumpScroll XrtTblJumpScroll XRTTBL_JUMP_VERTICAL CSG

各スクロールバーのスクロールの動作を指定します。XRTTBL_JUMP_NONEの場合は、スムーズにスクロールします。XRTTBL_JUMP_HORIZONTALの場合は列毎に、XRTTBL_JUMP_VERTICALの場合は行毎にスクロールします。XRTTBL_JUMP_ALLにすると、両方のスクロールバーがジャンプ・スクロールします。

XmNxrtTblLabelContext XmString NULL CSG

現在のcontextのラベル値を指定します。データ型はXmStringでも、Stringを指定することができます(Motif 1.2にリンクしている場合)。XRT/tableは、内部でXmStringに変換します。XmNxrtTblXmStringSetModeの設定によってはStringを指定することもできます。

このリソースを設定する時、セル値はcontextの各ラベルにコピーされます。このリソースを取り出す場合は、カレントcontextは一つの行または列ラベルでなければなりません。

XmNxrtTblLabelCreateCallback　XtCallbackList　NULL　CSG

テーブルが空白の(XrtTblXmStringList構造体に値がない)ラベルを表示するときに呼び出されるコールバックのリストを指定します。コールバックは、ウィジェットの初期表示、スクロールとエクスポーズの際に呼び出されます。各コールバックには、XrtTblLabelCreateCallbackStructのポインタが渡されます。

```
typedef struct {
  XrtTblReason     reason;          // Read-only
  XEvent           *event;          // Read-only
  int          row, column; // Read-only
  XmString         string;
  Boolean      store;           // initially False
  Boolean      free_string; // initially False
XrtTbllabelCreateCallbackStruct;
```

*reason*はXRTTBL_REASON_LABELCREATEにセットされます。*row*と*column*はセルの位置を指定します。*string*は、コールバック・ルーチンによって返されたセル値です。ストリングがテーブルに記憶されるようにするには、*store*をTrueにセットします。(このラベルに対しては、コールバックはもう呼び出されません。) *free_string*がTrueであれば、*string*は表示後に解放されます。

デフォルトでは、*string*はXmStringとString型の両方の値を認識できます。(但し、Motif 2.xにリンクしていない場合) XmNxrtTblXmStringSetModeを使用して、StringまたはXmStringどちらかの型に制限できます。

このリソースがセットされている場合、XmNxrtTblLabelCreateFuncは無視されます。

XmNxrtTblLabelCreateFunc　XrtTblXmStringProc　NULL　CSG

XrtTblXmStringList構造体に値がない空白のラベルを表示する時に呼ばれるルーチンを指定します。ルーチンは、ウィジェットの初期表示、スクロール、エクスポーズの時に呼び出されます。このルーチンは、次のプロトタイプを持っています。

```
XmString label_value_func(Widget table, int row, int column)
```

*table*はXRT/tableウィジェット。*row*または*column*は、XRTTBL_LABELにセットされ、行または列ラベルを指定します。戻り値は、StringまたはXmStringです。(XmNxrtTblXmStringSetModeの値に依存します。)返されたストリングは表示後、ウィジェットによって解放されます。*XrtTblCreateSpreadsheetLabel()*を使用してスプレッドシート形式のラベルを作成することができます。このルーチンの登録には、*XtSetValues()*を使用し、*XtAddCallback()*は使用しないでください。XmNxrtTblLabelCreateCallbackがセットされている時は、このルーチンは無視されます。

XmNxrtTblLeftColumn　int　dynamic　CSG

ウィンドウの左の部分に(部分的にでも)表示される、フリーズされていない列を指定します。このリソースは、ユーザのスクロールによってアップデートします。セットされると、その列をウィンドウの左の部分に表示するようテーブルはスクロールします。このリソースの値は、フリーズされている列の数かそれ以上です。

XmNxrtTblMarginHeight　Dimension　2　CSG

セルの罫線の内側とセルの内容の上/下部の間のマージンを(pixelで)指定します。このリソースは、テーブル全体に影響します。

このリソースの値は、XmTextのリソースXmNhighlightThicknessの値より小さくできません。トラバース可能なセルがない時は、このリソースは、0にセットできます。

XmNxrtTblMarginWidth　Dimension　5　CSG

セルの罫線の内側とセルの内容の右/左部の間のマージンを(pixelで)指定します。このリソースは、テーブル全体に影響します。このリソースの値は、XmTextのリソースXmNhighlightThicknesの値より小さくできません。トラバース可能なセルがない時は、このリソースは0にセットできます。

XmNxrtTblMaxLengthContext　int　XRTTBL_MAXINT　CSG

現在のcontextのセルにユーザが入力できる文字の最大数を指定します。これは、XmTextウィジェットのXmNmaxLengthリソースに対応します。このリソースは、*XrtTblSetCellValues()*、*XrtTblGetCellValues()*またはマクロによって変更されたセル値には影響しません。複数行のセルでのnewline文字は、一文字として数えられます。

XmNxrtTblMaxLengthSeries　XrtTblSeries*　NULL　CSG

XrtTblIntValue構造体のseriesとして、ユーザがセルに入力できる最大文字数を指定します。

XmNxrtTblMinCellVisibility　int　100　CSG

トラバースされた時に表示されるセルの最小分量を指定します。テーブルがスクロールして非表示のセルにトラバースされた場合、このリソースが、ビューにスクロールされるセルの分量(割合)を決定します。100の場合はセルは完全に表示されます。10の場合は、セルの10%が表示され、0の場合はセルは表示されません。このリソースは、*XrtTblMakeCellVisible()*の動作に影響を与えます。

XmNxrtTblMinFlexibleHeight　Dimension　5　CSG

XmNxrtTblPixelHeightContextがXRTTBL_FLEXIBLEにセットされている行の最小の高さです。

XmNxrtTblMinFlexibleWidth　Dimension　5　CSG

XmNxrtTblPixelWidthContextがXRTTBL_FLEXIBLEにセットされている列の最小幅です。

XmNxrtTblMode　XrtTblMode　XRTTBL_MODE_TABLE　CSG

表示モードを指定します。値は、次のうちの一つです。

XRTTBL_MODE_LIST	XmListの操作を行う
XRTTBL_MODE_TABLE	テーブルは通常の操作を行う

このリソースがXRTTBL_MODE_LISTにセットされている場合、セルを選択すると、その行の全てのセルも選択されます。XmNxrtTblSelectionPolicyは、行、複数行、複数範囲を選択できるかどうかをコントロールします。

XmNxrtTblMultilineContext Boolean False CSG

現在のcontextのセルにユーザが複数行のテキストを入力できるかどうか(XmTextウィジェットのXmNeditModeがXmMULTI_LINE_EDITにセットされるかどうか)を指定します。このリソースは、アプリケーションによって追加、変更された値には影響しません。セル値が既に複数行であれば、XmNxrtTblMultilineContextの値に関らず、セルがトラバースされた時に複数行のXmTextウィジェットが表示されます。

XmNxrtTblMultilineSeries XrtTblSeries* NULL CSG

XrtTblBooleanValue構造体のseriesとして、セルが複数行になるよう指定します。

XmNxrtTblNumColumns int 5 CSG

テーブルの列の数を指定します。このリソースは、テーブルのXrtTblCellValues構造体には影響を与えません。このリソースの値は、列の数が*XrtTblAddColumns()*または*XrtTblDeleteColumns()*によって変更された時に、アップデートされます。0にセットされている場合はセルは表示されませんが、定義されている行ラベルは表示されます。

XmNxrtTblNumRows int 10 CSG

テーブルの行の数を指定します。このリソースは、テーブルのXrtTblCellValues構造体には影響を与えません。このリソースの値は、行の数が*XrtTblAddRows()*または*XrtTblDeleteRows()*によって変更された時に、アップデートされます。0にセットされている場合はセルは表示されませんが、定義されている列ラベルは表示されます。

XmNxrtTblOverflowContext XrtTblOverflow XRTTBL_OVERFLOW_NONE CSG

現在のcontextの、セルまたはラベル値をセルの境界を越えて表示(overflow)させるかどうか指定します。XRTTBL_OVERFLOW_NONEの場合は、テキストはセルの境界で切り取られます。XRTTBL_OVERFLOW_HORIZONTALであれば、テキストは横隣の空白のセルに渡って表示されます。

テキストがoverflowする方向は、(XmNxrtTblAlignmentContextで指定される)セルのテキストの行揃えに依存します。コールバックや方法等(XmNxrtTblCellValueCallback、XmNxrtTblCellValueFunc、XmNxrtTblLabelCreateCallback、XmNxrtTblLabelCreateFunc)によってセットされているセル/ラベル値は、overflowしません。また、行ラベルや結合されているセルもoverflowできません。

既にXmNxrtTblWordWrapContextまたはXmNxrtTblWordWrapSeriesをセットしている場合は、このリソースはセットしないでください。

XmNxrtTblOverflowSeries XrtTblSeries* NULL CSG

XrtTblIntValue構造体のseriesとして、セル値がセルの境界を越えて表示(overflow)されるかどうか指定します。

既にXmNxrtTblWordWrapContextまたはXmNxrtTblWordWrapSeriesをセットしている場合は、このリソースはセットしないでください。

XmNxrtTblPixelHeightContext int XRTTBL_NOVALUE CSG

現在のcontextの行の高さをpixelで指定します。縦方向にセルに入り切らないセル値、pixmap、ウィジェットは切り取られます。XRTTBL_NOVALUE(デフォルト)にセットされていなければ、このリソースが行の高さをコントロールします。

XRTTBL_NOVALUEの場合はXmNxrtTblCharHeightContextが行の高さをコントロールします。このリソースを設定する時は、contextは行全体(x, XRTTBL_ALL)でなければなりません。次の値が有効です。

XRTTBL_VARIABLE	その行の一番高いセルにセットする
XRTTBL_AS_IS	変化なし
XRTTBL_FLEXIBLE	テーブルの高さにより行の高さも変化する

XRTTBL_FLEXIBLEの場合、フレキシブルな行とその他の行の高さの合計がテーブル全体の高さの合計と合うように計算されます。また、全てのフレキシブルな行は、同じ高さにセットされます。スペースがない場合は、全てのフレキシブルな行の高さは、XmNxrtTblMinFlexibleHeightの値にセットされます。

このリソースは、ユーザがインタラクティブに行をリサイズした時にアップデートされます。行の高さが文字指定になっている場合は、ユーザ・インタラクションによってpixelの高さに置き換えられます。

XmNxrtTblPixelHeightSeries XrtTblSeries* NULL CSG

XrtTblIntValue構造体のseriesとして、行の高さをpixelで指定します。

XmNxrtTblPixelWidthContext int XRTTBL_NOVALUE CSG

現在のcontextの列幅をpixelで指定します。横方向にセルに入り切らないセル値、pixmap、ウィジェットは切り取られます。XRTTBL_NOVALUE(デフォルト)にセットされていなければ、このリソースが幅をコントロールします。XRTTBL_NOVALUEの場合はXmNxrtTblCharWidthContextが列幅をコントロールします。このリソースを設定する時は、contextは列全体(XRTTBL_ALL, x)でなければなりません。

次の値が有効です。

XRTTBL_VARIABLE	その列の一番大きなセルにセットする
XRTTBL_AS_IS	変化なし
XRTTBL_FLEXIBLE	テーブルの幅により列幅も変化する

XRTTBL_FLEXIBLEの場合、フレキシブルな列とその他の列の幅の合計がテーブル全体の幅の合計と合うように計算されます。また、全てのフレキシブルな列は、同じ幅にセットされます。スペースがない場合は、全てのフレキシブルな列幅は、XmNxrtTblMinFlexibleWidthの値にセットされます。

このリソースは、ユーザがインタラクティブに列をリサイズした時にアップデートされます。列幅が文字指定になっている場合は、ユーザ・インタラクションによってpixelの幅に置き換えられます。

XmNxrtTblPixelWidthSeries XrtTblSeries* NULL CSG

XrtTblIntValue構造体のseriesとして、列幅をpixelで指定します。

XmNxrtTblPixmapContext Pixmap XmUNSPECIFIED_PIXMAP CSG

現在のcontextで表示されるpixmapを指定します。セルに値がある場合は、pixmapはテキストの後ろに背景として表示されます。セルがウィジェットを持っている場合は、XRT/tableはウィジェットのXmNbackgroundPixmapリソースをそのpixmapにセットします。

XmNxrtTblPixmapLayoutContextはセル内のpixmapの位置を決定します。pixmapはコピーされません。

XRT/tableのリソース変換プログラムは、XBMとXPMフォーマットのpixmapをサポートします。

XmNxrtTblPixmapLayoutContext XrtTblPixmapLayout XRTTBL_PIXMAP_MIDDLECENTER CSG

現在のcontextで、(XmNxrtTblPixmapContextで設定される)背景のpixmapのレイアウトを指定します。有効な値はXRTTBL_PIXMAP_TILEと、下の9個の値です。

XRTTBL_PIXMAP_TOPLEFT	XRTTBL_PIXMAP_TOPCENTER	XRTTBL_PIXMAP_TOPRIGHT
XRTTBL_PIXMAP_MIDDLELEFT	XRTTBL_PIXMAP_MIDDLECENTER	XRTTBL_PIXMAP_MIDDLERIGHT
XRTTBL_PIXMAP_BOTTOMLEFT	XRTTBL_PIXMAP_BOTTOMCENTER	XRTTBL_PIXMAP_BOTTOMRIGHT

XRTTBL_PIXMAP_TILEにセットされている場合は、pixmapはタイルのように並べられます。

XmNxrtTblPixmapLayoutSeries XrtTblSeries* NULL CSG

XrtTblIntValue構造体のseriesとして、背景のpixmapのレイアウトを指定します。

XmNxrtTblPixmapSeries XrtTblSeries* NULL CSG

XrtTblIntValue構造体のseriesとして、背景のpixmapを指定します。

XmNxrtTblPixmapShadowContext Boolean True CSG

セルまたはラベルにpixmapがある時、そのcontextに影をつけるかどうか指定します。Falseの場合は、pixmapがセルの端まで表示されます。

XmNxrtTblPixmapShadowSeries XrtTblSeries* NULL CSG

pixmapを持つセル/ラベルに影をつけるかどうかを、XrtTblBooleanValue構造体のseriesとして指定します。

XmNxrtTblPreviewScrolling XrtTblPreviewScroll XRTTBL_PREVIEW_NONE CSG

スクロールバーがドラッグされている時に、スクロールバーのつめの横に小さなウィンドウを表示するかどうかを指定します。このウィンドウには、スクロールが終了した時にテーブルの左上角に表示される行/列を表示します。デフォルト値は、XRTTBL_PREVIEW_NONEで、このウィンドウは表示されません。有効な値は次の通りです。

XRTTBL_PREVIEW_NONE
XRTTBL_PREVIEW_VERTICAL
XRTTBL_PREVIEW_HORIZONTAL
XRTTBL_PREVIEW_BOTH

値がXRTTBL_PREVIEW_NONEでなければ、テーブルはスクロールが完了(スクロールバーのつめがリリース)するまで、再描画を行いません。

このウィンドウには、テーブルの表示をコントロールする属性と同じものが多く使用されます。前景色はXRT/tableのXmNforeground、背景色はXRT/tableのXmNbackground、フォント・リストはXRT/tableのデフォルトのfontlistです。

これは、contextが(ALL, ALL)の場合のXmNxrtTblFontListSeriesの最初のエントリか、そのようなエントリが存在しない場合は'fixed'です。デフォルトの罫線はXRTTBL_BORDER_NONEです。

これらの値は、次の関数を用いて変更することができます。

```
void
XrtTblSetPreviewProperties(
  Widget       w,
  Pixel        foreground,
  Pixel        background,
  XmFontList fontlist, /* This value is copied internally */
  XrtTblBorderTypeborder
  );

void
XrtTblGetPreviewProperties(
  Widget       w,
  Pixel        *foreground,/* Returned */
  Pixel        *background,/* Returned */
  XmFontList   *fontlist, /* Returned. This value is not
                             Copied and should not be freed. */
  XrtTblBorderType*border /* Returned */
);
```

XmNxrtTblPrintCallback　XtCallbackList　NULL　CSG

テーブルの各ページがプリントされる際に呼び出されるコールバックのリストを指定します。各ルーチンには、XrtTblPrintCallbackStructのポインタが渡されます。

```
typedef struct {
  int          reason;          // Read-only
  XEvent           *event;          // Read-only
  XrtTblRange  **range;         // Read-only
  int          current_page;    // Read-only
  int          total_pages; // Read-only
  int          across_pages;    // Read-only
  int          down_pages;      // Read-only
  int          page_number;
  String           header, footer;
  FILE             *fp;
  Boolean      write_file_header;
  Boolean      write_file_trailer;
  Boolean      doit;            // initially True
} XrtTblPrintCallbackStruct;
```

*reason*は、XRTTBL_REASON_PRINT_BEGINかXRTTBL_REASON_PRINT_ENDのどちらかです。*event*は常にNULL、*range*はそのページのセル範囲、*current_page*は現在プリントされているページです。*across_pages*は印刷するテーブルの幅に必要なページ数、*down_pages*はテーブルの高さに必要なページ数です。*total_pages*は、印刷されるページ数(=*across_pages* x *down_pages*)です。*page_number*はページ番号で、(*XrtTblVaDrawPS()*によって)ヘッダーまたはフッターに"#"というシンボルが指定されていれば、ページ番号が挿入されます。*header*と*footer*は、現在のページのヘッダーとフッターです。*fp*は出力されるファイルのポインタです。*write_file_header*は、PostScriptのヘッダーを書くかどうかを示します。(*current_page*が1の時、*write_file_header*はTrueです。) *write_file_trailer*は、PostScriptトレイラー(trailer)を書くかどう

かを示し、*current_page=total_pages*の時は自動的にTrueにセットされます。*doit*は、印刷時にそのページを抜かすかどうかを示しますが、*current_page*または*total_page*では、このフラグは意味を持ちません。*header*または*footer*が変更された場合は、既存のヘッダー/フッターを解放し、新しいStringを割り当ててください。ページは、左から右、上から下の順にプリントされます。

XmNxrtTblPrintWidgetCallback　XtCallbackList　NULL　CSG

セルにあるウィジェットが印刷される際に呼び出されるコールバックのリストを指定します。各ルーチンには、XrtTblPrintWidgetCallbackStructのポインタが渡されます。

```
typedef struct {
  int     reason;          // Read-only
  XEvent      *event;          // Read-only
  Widget      widget;          // Read-only
  int     row, column; // Read-only
  FILE        *fp;             // Read-only
  int     current_page;    // Read-only
  int     page_number; // Read-only
  Boolean doit;            // initially True
} XrtTblPrintWidgetCallbackStruct;
```

*reason*は、XRTTBL_REASON_PRINTWIDGET_BEGINまたはXRTTBL_REASON_PRINTWIDGET_ENDのどちらかです。*event*は常にNULL、*row*と*column*は印刷されるセルを指定します。タイトル・ウィジェットが印刷される時は、*row*と*column*はXRTTBL_NOVALUEになります。*fp*は出力されるファイルのポインタです。*current_page*は既に印刷されているページ数、*page_number*はページ番号で、(*XrtTblVaDrawPS()*によって)ヘッダーまたはフッターに"#"というシンボルが指定されていれば、ページ番号が挿入されます。ウィジェットを印刷しない場合は、*doit*をFalseにセットしてください。

XmNxrtTblPSFontMapList　XrtTblPSFontMap*　NULL　CSG

Xフォントに対応するPostScriptフォント(とオプションでポイント・サイズ)のリストを定義します。リストの各エレメントは、XrtTblPSFontMap型です。

```
typedef struct {
  String       font_name, tag;
  String       PS_font;
  double       point_size; // 0 = infer
} XrtTblPSFontMap;
```

Xフォントは、完全なフォント名(*font_name*)またはXmNfontListタグ(*tag*)で示すことができます。*PS_font*はPostScriptフォントです。*poit_size*はポイント・サイズで、0の時は、サイズはフォントから推測されます。リストの最後のアイテムは、NULLにセットされた*PS_font*メンバです。

XmNxrtTblRangeContext　XrtTblRange*　NULL　CSG

現在のcontextとして範囲を指定します。有効な範囲にセットされると、このリソースがカレントcontextを定義し、XmNxrtTblContextの値を無視します。次に続くContextリソースの設定は、範囲の各セルまたはラベルに適用されます。XmNxrtTblContextがセットされると、このリソースは副作用でNULLにセットされます。[3]

このリソースをセットする場合は*XrtTblSetRange()*を使用してください。XmNxrtTblRangeContextは、リソース・ファイルではセットできません。

[3] contextとContextリソースに関する詳細は、2.4を参照してください。

XmNxrtTblRepaint XrtTblRepaint XRTTBL_REPAINT_ON CSG

リソースがセットされる度にXRT/tableウィジェットを再描画、再計算するかどうかを指定します。有効な値は、次の通りです。

XRTTBL_REPAINT_ON	必要時に描画し、セルとテーブルのサイズは計算される
XRTTBL_REPAINT_OFF	描画はされないが、セルとテーブルのサイズは計算される
XRTTBL_REPAINT_NOCALC	再描画やサイズの計算は行われない

XmNxrtTblRepeatBackgroundColors String* NULL CSG

XmNxrtTblBackgroundSeries が XRTTBL_REPEAT_ROW または XRTTBL_REPEAT_COLUMNの時に使用される色のリストです。このリストの最後のエレメントはNULLです。

XmNxrtTblRepeatForegroundColors String* NULL CSG

XmNxrtTblForegroundSeries が XRTTBL_REPEAT_ROW または XRTTBL_REPEAT_COLUMNの時に使用される色のリストです。このリストの最後のエレメントはNULLです。

XmNxrtTblResizeCallback XtCallbackList NULL CSG

ウィジェットのリサイズの前後に、アプリケーションに通知するために呼び出されるコールバックのリストを指定します。XrtTblResizeCallbackStructのポインタが各コールバックに渡されます。*XtAddCallback()*を用いてリストにコールバックを追加してください。コールバックは次のように構成されます。

```
void
CallbackProcedure(
  Widget        table,
  XtPointer     client_data,
  XtPointer     call_data // points to an XrtTblResizeCallbackStruct
  )
```

XrtTblResizeCallbackStructは次のように定義されます。

```
typedef struct {
  XrtTblReason  reason;          // Read-only
  XEvent        *event;          // Read-only
  Dimension     width, height;   // Read-only
  Dimension     visible_width;   // Read-only
  Dimension     visible_height;  // Read-only
} XrtTblResizeCallbackStruct;
```

*reason*はXRTTBL_REASON_RESIZE_BEGINまたはXRTTBL_REASON_RESIZE_ENDのどちらかです。*width*と*height*は現在、または新しいウィジェットのサイズ。*visible_width*と*visible_height*は表示されているセル/ラベル領域のサイズです。

XmNxrtTblResizeCellCallback XtCallbackList NULL CSG

ユーザが行や列をリサイズする時に呼び出されるコールバックのリストを指定します。ResizeCellコールバックは、*ResizeCell()*によって呼び出されます。

XrtTblResizeCellCallbackStructのポインタが各コールバックに渡されます。*XtAddCallback()*を使用してリストにコールバックを追加してください。コールバックは次のように構成されます。

```
void
CallbackProcedure(
  Widget        table,
  XtPointer     client_data,
  XtPointer     call_data//points to an XrtTblResizeCellCallbackStruct
)
```

XrtTblResizeCellCallbackStructは次のように定義されます。

```
typedef struct {
  XrtTblReason  reason;              // Read-only
  XEvent        *event;              // Read-only
  String        *params;             // Read-only
  Cardinal      num_params;          // Read-only
  int           row, column;     // Read-only
  int           current_row_height,   // Read-only
  int           current_column_width, // Read-only
  int           new_row_height,
  int           new_column_width,
  Boolean       doit;
} XrtTblResizeCellCallbackStruct;
```

*reason*はXRTTBL_REASON_RESIZE_BEGIN、XRTTBL_REASON_RESIZE_END、XRTTBL_REASON_RESIZE_DONEのどれかです。*row*、*column*はリサイズされた行と(または)列を指定(行だけのリサイズの場合、列はXRTTBL_NOVALUE、またはその逆)。*current_row_height*と*current_column_width*は現在の行/列の高さ/幅。*new_row_height*と*new_column_width*は、(ユーザのリサイズによってセットされる)新しい行/列の高さ/幅。*doit*はインタラクティブなリサイズを許可するかどうかを指定するフラグです。

XmNxrtTblRowLabelOffset　int　0　CSG

行ラベルとセル領域の間のスペースをpixelで指定します。負の値が指定された場合は、ラベルはセル領域の上に重なります。

XmNxrtTblRowLabelPlacement　XrtTblColumnPlacement　XRTTBL_PLACE_LEFT　CSG

行ラベルの位置を指定します。有効な値はXRTTBL_PLACE_LEFTとXRTTBL_PLACE_RIGHTです。

XmNxrtTblRowLabels　XrtTblXmStringList*　NULL　CSG

行ラベルを指定します。データ型はXmStringですが、Stringも指定できます(Motif 1.2にリンクしている場合)。XRT/tableは、内部でこれをXmStringに変換します。XmNxrtTblXmStringSetModeの設定によっては、Stringを指定することもできます。

XmNxrtTblScrollCallback　XtCallbackList　NULL　CSG

テーブルがスクロールされる際に呼び出されるコールバックのリストを指定します。各ルーチンには、XrtTblScrollCallbackStructのポインタが渡されます。

```
typedef struct {
  int          reason;      // Read-only
  XEvent           *event;      // Read-only
  unsigned char    direction;   // Read-only
  int              new_value
  String           preview_text     //Read/Write
  XmScrollBarCallbackStruct *sb_cbs; // Read-only
} XrtTblScrollCallbackStruct;
```

*reason*は、XRTTBL_REASON_SCROLL_BEGINまたはXRTTBL_REASON_SCROLL_ENDのどちらかです。*event*は渡されたXEventです。*direction*は、XmHORIZONTALまたはXmVERTICAL。*new_value*は、XmNxrtTblTopRowまたはXmNxrtTblLeftColumnの新しい値、*preview_text*はプレビュー・ウィンドウに表示されるテキストです。値を置き換えるには、*XtFree()*で古い値を解放し、新しい値に対してメモリを割り当ててください。XRT/tableは、このコールバック構造体のエレメントによって使用されたメモリを解放します。*sb_cbs*は、ScrollBarコールバックに渡されたコールバック構造体のポインタです。

XmNxrtTblSelectCallback　XtCallbackList　NULL　CSG

ユーザがセルを選択する時に呼び出されるコールバックのリストを指定します。Selectコールバックは、*Select()*アクションによって開始されます。コールバックは、選択される各セルに呼ばれます。XrtTblSelectCallbackStructのポインタが各コールバックに渡されます。*XtAddCallback()*を用いてリストにコールバックを追加してください。コールバックは次のように構成されます。

```
void
CallbackProcedure(
  Widget     table,
  XtPointer  client_data,
  XtPointer  call_data // points to an XrtTblSelectCallbackStruct
)
```

XrtTblSelectCallbackStructは、次のように定義されます。

```
typedef struct {
  XrtTblReason     reason;      // Read-only
  XEvent           *event;      // Read-only
  String           *params;     // Read-only
  Cardinal         num_params; // Read-only
  int          range_index; // Read-only
  XrtTblRange  *range;      // Read-only
  int          row, column;
  int          click_count;
  Boolean      doit;
} XrtTblSelectCallbackStruct;
```

*reason*は XRTTBL_REASON_SELECT_BEGIN 、 XRTTBL_REASON_SELECT_END 、 XRTTBL_REASON_SELECT_EXTEND 、 XRTTBL_REASON_UNSELECT のどれかです。*range_index*は、(XmNxrtTblSelectedCellListで指定される)選択されたセル・リスト内の*range*の位置。*range*は選択された(または選択解除される)セル範囲。*row*、*column*は選択または選択解除されるセルまたはラベルの位置を指定。*click_count*は、(XmNmultiClickTimeで指定される)マルチ・クリック内のユーザによるマウスのクリック回数を指定。*doit*は選択/選択解除を許可するかどうかを指定するフラグです。

XmNxrtTblSelectedBackground String "Black" CSG

選択されたセルの背景色を指定します。値は、Xカラー・データベースの色またはRGBの16進数で指定できます。"None"にセットされていると、そのセルに定義されている色(XmNxrtTblForegroundContextとXmNxrtTblBackgroundContext)を使用し、選択されていないセルと同じ色になります。

モノクロのサーバで、このリソースを黒以外の色に指定すると白に表示されます。

XmNxrtTblSelectedCellList XrtTblRange** NULL CSG

現在選択されているセル範囲のリストを指定します。(このリストの最後のエレメントはNULLです。)ユーザが選択を行った時、このリソースはダイナミックにアップデートされます。(XmNxrtTblSelectionPolicyで指定される)選択のポリシーが、ユーザまたはアプリケーションにより選択できる分量をコントロールします。各選択された範囲は次の構造体で定義されます。

```
typedef struct {
  int     start_row, start_column;
  int     end_row, end_column;
} XrtTblRange;
```

各範囲の開始と終了点が、対角線上に位置して範囲を定義します。ユーザはどの方向にも選択を開始することができるので、*starat_row*と(または)*start_column*は、*end_row*と(または)*end_column*より大きくなることもあります。ユーザが行/列ラベルをクリックして行/列全体を選択した時は、*end_row*または*end_columns*はXRTTBL_MAXINTにセットされます。選択された範囲リストから、一つの範囲を取り出すには、*XrtTblGetSelectedRange()*を使用してください。

XmNxrtTblSelectedForeground String "White" CSG

選択されたセルの前景色を指定します。値は、Xカラー・データベースの色またはRGBの16進数で指定できます。XmNxrtTblSelectedBackgroundが"None"でない時に、

このリソースが“None”にセットされると、選択されたセルの前景色は、そのセルに定義されている前景色(XmNxrtTblForegroundContext)と同じになります。このリソースは、XmNxrtTblSelectedBackgroundが“None”にセットされると無視されます。

モノクロのサーバで、このリソースを白以外の色に指定すると黒に表示されます。

XmNxrtTblSelectionPolicy　XrtTblSelectionPolicy　XRTTBL_SELECT_NONE　CSG

エンドユーザとアプリケーションによる選択の決まりを指定します。有効な値は次の通りです。

XRTTBL_SELECT_NONE	選択はできない
XRTTBL_SELECT_SINGLE	一つのセル/行が選択できる
XRTTBL_SELECT_RANGE	一つのセル/行範囲が選択できる
XRTTBL_SELECT_MULTIRANGE	複数のセル/行範囲が選択できる

XmNxrtTblModeリソースが、セルまたは行が選択されるかどうかをコントロールします。

XRTTBL_SELECT_NONEにセットされていると、(XmNxrtTblSelectCallbackで指定される)Selectコールバックは、ユーザがセルをトラバースまたは選択しようとした時に呼び出されません。選択されたセルのリスト(XmNxrtTblSelectedCellList)は、このリソースの設定によって変更されません。

XmNxrtTblSensitiveContext　Boolean　True　CSG

現在のcontextに対して、セルがセンシティブであるかどうかを指定します。インセンシティブなセルは灰色で表示されますが、ユーザはそのセルを選択したり、そこにトラバースすることができます。テーブル全体がインセンシティブな場合は、このリソースは何もしません。

XmNxrtTblSensitiveSeries　XrtTblSeries *　NULL　CSG

XrtTblBooleanValue構造体のseriesとしてセルのセンシティビティを指定します。このリソースはContextValueCallbackで使用できます。詳細は、6.11.5 ”contextを管理する”の章を参照してください。

XmNxrtTblSetTextCursorPos　Boolean　False　CSG

XmTextウィジェットの挿入カーソルを、セル内のマウス・クリックの位置にセットするかどうかを指定します。

XmNxrtTblTextInitSelectがセットされている場合は、このリソースをセットしないでください。

XmNxrtTblShadowThickness　Dimension　2　CSG

セルとラベルの周囲の罫線の幅を指定します。

XmNxrtTblSortCallback　XtCallbackList　NULL　CSG

*XrtTblSortTable()*のコールの後で呼び出されるコールバックのリストを指定します。各ルーチンには、XrtTblSortCallbackStructのポインタが渡されます。

```
typedef struct {
  int              reason;  // Read-only
  XEvent               *event;  // Read-only
  int              column;  // Read-only
  XrtTblCompareStruct *data;    // Read-only
  int              num_rows; // Read-only
} XrtTblSortCallbackStruct;
```

*reason*は常にXRTTBL_REASON_SORTです。*event*はNULL、*column*はソートされる列です。*data*は新たにソートされた順番(data[i].rowは、行iへ移動)、*num_rows*は、データの長さです。

XmNxrtTblSpace　Dimension　6　CSG

スクロールバーとセル/ラベルの間をpixel数で指定します。このリソースの値は、(XmNxrtTblHorizScrollBarPositionとXmNxrtTblVertScrollBarPositionで指定される)スクロールバーの接続している場所には影響されません。

XmNxrtTblSpanList　XrtTblRange**　NULL　CSG

現在結合(span)されているセルまたはラベル範囲のNULL終端リストを指定します。結合された範囲は、複数のセルまたはラベルが一緒になり一つのセルであるように処理されます。結合された各範囲は、次の構造体で定義されます。

```
typedef struct {
  int      start_row, start_column;
  int      end_row, end_column;
} XrtTblRange;
```

(*start_row*と*start_column*で指定される)左上のセルは、結合範囲の元になるセルです。元のセルのセル/ラベル値や属性が、結合されているセルに表示されます。結合範囲の属性は必ず元のセルにセットしてください。結合範囲は、セルとラベルの両方を含むことはできません。また、結合範囲には必ず二つ以上のセル/ラベルが必要です。一つのセルの範囲が指定された場合は、このリストから削除されます。

XmNxrtTblStringPickListContext　String *　NULL　CSG

現在のcontextに対して有効な値のリストを指定します。このリストの終端はNULLにしてください。テーブルは、渡された値の内部コピーを作成します。

XmNxrtTblStringPickListSeries　XrtTblSeries *　NULL　CSG

XrtTblPickListValue構造体のseriesとして、picklistを指定します。

XmNxrtTblText　Widget　NULL　SG

セルの編集に使用されるXmTextウィジェットのIDを指定します。

XmNxrtTblTextInitSelect Boolean False SG

トラバースされた時、カレント・セルのテキストを選択されている状態にするかどうかを指定します。Trueにセットされている時は、*XmTextSetSelection()*が呼び出されテキストが選択されます。XmNxrtTblSetTextCursorPosがセットされている場合は、このリソースはセットしないでください。

XmNxrtTblTopRow int dynamic CSG

ウィンドウの上部に(部分的にでも)表示されるフリーズされていない行を指定します。このリソースは、テーブルのスクロールによってアップデートされます。このリソースがセットされると、テーブルはその行をウィンドウの上部に表示するようにスクロールします。このリソースは、常に、フリーズされている行数と同じかそれ以上です。

XmNxrtTblTopShadowContext Pixel XRTTBL_NOVALUE CSG

現在のcontextの上部の影の色を指定します。XRTTBL_NOVALUEにセットされている場合は、XmNxrtTblBackgroundContextから自動的に適当な色が選択されます。このリソースは、必ず、Motif Pixel色値にセットしてください。このリソースには、対応するSeriesリソースがありません。XRT/tableは、影の色をXmNxrtTblBackgroundSeriesに記憶します。

XmNxrtTblTrackCursor Boolean True CSG

ユーザがウィジェット上でマウスを動かした時に、マウス・ポインタを追跡させるかどうかを指定します。デフォルトでは、XRT/tableはマウス・ポインタを移動し、その場所でユーザが可能なインタラクションを示す様にポインタの形を変更します。Falseであると、テーブルはアプリケーションで定義されているポインタを使用します。

XmNxrtTblTrackCursorCallback XtCallbackList NULL CSG

ユーザがテーブル上でマウス・ポインタを移動した時に呼び出されるコールバックのリストを指定します。TrackCursorコールバックは、ポインタがセル/ラベルの境界を交差する時に呼び出されます。アプリケーションはこのコールバックを使用して、テーブルの特定な領域でカーソルの形が変化するように設定できます。XrtTblTrackCursorCallbackStrucntのポインタが各コールバックに渡されます。*XtAddCallback()*を使用してリストにコールバックを追加してください。コールバックは次のように構成されます。

```
void
CallbackProcedure(
  Widget        table,
  XtPointer     client_data,
  XtPointer     call_data// points to an XrtTblTrackCursorCallbackStruct
)
```

XrtTblTrackCursorCallbackStructは次のように定義されます。

```
typedef struct {
  XrtTblReason          reason;          // Read-only
  XEvent                *event;          // Read-only
  int                   row, column; // Read-only
  XrtTblBorderSides  sides;          // Read-only
  Boolean               enter_cell;      // Read-only
  Cursor                old_cursor;      // Read-only
  Cursor                new_cursor;
} XrtTblTrackCursorCallbackStruct;
```

*reason*は、セル/ラベルを離れる時にXRTTBL_REASON_CURSOR_LEAVE、セル/ラベルの境界ではXRTTBL_REASON_CURSOR_EDGE、セル/ラベル内では

XRTTBL_REASON_CURSOR_INSIDEです。[1] *row*、*column*はマウス・ポインタの位置を指定(セル/ラベルの外ではXRTTBL_NOVALUE)。*sides*はマウス・ポインタがある境界の位置、*enter_cell*は、それ以前に報告されていないセル内にポインタがあればTrueです。*old_cursor*と*new_cursor*は現在と新しいマウス・ポインタです。コールバック・ルーチンは、カーソルが変化しないように*new_cursor*を*old_cursor*に変更することができます。

XmNxrtTblTraversableContext　Boolean　True　CSG

現在のcontextのセルが、ユーザまたはアプリケーションによってトラバースできるかどうかを指定します。トラバースできないセルは、ユーザによって編集できません。

XmNxrtTblTraversableSeries　XrtTblSeries*　NULL　CSG

XrtTblBooleanValue構造体のseriesとして、セルのトラバースを指定します。

XmNxrtTblTraverseCellCallback　XtCallbackList　NULL　CSG

ユーザがセルからセルにトラバースする時に呼ばれるコールバックのリストを指定します。TraverseCellコールバックは、*TraverseToCell()*アクションによって開始され、 XmTextウィジェットがアンマップされテーブルがフォーカスを得る時に呼び出されます。このコールバックは、ユーザがセルの編集を終了した後で、XmTextウィジェットが次のセルに移動する前に呼び出されます。XrtTblTraverseCellCallbackStructのポインタが各コールバックに渡されます。*XtAddCallback()*を使用してリストにコールバックを追加してください。コールバックは次のように構成されます。

```
void
CallbackProcedure(
  Widget        table,
  XtPointer     client_data,
  XtPointer     call_data//points to an XrtTblTraverseCellCallbackStruct
)
```

XrtTblTraverseCellCallbackStructは次のように定義されます。

```
typedef struct {
  XrtTblReason  reason;          // Read-only
  XEvent        *event;          // Read-only
  String        *params;         // Read-only
  Cardinal      num_params;      // Read-only
  int           row, column; // Read-only
  int           next_row;
  int           next_column;
} XrtTblTraverseCellCallbackStruct;
```

*reason*は常にXRTTBL_REASON_TRAVERSE_CELL。*row*と*column*はカレント・セルの位置を指定。*next_row*と*next_column*はトラバースされるセルの位置です。

XmNxrtTblUserDataContext　XtPointer　NULL　CSG

現在のcontextに、アプリケーションが付属として持つことができるユーザ・データのポインタを指定します。このリソースは、XRT/tableによって使用されません。

XmNxrtTblUserDataSeries　XrtTblSeries*　NULL　CSG

XrtTblPointerValue構造体のseriesとして、ユーザのデータを指定します。

---

[1] *reason*は、マウスがセル/ラベルの外(スクロールバーやタイトル等）でも、XRTTBL_REASON_CURSOR_INSIDEになります。但し、この場合、*row*と*column*はXRTTBL_NOVALUEです。

XmNxrtTblUseXmGetColors Boolean True CSG

セルの影の色の計算に、*XmGetColors()*を呼び出すかどうか指定します。Falseの場合は、XRT/table 2.0のアルゴリズムを使用して影の色を計算します。

XmNxrtTblValidateCellCallback XtCallbackList **下記参照** CSG

セル/ラベルが編集され、その編集がセル/ラベル値の構造体に確定される前と後に呼び出されるコールバックのリストを指定します。ValidateCellコールバックは、*TraverseToCell()*と*CommitEdit()*によって呼び出されます。XrtTblValidateCellCallbackStructのポインタが、各コールバックに渡されます。デフォルトのコールバックは*XrtTblCheckDispatcher()*で、このルーチンは編集されたセル/ラベル値を検証し、XrtTblValidateCellCallbackStruct構造体を変更します。*XtAddCallback()*を使用してリストにコールバックを追加してください。コールバックは次のように構成されます。

```
void
CallbackProcedure(
  Widget        table,
  XtPointer     client_data,
  XtPointer     call_data//points to an XrtTblValidateCellCallbackStruct
)
```

XrtTblValidateCellCallbackStructは次のように定義されます。

```
typedef struct {
  XrtTblReason     reason;          // Read-only
  XEvent           *event;          // Read-only
  int          row, column; // Read-only
  Boolean      value_changed;   // Read-only
  XrtTblDataType   datatype;        // Read-only
  XtArgVal         *converted_value; // Read-only
  XtPointer        user_data;       // Read-only
  String           value;
  Boolean      doit;
  Boolean      bell;
} XrtTblValidateCellCallbackStruct;
```

reason は XRTTBL_REASON_VALIDATE_CELL_BEGIN(編集を確定する前)または XRTTBL_REASON_VALIDATE_CELL_END(編集を確定した後)です。*row*と*column*はカレント・セルの位置を指定。*value_changed*は、ユーザがセル値を変更しようとした時はTrue。*datatype*はセルのXmNxrtTblDataTypeContext値。*converted_value*は、検証ルーチンによって変換された値のポインタ。*user_data*はセルのXmNxrtTblUserDataContext値。*value*は、*doit*がTrueであれば確定される値です。値を確定させない時は*doit*をFalseにします。*bell*は編集されたセルが確定されていないことを示すためにターミナルのベルを鳴らすかどうか指定します。(*doit*がFalseの時*bell*はTrueです。その反対の場合も同じです。)

ValidateCellコールバックは、値がインタラクティブにセルの編集によって変更される場合(XmNxrtTblValidatePolicy が XRTTBL_VALIDATE_USER_EDIT) 、 *reason* が XRTTBL_REASON_VALIDATE_CELL_ENDの時にだけ、呼び出されます。

XmNxrtTblValidatePolicy XrtTblValidatePolicy XRTTBL_VALIDATE_USER_EDIT CSG

いつセルやラベルへの変更を(XmNxrtTblDataTypeContextで指定される)データ型に対して検証するか指定します。XRTTBL_VALIDATE_USER_EDITの場合は、ユーザによる変更のみが検証されます。XRTTBL_VALIDATE_SETであると、アプリケーションによる変更が検証されます。XRTTBL_VALIDATE_ALWAYSの場合はユーザとアプリケーション両方による

変更が検証されます。XRTTBL_VALIDATE_NEVERであれば検証は行われません。

XRT/tableは、XmNxrtTblValidateCellCallbackで指定されるValidateCellコールバックを使用してセル/ラベルの編集を検証します。

XmNxrtTblVerticalScrollBar Widget NULL SG

縦のスクロールバーのウィジェットIDを指定します。スクロールバーを表示しない時は、XmNxrtTblDisplayVertScrollBarを使用し、これをNULLにセットしないでください。

XmNxrtTblVertScrollBarBottomAttachmentXrtTblAttachment XRTTBL_ATTACH_CELLS CSG

縦のスクロールバーの下部がどのようにテーブルに接続されるか指定します。XRTTBL_ATTACH_CELLSの場合は、スクロールバーの下部は表示されているセルの下端に、XRTTBL_ATTACH_SIDEであれば、スクロールバーの下部がウィジェットの下端に来ます。

XmNxrtTblVertScrollBarPlacement XrtTblColumnPlacement XRTTBL_PLACE_RIGHT CSG

縦のスクロールバーを配置する場所を指定します。XRTTBL_PLACE_RIGHTであればスクロールバーはテーブルの右側、XRTTBL_PLACE_LEFTであればテーブルの左側に配置されます。

XmNxrtTblVertScrollBarPosition XrtTblSBPosition XRTTBL_SBPOSITION_CELL CSG

縦のスクロールバーをどのようにテーブルに接続するか指定します。XRTTBL_SBPOSITION_CELLSであると、スクロールバーの端はセル/ラベルのビューポート(viewport)の端の位置に来ます。XRTTBL_SBPOSITION_SIDEの場合、スクロールバーはウィジェットの端の位置に来ます。XmNxrtTblSpaceは、スクロールバーとテーブルの間の距離を指定します。

XmNxrtTblVertScrollBarTopAttachment XrtTblAttachment XRTTBL_ATTACH_CELLS CSG

縦のスクロールバーの上部がどのようにテーブルに接続されるか指定します。XRTTBL_ATTACH_CELLSの場合は、スクロールバーの上部は表示されているセルの上端に来ます。XRTTBL_ATTACH_SIDEであれば、スクロールバーの上部がウィジェットの上端に来ます。

XmNxrtTblVisibleColumns int dynamic CSG

現在(部分的にでも)表示されている、フリーズされていない列数を指定します。ユーザがインタラクティブに列またはウィジェットをリサイズすると、このリソースはアップデートされます。このリソースがセットされると、ウィジェットはリサイズの決まり(XmNxrtTblAllowResize)を置き換えて、指定された列数を表示するようリサイズします。0にセットすると、全ての列を表示するようにリサイズし、XmNxrtTblVisibleColumnsをXmNxrtTblNumColumnsと同じにセットします。(このリソースがデフォルトの時にウィジェットを作成する場合も同じようにセットされます。)

XmNxrtTblVisibleRows int dynamic CSG

現在(部分的にでも)表示されている、フリーズされていない行数を指定します。ユーザがインタラクティブに行またはウィジェットをリサイズすると、このリソースはアップデートされます。このリソースがセットされると、ウィジェットはリサイズの決まり(XmNxrtTblAllowResize)を置き換えて、指定された行数を表示するようリサイズします。0にセットすると、全ての行を表示するようにリサイズし、XmNxrtTblVisibleRowsをXmNxrtTblNumRowsと同じにセットします。(このリソースがデフォルトの時にウィジェットを作成する場合も同じようにセットされます。)

XmNxrtTblWidgetClassList　XrtTblWidgetClass*　**下記参照**　CSG

各ウィジェット・クラス名に対するウィジェットのクラス・ポインタのNULL終端リストを指定します。このリストは、XRT/tableが認識し、リソース・ファイルから作成できるウィジェット・クラスを定義します。セットされると、値が現在のリストに追加されます。デフォルトのリストは、OSF/Motif、XRT/table、XrtLabelウィジェットのクラスとクラス・ポインタを持ちます。

XmNxrtTblWidgetContext　Widget　NULL　CSG

現在のcontextのセル/ラベルに表示されるウィジェットを指定します。セットされると、表示に必要な数のコピー(クローン)が作成されます。XRT/tableは、OSF/Motif、XRT/table、XRT/field、XrtLabelウィジェットの他、XmNxrtTblWidgetClassListに追加されたクラスを認識します。認識されないウィジェット・クラスにセットする時は、CreateWidgetコールバックを使用してウィジェットを直接作成してください。

(XRTTBL_LABEL, XRTTBL_LABEL)というcontextを使用して、行と列ラベルの交差する場所にウィジェットを配置することができます。

このリソースがNULLにセットされると、そのcontextに既に指定してあるウィジェットは、非管理(unmanaged)になります。(但し、ウィジェットは破棄されません。)

XmNxrtTblWidgetSeries　XrtTblSeries*　NULL　CSG

XrtTblWidgetValue構造体のseriesとして、セル/ラベルに表示されるウィジェットを指定します。

XmNxrtTblWidgetSetValueCallback　XtCallbackList　NULL　CSG

ウィジェットを含む各セルに対して呼び出されるコールバックのリストを指定します。ルーチンは、リソースがウィジェットにセットされる時に呼ばれ、アプリケーションは、そのリソース値を不可にしたり変更したりすることができます。各ルーチンには、XrtTblWidgetSetValueCallbackStructのポインタが渡されます。

```
typedef struct {
  int     reason;      // Read-only
  XEvent      *event;      // Read-only
  int     row, column;// Read-only
  Widget      widget;      // Read-only
  Arg     *arglist;    // Read-only
  int     num_arglist;
  Boolean doit;
} XrtTblWidgetSetValueCallbackStruct;
```

*reason*はXRTTBL_REASON_SETVALUE_BEGINまたはXRTTBL_REASON_SETVALUE_ENDのどちらかです。*event*は常にNULL、*row*と*column*はセルまたはラベルの位置です。*arglist*はセットされるargsのリストで、argsが適用される場合は、*doit*はTrueにセットします。

XmNxrtTblWidgetShadowContext　Boolean　False　CSG

現在のcontextで、セルまたはラベルがウィジェットを持っている時に影を描くかどうかを指定します。Falseの場合は、ウィジェットがセルの端まで表示されます。

XmNxrtTblWidgetShadowSeries　XrtTblSeries*　NULL　CSG

ウィジェットを持つセルまたはラベルの影を描くかどうかをXrtTblBooleanValue構造体のseriesとして指定します。

XmNxrtTblWordWrapContext　Boolean　False　CSG

現在のcontextで、テキストがワードラップ(スペース、tab、newlineによって改行)して表示されるかどうかを指定します。このリソースの設定により、セルが編集される時は、XmTextのXmNWordWrapリソースもセットされます。

XmNxrtTblOverflowContextまたはXmNxrtTblOverflowSeriesをセットしている時は、このリソースはセットしないでください。

XmNxrtTblWordWrapSeries　XrtTblSeries*　NULL　CSG

XrtTblBooleanValue構造体のseriesとして、セルまたはラベルのワードラップを指定します。

XmNxrtTblOverflowContextまたはXmNxrtTblOverflowSeriesをセットしている時は、このリソースはセットしないでください。

XmNxrtTblXmStringGetMode　XrtTblXmStringGet　XRTTBL_GET_XMSTRING　CSG

テーブルから、XmStringリソースがどのようにして取り出されるか指定します。

XRTTBL_GET_STRING　Stringへ変換される

XRTTBL_GET_XMSTRING　XmStringとして取得する

どちらの場合も、戻り値は解放しないでください。

XmNxrtTblXmStringSetMode　XrtTblXmStringSet　**下記参照**　CG

アプリケーションによって、XmStringリソースがどのようにセットされるか指定します。

XRTTBL_SET_EXAMINE　ランタイム時に決定される

XRTTBL_SET_STRING　Stringとしてセットされる

XRTTBL_SET_XMSTRING　XmStringとしてセットされる

Motif 2.xを使用している時は、このリソースをXRTTBL_SET_EXAMINEにセットすることはできません。XRTTBL_SET_EXAMINEにセットすると、アプリケーションが終了してしまいます。

Motif 1.2では、このリソースのデフォルト値はXRTTBL_SET_EXAMINEです。Motif 2.xでは、デフォルト値はXRTTBL_SET_XMSTRINGです。

XrtTblCellValues構造体が直接セットされている場合は(構造体自身が使用する型を指定するので)、このリソースの値は無視されます。

XmNxrtTblXmStringSetModeがXRTTBL_SET_STRINGにセットされると、XmStringリソースは内部でStringとして記憶されます。

A.3 Constraintリソース

XRT/tableは、テーブルの子ウィジェットのconstraintリソースを定義します。

XmNxrtTblBottomOffset　int　XRTTBL_NOVALUE　CSG

タイトルの下部とセル/ラベルの下部(XmNxrtTblLocationがXRTTBL_LOCATION_BOTTOMの時は、ウィンドウの下部)の間をpixel数で指定します。テーブルの左または右に位置するタイトルは、このリソースとXmNxrtTblTopOffsetがXRTTBL_NOVALUEであると、縦方向の中央に配置されます。

XmNxrtTblLeftOffset　int　XRTTBL_NOVALUE　CSG

タイトルの左端とセル/ラベルの左端(XmNxrtTblLocationがXRTTBL_LOCATION_LEFTの時は、ウィンドウの左端)の間をpixel数で指定します。テーブルの上または下に位置するタイトルは、このリソースとXmNxrtTblRightOffsetがXRTTBL_NOVALUEであると、横方向の中央に配置されます。

XmNxrtTblLocation　XrtTblLocation　XRTTBL_LOCATION_NONE　CSG

タイトル・ウィジェットを配置する位置を指定します。有効な値は、XRTTBL_LOCATION_LEFT、XRTTBL_LOCATION_RIGHT、XRTTBL_LOCATION_TOP、XRTTBL_LOCATION_BOTTOM、またはXRTTBL_LOCATION_NONE(タイトルとしては表示されない)です。

Managerウィジェットがタイトルとして使用される時は、子ウィジェットが追加された後に(Managerウィジェットの)XmNxrtTblLocationをセットしてください。

XmNxrtTblResizeToCell　Boolean　True　CSG

セル内のウィジェットの配置を指定します。Falseの場合、ウィジェットはセルのXmNxrtTblAlignmentContextの値に従って配置されます。Trueの場合は、ウィジェットはセルのサイズに合うようにリサイズされます。

XmNxrtTblRightOffset　int　XRTTBL_NOVALUE　CSG

タイトルの右端とセル/ラベルの右端(XmNxrtTblLocationがXRTTBL_LOCATION_RIGHTの時は、ウィンドウの右端)の間をpixel数で指定します。テーブルの上または下に位置するタイトルは、このリソースとXmNxrtTblLeftOffsetがXRTTBL_NOVALUEであると、横方向の中央に配置されます。

XmNxrtTblTopOffset　int　XRTTBL_NOVALUE　CSG

タイトルの上部とセル/ラベルの上部(XmNxrtTblLocationがXRTTBL_LOCATION_TOPの時は、ウィンドウの上部)の間をpixel数で指定します。テーブルの右または左に位置するタイトルは、このリソースとXmNxrtTblBottomOffsetがXRTTBL_NOVALUEであると、縦方向の中央に配置されます。

XmNxrtTblWidgetLocation　XrtTblContext*　NULL　CSG

テーブル内でウィジェットの位置を指定します。有効なcontextならどれでも使用できます。このリソースは、テーブルのXmNxrtTblWidgetContextをウィジェットの値にセットすることと同じ役割を持ちます。

A.4 継承されたリソース

XRT/tableは、親のクラスからリソースを継承します。次のテーブルは、リソースとそれを定義する親のクラスのリストです。詳細はMotifまたはXt関連の資料をご覧ください。

Resource	Inherited From
XmNaccelerators	Core
XmNancestorSensitive	Core
XmNbackground	Core
XmNbackgroundPixmap	Core
XmNborderColor	Core
XmNborderPixmap	Core
XmNborderWidth	Core
XmNbottomShadowColor	XmManager
XmNbottomShadowPixmap	XmManager
XmNchildren	Composite
XmNcolormap	Core
XmNdepth	Core
XmNdestroyCallback	Core
XmNforeground	XmManager
XmNheight	Core
XmNhelpCallback	XmManager
XmNhighlightColor	XmManager
XmNhighlightPixmap	XmManager

Resource	Inherited From
XmNinitialFocus	XmManager
XmNinitialResourcesPersistent	Core
XmNinsertPosition	Composite
XmNmappedWhenManaged	Core
XmNnavigationType	XmManager
XmNnumChildren	Composite
XmNscreen	Core
XmNsensitive	Core
XmNshadowThickness	XmManager
XmNstringDirection	XmManager
XmNtopShadowColor	XmManager
XmNtopShadowPixmap	XmManager
XmNtranslations	Core
XmNtraversalOn	XmManager
XmNuserData	XmManager
XmNwidth	Core
XmNx	Core
XmNy	Core

B

手続きと方法

XRT/tableの手続きと方法のアルファベット順の一覧です。"(source)"とある手続きには、ソースコードが提供されています。ソースは、*$XRTHOME/src/table/util*にあります。手続きを変更する時は、他のディレクトリにコピーしてください。(名前を変更しない場合は、XRT/tableの前にリンクしてください。)

XmCreateXrtTable()

XRT/tableウィジェットを作成します。

```
Widget
XmCreateXrtTable(
  Widget   parent,
  char *name,
  Arg  *arglist,
  int  argcount
  )
```

*parent*は親、*name*は作成されるウィジェット名、*arglist*は、(*XtSetArg()*で作成される)*argcount*引数のリストです。

XrtTblAddCallback()

XRT/tableのコールバック・リストにコールバックを追加します。この方法は、テーブルにコールバックの追加を通知すること以外は、*XtAddCallback()*と同じです。

```
void
XrtTblAddCallback(
  Widget       table,
  String       cb_name,
  XtCallbackProc    cb,
  XtPointer    tag
)
```

*table*はXRT/tableウィジェット。*cb_name*はコールバック・リストのリソース名、*cb*は追加されるコールバックです。*tag*は、コールバック手続きに渡される引数です。

`XmNxrtTblCellValueCallback`と`XmNxrtTblLabelCreateCallback`をセットする時には必ず*XrtTblAddCallback()*を使用します。

XrtTblAddColumns()

テーブルに新しい列を挿入し、既に存在するセル値と属性を右にシフトします。*XrtTblAddColumns()*は、引数が無効な場合は`False`を返します。

```
Boolean
XrtTblAddColumns(
  Widget  table,
  int position,
  int num_columns,
  Boolean shift_labels,
  void    **values,
  int num_values
  )
```

*table*はXRT/tableウィジェット。*position*は、新しい列が追加される場所の列インデックス(`XmNxrtTblNumColumns`の値にセットしてあれば、テーブルの最後の列の後に挿入されます)。*num_columns*は追加される列の数。*shift_labels*は既に存在する列のラベルを新しい列の右にシフトするかどうかを指定。*values*はXmStringsまたはStrings (どちらになるかは、`XmNxrtTblXmStringSetMode`の値に依存)の配列。*num_values*は*values*にあるエレメントの数です。カレント・セルが*position*かその下にあれば、編集はキャンセルされ、XmTextウィジェットはアンマップされます。

XrtTblAddRows()

テーブルに新しい行を挿入し、既に存在するセル値と属性を下にシフトします。*XrtTblAddRows()*は、引数が無効であると`False`を返します。

```
Boolean
XrtTblAddRows(
  Widget  table,
  int position,
  int num_rows,
  Boolean shift_labels,
  void    **values,
  int num_values
  )
```

*table*はXRT/tableウィジェット。*position*は新しい行が追加される行(`XmNxrtTblNumRows`の値にセットしてあれば、行はテーブルの最後の行の後に追加されます)。*num_rows*は追加される行の数。*shift_labels*は既に存在する行のラベルを新しい行の下にシフトするかどうかを指定。*values*はXmStringsまたはStrings (どちらになるかは、`XmNxrtTblXmStringSetMode`の値に依存)の配列。*num_values*は*values*にあるエレメントの数。カレント・セルが*position*かその下にある時は、編集はキャンセルされ、XmTextウィジェットはアンマップされます。

XrtTblCancelEdit()

インタラクティブなセルの編集をキャンセルし、場合によってはカレント・セルをアンマップします。ユーザによってカレント・セルに入力または変更されたテキストは破棄されます。*XrtTblCancelEdit()*は、進行中の編集がない時や引数が無効な場合はFalseを返します。

```
Boolean
XrtTblCancelEdit(
  Widget  table,
  Boolean unmap
  )
```

*table*はXRT/tableウィジェット。*unmap*はカレント・セルをアンマップするかどうかを指定します。

XrtTblCellValuesCopy()

セル値の構造体全体をコピーします。コピーされた構造体が返されます。

```
XrtTblCellValues *
XrtTblCellValuesCopy(
  XrtTblCellValues *cells
  )
```

XrtTblCheckDispatcher()

セル/ラベル値を検証するデータ検証ルーチンを呼び出します。この手続きは、(XmNxrtTblDataTypeContextで指定される)セル/ラベルのデータ型を(XmNxrtTblDataTypeListで指定される)検証ルーチンのリストと照合し、そのデータ型に指定されている検証ルーチンを呼び出します。

```
void
XrtTblCheckDispatcher(
  Widget       table,
  XtPointer    call_data,
  XtPointer    cbs
  )
```

*call_data*は、*XtAddCallback()*を登録する時にセットされます。(デフォルトはNULL) *cbs*はXrtTblValidateCellCallbackStruct構造体のポインタです。

XrtTblCheckIsBoolean() (source)

セル/ラベル値がBooleanかどうか検証します。ストリングが、"t"、"true"、"yes"、"on"、"1"、"f"、"false"、"no"、"off"、"0"でなければ、この手続きは、渡されたXrtTblValidateCellCallbackStruct構造体の*doit*をFalseにセットします。ストリングの検証は、大文字小文字に関係なく行われます。ストリングが空であれば、*doit*はFalseにセットされます。

```
void
XrtTblCheckIsBoolean(
  Widget       table,
  XtPointer    call_data,
  XtPointer    cbs
  )
```

*cbs*はXrtTblValidateCellCallbackStruct構造体のポインタです。

XrtTblCheckIsDouble() (source)

セル/ラベル値がdoubleかどうか検証します。ストリングがdoubleまたは空であると、この手続きは、渡されたXrtTblValidateCellCallbackStruct構造体の*doit*をTrueにセットします。ストリングが空であれば、*converted_value*メンバは0.0にセットされます。

```
void
XrtTblCheckIsDouble(
  Widget       table,
  XtPointer    call_data,
  XtPointer    cbs
  )
```

*cbs*はXrtTblValidateCellCallbackStruct構造体のポインタです。

XrtTblCheckIsEmpty() (source)

ストリングが空(またはwhitespace文字)であれば、Trueを返します。

```
Boolean
XrtTblCheckIsEmpty(
  String       string
  )
```

XrtTblCheckIsFloat() (source)

セル/ラベル値がfloatかどうか検証します。ストリングがfloatまたは空であると、渡されたXrtTblValidateCellCallbackStruct構造体の*doit*をTrueにセットします。ストリングが空であれば、*converted_value*メンバは0.0にセットされます。

```
void
XrtTblCheckIsFloat(
  Widget       table,
  XtPointer    call_data,
  XtPointer    cbs
  )
```

*cbs*はXrtTblValidateCellCallbackStruct構造体のポインタです。

XrtTblCheckIsInt() (source)

セル/ラベル値がintegerかどうか検証します。ストリングがintegerまたは空であると、この手続きは、渡されたXrtTblValidateCellCallbackStruct構造体の*doit*をTrueにセットします。ストリングが空であれば、*converted_value*メンバは0にセットされます。

```
void
XrtTblCheckIsInt(
  Widget       table,
  XtPointer    call_data,
  XtPointer    cbs
  )
```

*cbs*はXrtTblValidateCellCallbackStruct構造体のポインタです。

XrtTblCommitEdit()

インタラクティブなセルの編集をセル値の構造体に書き込み、ユーザによる変更を保管します。場合によっては、カレント・セルをアンマップします。

XmNxrtTblValidateCellCallbackリストのコールバックが、変更が有効かどうかを確かめるために呼び出されます。*XrtTblCommitEdit()*は、進行中の編集がない時や引数が無効の時は、Falseを返します。

```
Boolean
XrtTblCommitEdit(
  Widget       table,
  Boolean      unmap
  )
```

*table*はXRT/tableウィジェット。*unmap*は、手続きが成功した時にカレント・セルをアンマップするかどうかを指定します。

XrtTblCreateSpreadsheetLabel() (source)

XmNxrtTblLabelCreateFuncリソースと共に使用され、この手続きはスプレッドシート形式のラベル(行は1から始まり、列はアルファベット順にAから並ぶ)を作成します。

```
XmString
XrtTblCreateSpreadsheetLabel(
  Widget   table,
  int row,
  int column
  )
```

XmNxrtTblXmStringSetModeがXRTTBL_SET_STRINGにセットされている時は、この手続きはStringを返します。それ以外の場合は、XmStringを返します。

XrtTblCvtPixelToString()

XRT/tableの色のリソースを設定する際に使用する色を、MotifのPixel色値から16進数のRGB値に変換します。引数が無効であるとFalseを返します。

```
Boolean
XrtTblCvtPixelToString(
  Widget   mywidget,
  Pixel    pixel,
  String   string   /* Returned */
  )
```

*mywidget*はXディスプレイでrealizeされるウィジェット。*pixel*は、Pixel色値を指定。 RGBに変換されたPixel値を持つ*string*が返されます。

XrtTblCvtStringToXmString()

StringをXmStringに変換します。*XrtTblSetXmStringConverter()*が呼び出された場合は、XRT/tableの変換プログラムが使用されます。それ以外の場合は、*XmStringCreateLtoR()*が使用されます。

```
String
XrtTblCvtStringToXmString(
  String       string
  )
```

XrtTblCvtXmStringToString()

XmStringをStringに変換します。Stringは使用後に解放してください。

```
String
XrtTblCvtXmStringToString(
  XmString      string
  )
```

XrtTblDeleteColumns()

テーブルから列を削除し、セル値を解放します。*XrtTblDeleteColumns()*は、引数が無効な場合は`False`を返します。その他の列の属性(背景色等)は、列の削除による影響は受けません。

```
Boolean
XrtTblDeleteColumns(
  Widget        table,
  int       position,
  int       num_columns,
  Boolean       shift_labels
  )
```

*table*はXRT/tableウィジェット。*position*は削除する範囲の始めの列番号。*num_columns*は削除される列の数。*shift_labels*は、削除される列の右にある列ラベルを左へシフトするかどうかを指定します。カレント・セルが*position*かその右にある時は、編集はキャンセルされ、XmTextウィジェットはアンマップされます。

XrtTblDeleteRows()

テーブルから行を削除し、セル値を解放します。*XrtTblDeleteRows()*は、引数が無効な場合は`False`を返します。その他の行の属性(背景色等)は、行の削除による影響は受けません。

```
Boolean
XrtTblDeleteRows(
  Widget        table,
  int       position,
  int       num_rows,
  Boolean       shift_labels
  )
```

*table*はXRT/tableウィジェット。*position*は削除する範囲の始めの行番号。*num_rows*は削除される行の数。*shift_labels*は、削除される行の下にある行ラベルを上へシフトするかどうかを指定します。カレント・セルが*position*かその下にある時は、編集はキャンセルされ、XmTextウィジェットはアンマップされます。

XrtTblDestroyCellValues()

`XrtTblCellValues`構造体を破棄し、メモリを解放します。

```
void
XrtTblDestroyCellValues(
  XrtTblCellValues  *cells
  )
```

*cells*は`NULL`の場合もあります。

XrtTblDrawPS()

PostScriptフォントと、デバイスに依存しないPostScriptオペレータを使用して、テーブルのEPS(encapsulated PostScript)出力を生成します。この手続きは、*XrtTblVaDrawPS()*と同様ですが、arglistを取ります。*XrtTblDrawPS()*は出力されるページ数を返します。引数の詳細は、*XrtTblVaDrawPS()*を参照してください。

```
int
XrtTblDrawPS(
  Widget       table,
  FILE         *file,
  XrtTblPSarg  *arglist,
  int      argcount
  )
```

*table*はXRT/tableウィジェット。*file*は既に開かれ、書き込み可能なファイルを指していなければなりません。*arglist*は*argcount*引数のリストです。

次は、*XrtTblDrawPS()*の使用例です。

```
int ac, num_pages;
double scale = FIT_TO_PAGE_WIDTH;
XrtTblPSarg al[20];

ac = 0;
XtSetArg(al[ac], XRTTBL_PS_SCALE, &scale); ac++;
num_pages = XrtTblDrawPS(table, fp, al, ac);
```

XrtTblEnableXrtField()

XRT/field のデータ型(XRTTBL_TYPE_FIELD_STRING 等)に対応するXmNxrtTblDataTypeContext値を持つセルの編集に、XRT/fieldを作成するかどうか指定します。この手続きは、テーブルが作成される前に呼び出してください。そうでない場合、警告メッセージが表示され、XmTextウィジェットが使用されます。また、XRT/fieldライブラリ(libxrtfield.a)がXRT/tableライブラリ(libxrttable.a)より先にリンクされるようにしてください。

```
void
XrtTblEnableXrtField(void)
```

XrtTblEventToRowCol()

Xイベントが起こったセルまたはラベルを捜します。*XrtTblEventToRowCol()*は、イベントがセルまたはラベル内でない時はFalseを返します。この手続きは、通常、アクション・ルーチンから呼び出されます。

```
Boolean
XrtTblEventToRowCol(
  Widget   table,
  XEvent   *event,
  int  *row,    /* Returned */
  int  *column  /* Returned */
  )
```

*table*はXRT/tableウィジェット。*row*と*column*はセルまたはラベルのインデックスです。

XrtTblFlash()

セルと(または)ラベルの*context*を点滅させます。セルは、指定された色で指定された時間点滅します。*XrtTblFlash()*は引数が無効であるとFalseを返します。

```
Boolean
XrtTblFlash(
  Widget       table,
  XrtTblContext*context,
  int      count,
  String       bg_color,
  String       fg_color,
  unsigned longon_duration,
  unsigned longoff_duration
  )
```

*table*はXRT/tableウィジェット。*context*は点滅させるテーブルの部分。*count*は点滅の回数(XRTTBL_MAXINTの場合は無限回数点滅します)。*bg_color*と*fg_color*は点滅の背景と前景の色(NULLの場合は、セルと反対の色を使用)。*on_duration*と*off_duration*は点滅時間をミリ秒で指定します(0は、点滅のオン(またはオフ)が500ミリ秒)。
複数のセルが同じ*on_duration*と*off_duration*を持つ場合、これらは同時に点滅します。

XrtTblGet3dData() (source)

XrtTblCellValues構造体を作成し、XRT/3dデータ・セットから変換されたデータをロードします。*XrtTblGet3dData()*は引数が無効であると、NULLを返します。

```
XrtTblCellValues *
XrtTblGet3dData(
  Xrt3dData    *data,
  String       format,
  )
```

*data*は、規則的または不規則的なグリッドのXRT/3dデータ・セット(但し、スキャッター・プロット以外)。*format*は*sprintf*フォーマット・ストリング("%.2f"等)で、数値データのdouble型からString型への変換に使用されます。

XrtTblGetCell()

テーブルに値が存在している場合はテーブルから、またはXmNxrtTblCellValueFuncかXmNxrtTblCellValueCallbackから値を取ります。XmNxrtTblXmStringSetModeがXRT_SET_STRINGであるとストリングが戻ります。

```
XmString
XrtTblGetCell(
      Widget   table,
      Int row,
      Int column
      )
```

*table*はXRT/tableウィジェット、*row*と*column*はセルまたはラベルのインデックスを指定します。戻されたXmStringは解放しないでください。

XrtTblGetCellDimensions()

セルまたはラベルのサイズを調べます。

```
Boolean
XrtTblGetCellDimensions(
  Widget   table,
  int  row,
  int  column,
  int  *pixel_width, /* Returned */
  int  *pixel_height,     /* Returned */
  int  *char_width,  /* Returned */
  int  *char_height  /* Returned */
  )
```

*table*はXRT/tableウィジェット。*row*と*column*はセルまたはラベルのインデックスを指定。*pixel_height*と*pixel_width*は、pixelで測定されるセルまたはラベルの高さと幅。*char_height*と*char_width*は、文字で計算されるセルまたはラベルの高さと幅です。

XrtTblGetCellValues()

新しいXrtTblCellValues構造体を作成し、XmNxrtTblCellValuesで指されているXrtTblCellValues構造体からコピーされた値をロードします。引数が無効である場合は、NULLが返されます。返された構造体は、使用後に*XrtTblDestroyCellValues()*で解放してください。

```
XrtTblCellValues *
XrtTblGetCellValues(
  Widget   table,
  int  start_row,
  int  start_column,
  int  end_row,
  int  end_column,
  Boolean  convert_to_String,
  )
```

*table*はXRT/tableウィジェット。*start_row*と*start_column*は範囲の始まりのセル・インデックスを指定。*end_row*と*end_column*は範囲の終わりのセル・インデックスを指定。XmString値は、*convert_to_String*がTrueであればStringに変換されます。

XrtTblGetCellValuesSize()

XrtTblCellValues構造体で、最後の空白でない行と、一番長い行の行番号を調べます。*XrtTblGetCellValuesSize()*は、構造体が空であるとNULLを返します。

```
Boolean
XrtTblGetCellValuesSize(
  XrtTblCellValues *cells,
  int      *last_row,   /* Returned */
  int      *last_column,    /* Returned */
  int      *longest_row /* Returned */
  )
```

*last_row*は最後の空白でない行。*last_column*は、*longest_row*で指定される位置の行の長さ(=セルの数 - 1)です。

XrtTblGetCurrentCell()

カレント・セルの場所を調べ、カレント・セルがない場合はFalseを返します。

```
Boolean
XrtTblGetCurrentCell(
  Widget   table,
  int *row,    /* Returned */
  int *column  /* Returned */
  )
```

*table*はXRT/tableウィジェット。*row*と*column*はセル・インデックスを指定します。

XrtTblGetGraphData() (source)

XrtTblCellValues構造体を作成し、XRT/graphデータ・セットから変換されたデータをロードします。この手続きはArrayデータ・セットのみを扱います。*XrtTblGetGraphData()*は、引数が無効であるとNULLを返します。

```
XrtTblCellValues *
XrtTblGetGraphData(
  Void     *data
  String       format,
  )
```

*data*はXRT/graphデータ・セット(XRT/graphのデータ・オブジェクト・ハンドルまたはXrtData構造体のポインタを渡すことができます)。 *format*は*sprintf*フォーマット・ストリング("%.2f"等)で、データのfloat型からString型への変換に使用されます。

XrtTblGetPreviewProperties()

プレビュー・スクロール・ウィンドウの表示に関する属性を取り出します。

```
void
XrtTblGetPreviewProperties(
      Widget       table,
      Pixel        foreground,
      Pixel        background,
      XmFontList   *fontlist
      XrtTblBorderType *border);
```

全ての値が戻ります。但し、*fontlist*はコピーされず、これを解放することはできません。

XrtTblGetRowColByWidget()

(XmNxrtTblWidgetContextで指定される)セル/ラベル内のウィジェット・セットの位置と親のテーブルを調べます。このウィジェットは、セル内のウィジェットの子の場合もあります。*XrtTblGetRowColByWidget()*は、ウィジェットがセルまたはラベル内にない時や、引数が無効な場合は、Falseを返します。

```
Boolean
XrtTblGetRowColByWidget(
  Widget       widget,
  Widget       *table,  /* Returned */
  int      *row,    /* Returned */
  int      *column  /* Returned */
  )
```

*table*は親のXRT/tableウィジェット。*row*、*column*は、テーブル内のウィジェットの位置を指定します。

XrtTblGetRowColLocation()

セルまたはラベルの左上角のpixel座標を調べます。引数が無効であると、Falseを返します。

```
Boolean
XrtTblGetRowColLocation(
  Widget       table,
  int      row, column,
  Window       *window,     /* Returned */
  int      *table_x,    /* Returned */
  int      *table_y,    /* Returned */
  int      *window_x,   /* Returned */
  int      *window_y    /* Returned */
  )
```

*table*はXRT/tableウィジェット。*window*はセル/ラベルの親のウインドウ。*table_x*と*table_y*はテーブルに対応する座標。*window_x*と*window_y*は親ウィンドウに対応する座標です。

XrtTblGetSelectedRange()

XmNxrtTblSelectedCellListで指されているXrtTblRange構造体から、現在選択されているセル範囲を取り出します。*XrtTblGetSelectedRange()*は、セルが選択されていない場合や引数が無効な時はFalseを返します。

```
Boolean
XrtTblGetSelectedRange(
  Widget       table,
  int      position,
  XrtTblRange  *range   /* Returned */
  )
```

*table*はXRT/tableウィジェット。*position*は選択されているセル・リスト内の場所。選択された範囲を持った*range*が返されます。

*XrtTblGetSelectedRange()*によって返される範囲は、上から下、左から右方向に構成されます。また、XRTTBL_MAXINTのような値は、有効な範囲の値に変換されます。

XrtTblGetSpanByRowCol()

指定されたセルまたはラベルを含む、結合(span)された範囲を調べます。*XrtTblGetSpanByRowCol()*は、セル/ラベルが結合された範囲にない場合や引数が無効な時はFalseを返します。

```
Boolean
XrtTblGetSpanByRowCol(
  Widget       table,
  int      row, column,
  XrtTblRange  *range   /* Returned */
  )
```

*table*はXRT/tableウィジェット。*row*と*column*はセル/ラベルの位置。結合された範囲を持った*range*が返されます

XrtTblGetWidgetByRowCol()

セルまたはラベルに表示されているウィジェットを返します。セル/ラベル内にウィジェットがない時やウィジェットが表示されていない時はNULLを返します。

```
Widget
XrtTblGetWidgetByRowCol(
  Widget       table,
  int      row, column
  )
```

*table*はXRT/tableウィジェット。*row*と*column*はセル/ラベルの位置を指定します。

XrtTblGetXrtFieldByType()

事前に作成されている、指定された型のXRT/fieldを見つけるか、または必要であればXRT/field子ウィジェットを作成します。

```
Widget
XrtTblGetXrtFieldByType(
    Widget       table,
    XrtTblDataType   type
    )
```

*table*はXRT/tableウィジェット。*type*はXRT/fieldウィジェットの型。`XrtTblDataType`のエレメントは、XRT/fieldの`XrtFldFieldType`のデータ型と同じです。

XrtTblImportExportListAdd()

ドラッグ&ドロップのターゲットのリストにアイテムを追加します。

```
XrtTblImportExportStruct **
XrtTblImportExportListAdd(
  XrtTblImportExportStruct      **ie_list,
  int              position,
  XrtTblImportExportStruct      *importExportElement
  )
```

*ie_list*は、追加されるターゲット・アイテムのリスト。*importExportElement*は、追加する新しいターゲット・アイテム。*position*は、リストでの新しいアイテムの場所を示し、リストの前に追加するには、これを0にセットします。アップデートされたターゲット・アイテムのリストが返されます。

XrtTblImportExportListCopy()

ドラッグ&ドロップのターゲット・アイテムのリストをコピーします。

```
XrtTblImportExportStruct **
XrtTblImportExportListCopy(
  XrtTblImportExportStruct **ie_list,
    )
```

*ie_list*は、コピーされるターゲット・アイテムのリストです。コピーが返されます。

XrtTblImportExportListDelete()

ドラッグ&ドロップのターゲット・アイテムのリストからアイテムを削除します。

```
XrtTblImportExportStruct **
XrtTblImportExportListDelete(
  XrtTblImportExportStruct      **ie_list,
  String               target
  )
```

*ie_list*は、削除の対象となるターゲット・アイテムのリスト。*target*は削除されるターゲット。アップデートされたターゲット・アイテムのリストが返されます。

XrtTblImportExportListDestroy()

ドラッグ&ドロップのターゲット・アイテムのリストを破棄します。

```
void
XrtTblImportExportListDestroy(
  XrtTblImportExportStruct **ie_list,
  )
```

*ie_list*は、破棄されるターゲット・アイテムのリストです。

XrtTblIsCellSelected()

指定されたセルが選択されているかどうか調べます。選択されている場合、この手続きはTrueを返し、選択されているセル・リストでの位置と範囲を取得します。

```
Boolean
XrtTblIsCellSelected(
  Widget       table,
  int      row, column,
  int      *range_index,/* Returned */
  XrtTblRange  *range       /* Returned */
  )
```

*table*はXRT/tableウィジェット。*row*と*column*はセルの位置を指定。*range_index*は選択されているセル・リスト内の位置。*range*は選択されている範囲を指定します。

XrtTblIsCellTraversable()

指定されたセルが現在トラバース可能であればTrueを返します。セルは、XmNxrtTblTraversableContextがTrueであればトラバース可能。ウィジェットがセル内にある時、XmNtraversalOnがTrueであれば、そのセルはトラバース可能。ラベルはトラバースできません。

```
Boolean
XrtTblIsCellTraversable(
  Widget       table,
  int      row, column
  )
```

*table*はXRT/tableウィジェット。*row*と*column*はセルの位置を指定します。

XrtTblIsCellVisible()

指定されているセルまたはラベルが表示されている時はTrueを返します。

```
Boolean
XrtTblIsCellVisible(
  Widget   table,
  int row, column
  )
```

XrtTblMakeCellVisible()

指定されたセルまたはラベルを(必要であればスクロールして)表示します。*XrtTblMakeCellVisible()*は、引数が無効であるとFalseを返します。

```
Boolean
XrtTblMakeCellVisible(
  Widget   table,
  int row, column
  )
```

XrtTblMoveColumns()

列の範囲を移動します。*XrtTblMoveColumns()*は引数が無効であるとFalseを返します。設定されている属性(背景色等)も列と一緒に移動し、Seriesリソースの値は、必要に応じてアップデートされます。

```
Boolean
XrtTblMoveColumns(
  Widget   table,
  int source,
  int num_columns,
  int destination,
  Boolean  shift_labels
  )
```

*table*はXRT/tableウィジェット。*source*は範囲の左に位置する列の番号。*num_columns*は移動する列の数。*destination*は、移動した列を挿入する位置にある列の番号。*shift_labels*は列ラベルも移動するかどうかを指定します。カレント・セルが*Min(source, destination)*かその右にあれば、編集はキャンセルされ、XmTextウィジェットはアンマップされます。

XrtTblMoveRows()

行の範囲を移動します。*XrtTblMoveRows()*は、引数が無効であるとFalseを返します。設定されている属性(背景色等)も行と一緒に移動し、Seriesリソースの値は、必要に応じてアップデートされます。

```
Boolean
XrtTblMoveRows(
  Widget   table,
  int source,
  int num_rows,
  int destination,
  Boolean  shift_labels
  )
```

*table*はXRT/tableウィジェット。*source*は範囲の一番上の行の番号。*num_rows*は移動する行の数。*destination*は、移動された行を挿入する場所にある行の番号。*shift_labels*は行ラベルも移動するかどうかを指定します。カレント・セルが*Min(source, destination)*かその下にあれば、編集はキャンセルされ、XmTextウィジェットはアンマップされます。

XrtTblMrmInitialize()

MrmにXRT/tableウィジェット・クラスを登録します。UILアプリケーションに使用されます。このルーチンは、*MrmInitialize()*の後に呼び出してください。

```
int
XrtTblMrmInitialize(void)
```

XrtTblRangeListAdd()

範囲のリストに範囲を挿入または追加し、大きくなったリストを返します。最後のエレメントはNULLにセットされます。返された範囲のリストは、使用後、*XrtTblRangeListFree()*を使用して解放してください。

```
XrtTblRange **
XrtTblRangeListAdd(
  XrtTblRange  **list,
  int      position,
  XrtTblRange  *range
  )
```

*list*は元の範囲のリストを指定(NULLであれば、位置(position)を無視して新しいリストが作成されます。*position*は*range*を挿入する場所のインデックス。(XRTTBL_MAXINTであれば、*range*はリストの最後にある範囲の後に追加されます。) *range*はリストに追加する範囲です。

XrtTblRangeListCopy()

範囲のリスト(XmNxrtTblSelectedCellListで指定される選択範囲のリスト等)をコピーします。返された範囲リストは、使用後、*XrtTblRangeListFree()*で解放してください。

```
XrtTblRange **
XrtTblRangeListCopy(
  XrtTblRange  **list
  )
```

*list*は、範囲リストを指定します。

XrtTblRangeListDelete()

範囲リストから範囲を削除し、小さくなったリストを返します。返された範囲リストは、使用後、*XrtTblRangeListFree()*で解放してください。

```
XrtTblRange **
XrtTblRangeListDelete(
  XrtTblRange  **list,
  int      position
  )
```

*list*は範囲リストを指定。*position*は削除する範囲のインデックスです。

XrtTblRangeListFind()

与えられた範囲のリストから指定された行と列を検索し、その行と列を含む範囲のうち、最初にみつかった範囲のインデックスを返します。適合するものがみつからなければ、XRTTBL_NOTFOUNDが戻ります。

```
int
XrtTblRangeListFind(
      XrtTblRange  **range,
      Int      row,
      Int      column
      )
```

*range*は検索の対象となる範囲、*row*と*column*は検索するセルを指定します。

XrtTblRangeListFindRange()

与えられた範囲のリストから指定された範囲を検索し、最初に適合する範囲のインデックスを返します。適合するものが見つからなければXRTTBL_NOTFOUNDが戻ります。

```
int
XrtTblRangeListFindRange(
      XrtTblRange  **range
      XrtTblRange  *request
      )
```

*range*は検索の対象となる範囲、*request*は検索される範囲を指定します。

XrtTblRangeListFree()

範囲リストを解放します。

```
void
XrtTblRangeListFree(
  XrtTblRange  **list
  )
```

*list*は範囲リストを指定します。

XrtTblReadAscii() (source)

新しいXrtTblCellValues構造体を作成し、ASCIIファイルから読み込まれた値をロードします。ファイルは、テーブルの行に対して区切りのあるデータを一行含んでいなければなりません。エラーが起きると、*XrtTblReadAscii()*はNULLを返します。

```
XrtTblCellValues *
XrtTblReadAscii(
  Widget   table,
  FILE *file,
  char delimiter
  )
```

*table*はXRT/tableウィジェット。*file*は読み取りアクセスのために開かれているファイルを指します。*delimiter*は各行のセル値の間にある文字です。ファイルにあるバック・スラッシュは、区切り文字をエスケープします。

XrtTblReadResources()

リソース・ファイルを読み、テーブルのリソースをファイルの値にセットします。エラーが起きると、この手続きは、Falseを返します。

```
Boolean
XrtTblReadResources(
  Widget   table,
  String   filename
  )
```

*table*はXRT/tableウィジェット、*filename*はリソース・ファイルです。

XrtTblRemoveAllCallbacks()

XRT/tableのコールバック・リストから全てのコールバックを削除します。この方法は、コールバックの削除をテーブルに通知する事以外は、*XtRemoveAllCallbacks()*と同じです。

```
void
XrtTblRemoveAllCallbacks(
  Widget   table,
  String   cb_name
  )
```

*table*はXRT/tableウィジェット。*cb_name*はコールバック・リストのリソース名です。

XrtTblRemoveCallback()

XRT/tableのコールバックのリストからコールバックを削除します。この方法は、コールバックの削除をテーブルに通知する事以外は、*XtRemoveCallbacks()*と同じです。

```
void
XrtTblRemoveCallback(
  Widget       table,
  String       cb_name,
  XtCallbackProc    cb
  )
```

*table*はXRT/tableウィジェット。*cb_name*はコールバック・リストのリソース名。*cb*は削除されるコールバックです。

XmNxrtTblCellValueCallbackとXmNxrtTblLabelCreateCallbackのリストからコールバックを削除する時は、必ず*XrtTblRemoveCallback()*を使用してください。

XrtTblRepaintRange()

XmNxrtTblRepaintがXRTTBL_REPAINT_ONの場合、セルと(または)ラベルの範囲を再描画します。Exposeコールバックは呼び出されません。引数が無効であると、Falseを返します。

```
Boolean
XrtTblRepaintRange(
  Widget       table,
  XrtTblRange  *range
  )
```

*table*はXRT/tableウィジェット。*range*は再描画する範囲を指定します。NULLであれば、テーブル全体が再描画されます。

この方法は、XmNxrtTblCellValueCallbackまたはXmNxrtLabelCreateCallbackが変更され、アプリケーションが再描画を必要とする時に呼び出します。

カレント・セルが指定範囲内にある時は、編集された値を持つXmTextウィジェットが(XrtTblCellValues構造体からコピーするか、XmNxrtTblCellValueCallbackを呼び出すことにより)リセットされます。

XrtTblRestoreAllCursors()

XRT/tableの全てのカーソルをデフォルトに戻します。

```
void
XrtTblRestoreAllCursors(
      Widget   table
      )
```

*table*はXRT/tableウィジェットです。

XrtTblSeriesAdd()

(*XrtTblSeriesCreate()*で作成された)seriesにcontextメンバを追加します。指定されたcontextが既にseries内に存在する場合は、置き換えられます。

```
void
XrtTblSeriesAdd(
  XrtTblSeries *series,
  XrtTblContext    *context,
  XtArgVal     value
  )
```

*sereis*は追加されるseries。*context*は新しいseriesメンバのcontextを指定。*value*は、新しい値を指定します。

XrtTblSeriesAddValue()

(*XrtTblSeriesCreate()*で作成された)seriesの指定位置(position)にメンバを加えます。その位置が既に存在している場合は、置き換えられます。

```
void
XrtTblSeriesAddValue(
  XrtTblSeries *series,
  int      position,
  XtArgVal     value
  )
```

*series*は追加されるseries。*position*はseries内のインデックスを指定(XRTTBL_MAXINTであれば、最後に追加されます)。*value*は新しい値を指定します。

XrtTblSeriesCopy()

seriesをコピーします。

```
XrtTblSeries *
XrtTblSeriesCopy(
  XrtTblSeries *series
  )
```

*series*は、コピーされるseriesです。

XrtTblSeriesCreate()

seriesを作成します。

```
XrtTblSeries *
XrtTblSeriesCreate(
  XrtTblValueType  type
  )
```

*type*は、作成するseriesの型を指定します。

XrtTblSeriesDelete()

seriesからメンバを削除します。

```
void
XrtTblSeriesDelete(
  XrtTblSeries *series,
  int      position
  )
```

*series*は削除の対象となるseries。*position*は削除するメンバのインデックスを指定します。

XrtTblSeriesDestroy()

seriesを破棄し、メモリを解放します。

```
void
XrtTblSeriesDestroy()
  XrtTblSeries *series
  )
```

*series*は、破棄されるseriesです。

XrtTblSeriesFind()

series内でセルまたはラベルを探します。*XrtTblSeriesFind()*は、series内でマッチするメンバのインデックスを返すかまたは、何もマッチしなければXRTTBL_NOTFOUNDを返します。

```
int
XrtTblSeriesFind(
  XrtTblSeries *series,
  int      row,
  int      column
)
```

*series*は検索されるseries。*row*と*column*は検索されるセルまたはラベルを指定します。

XrtTblSeriesIterate()

seriesのメンバの検査に使用されます。この手続きは、指定された位置(position)のseriesのメンバを返し、ポジション・カウンタを進めます。*XrtTblSeriesIterate()*は、位置(position)が無効であるとFalseを返します。

```
Boolean
XrtTblSeriesIterate(
  XrtTblSeries *series,
  int      *position,
  int      *row,        /*Returned */
  int      *column,     /*Returned */
  XtArgVal     *value       /*Returned */
  )
```

*position*はseries内のメンバ(0にイニシャライズされている必要があります)。*row*と*column*は、検索されたセルのインデックス。*value*は、その位置(position)での値です。

XrtTblSeriesSetValue()

series内の指定された位置(position)で、seriesメンバをセットします。この手続きは、指定された位置(position)が最後のseriesメンバの位置(position)より大きい場合は、新しいseriesメンバを割り当て、追加します。

```
void
XrtTblSeriesSetValue(
  XrtTblSeries *series,
  int      position,
  XrtTblContext*context
  XtArgVal     value
  )
```

*position*はseries内のメンバ。*context*はテーブルのcontext。*value*は、seriesメンバにセットされる値です。

XrtTblSetAllCursors()

XRT/tableの全てのカーソルを指定されたカーソルにセットします。

```
void
XrtTblSetAllCursors(
      Widget   table
      Cursor   cursor
      )
```

*table*はXRT/tableウィジェット、*cursor*はウィジェットに適用されるカーソルです。

XrtTblSetCell()

指定されたセルの値をセットします。また、オプションでそのセルの値を検証し、再描画させることもできます。*value*のデータ型はXmStringですが、Motif 1.2にリンクする場合は、Stringを使用することもできます。`XmNxrtTblXmStringSetMode`をセットすることにより、Stringを指定することもできます。

```
void
XrtTblSetCell(
      Widget       table,
      Int      row,
      Int      column,
      XmString     value,
      Boolean      validate,
      Boolean      redraw
      )
```

*table*はXRT/tableウィジェット、*row*と*column*はセットされるセル、*value*はそのセルにセットされる値です。*validate*は*value*を検証するかどうか、*redraw*はセルを再描画させるかどうかを指定します。

`XmNxrtTblPixelWidthContext`が`XRTTBL_VARIABLE`でも、このルーチンは、新しいデータに合うようにセルをリサイズしません。

XrtTblSetCellValues()

提供されているXrtTblCellValues構造体から、XmNxrtTblCellValuesで指されているXrtTblCellValues構造体に値をコピーします。*XrtTblSetCellValues()*は、引数が無効であればFalseを返します。

```
Boolean
XrtTblSetCellValues(
  Widget           table,
  int          start_row,
  int          start_column,
  XrtTblCellValues      *values,
  Boolean          resize
  )
```

*table*はXRT/tableウィジェット。*start_row*と*start_column*は、コピー先のXrtTblCellValues構造体でコピーの始まるセルを指定。*values*はコピー元のXrtTblCellValues構造体。*resize*は、テーブルの行と列の数を増加するかどうかを指定します。

XrtTblSetContext()

XmNxrtTblContextリソースを指定する時に使用されるcontextを定義します。この手続きは、静的ストレッジのポインタを返すので、ポインタは解放しないでください。

```
XrtTblContext *
XrtTblSetContext(
  int  row,
  int  column
  )
```

*row*と*column*はcontextを指定します。セル・インデックスの他、*row*と(または)*column*には次の特別な値を使用することができます。

XRTTBL_ALL	行または列の全てのセルとラベル
XRTTBL_LABEL	行または列ラベル
XRTTBL_ALLCELLS	行または列の全てのセル(ラベルは含まれない)

XrtTblSetPreviewProperties()

プレビュー・スクロール・ウィンドウの表示を指定します。初期値は、テーブルの属性から得られます。

```
void
XrtTblSetPreviewProperties(
      Widget       w,
      Pixel        foreground,
      Pixel        background,
      XmFontList   fontlist,/*This is copied internally*/
      XrtTblBorderType border);
```

*border*はデフォルトでXRTTBL_BORDER_NONEです。*foreground*と*background*は、XRT/tableのXmNforegroundとXmNbackground、*fontlist*はXRT/tableのデフォルトのフォント・リストです。

XrtTblSetPSarg()

XrtTblPSArgリストのリソース名と値をセットします。

```
void
XrtTblSetPSarg(
  XrtTblPSarg       arg,
  XrtTblDrawPSarg       name,
  XtArgVal          value
  )
```

*arg*はarglist。*name*は引数の名前を指定。*value*は*name*にセットされる値。*name*のデータ型がfloatであれば(XRTTBL_PS_SCALE等)、その値はポインタとして渡されなければなりません。

XrtTblSetRange()

XmNxrtTblRangeContextリソース値の、範囲contextを作成します。この手続きは、静的ストレージのポインタを返すので、ポインタは解放しないでください。

```
XrtTblRange *
XrtTblSetRange(
  int       start_row, start_column,
  int       end_row, end_column
  )
```

XrtTblSetReverseConverters() (source)

*XtConvertAndStore()*と共に使用される逆リソース変換プログラムを登録します。アプリケーションは、人間が読める形のリソース値を得ることができます。

```
void
XrtTblSetReverseConverters(void)
```

XrtTblSetXmStringConverter()

XmRXmStringリソース変換プログラムをセットし、セルやラベルのXmString値を作成します。この手続きは、テーブルが作成される前に呼び出して下さい。

```
void
XrtTblSetXmStringConverter(void)
```

変換は、次のシンタックスを用いてフォントの変更(タグ)を指定しています。

```
@font@      XmNfontListのフォントのタグ
@P@         前のフォントに戻る
\@          "@"を文字として扱う
```

例えば、"`For a @bold@free evaluation@P@ of any product`"というストリングは、次のように表示されます。[1]

For a **free evaluation** of any product

[1] フォント・リストは、”fixed,-*-times-bold-r-normal-*-140-*=bold”です。

XrtTblSortTable()

指定された列に基づき、XrtTblCellValues構造体の行をソートします。この手続きは、アプリケーションによって提供された関数を使用してソートを行います。*XrtTblSortTable()*は、セル値の関数(XmNxrtTblCellValueCallback)によって提供されたセル値をソートしません。ウィジェットはrealizeしてください。

```
Boolean
XrtTblSortTable(
  Widget       table,
  int      column,
  int      (* compare)
           (void *, void *)
  )
```

*table*はXRT/tableウィジェット。*column*はソートの基になるインデックス。*compare*は、基になる列の二つのセルから値を渡される手続きで、一番目と二番目の値を比較し、(小であれば)-1、(同じであれば)0、(大であれば)1を返します(*qsort(3)*参照)。

*compare*がNULLであると、デフォルトで、データは次の関数を使用してアルファベットの昇順にソートされます。

```
int
sort_compare(
  XrtTblCompareStruct       *row1,
  XrtTblCompareStruct       *row2
  )
{
  if (row1->string && row2->string)
       return (strcmp(row1->string, row2->string));
       else if (row1->string)
            return (-1);
       else if (row2->string)
            return (1);
       else
            return (0);
}
```

XrtTblCompareStructは、次の定義を持っています。

```
typedef struct {
  Widget   table;
  int  row, column;
  String   string;
}XrtTblCompareStruct *;
```

*XrtTblSortTable()*は、テーブルに記憶されている値だけを変更しますので、結合された範囲、セルのリソース、選択されているセルでは、この手続きは使用できません。

新しい行の順番の通知を受けるには、XmNxrtTblSortCallbackを使用してください。

XrtTblSyncScrolling()

複数のテーブルまたは、スクロールバーのスクロールを一致させます。スクロールバーの長さが異なる場合は、テーブルはそれぞれ相対的にスクロールします。

(一つのテーブルまたはスクロールバーがスクロールすると、もう一方もスクロールします。)*XrtTblSyncScrolling()*は、引数が無効な場合は`False`を返します。

```
Boolean
XrtTblSyncScrolling(
  Widget       my_widget1,
  Widget       my_widget2,
  int      orientation
  )
```

*my_widget1*と*my_widget2*は、二つのテーブル・ウィジェットか、または二つのXmScrollBarウィジェットです。*orientation*は`XmHORIZONTAL`、`XmVERTICAL`、または`XmNONE`です。

XrtTblTraverseToCell()

カレント・セルを新しい位置に移動します。始めに、*XrtTblCommitEdit()*が、カレント・セルの編集を確定するために呼び出されます。*XrtTblCommitEdit()*が`False`を返すと、*XrtTblTraverseToCell()*は`False`を返し、XmTextウィジェットを移動しません。それ以外の場合は、*XrtTblTraverseToCell()*は、指定されたセルにトラバースしようとします。そのセルがトラバース不可であれば、この手続きは`False`を返します。`XmNxrtTblEnterCellCallback`リストのコールバックは、指定されたセルに呼ばれXmTextウィジェットの編集可・不可をセットし、XmTextウィジェットはそのセルにマップされます。

```
Boolean
XrtTblTraverseToCell(
  Widget       table,
  int      row,
  int      column,
  Boolean      select
  )
```

*table*はXRT/tableウィジェット。*row*と*column*はトラバースされるセルのインデックスを指定。*select*はセルが選択されるかどうか(`XmNxrtTblSelectCallback`ルーチンを呼び出すかどうか)を指定します。

XrtTblVaDrawPS()

PostScriptフォントと、デバイスに依存しないPostScriptオペレータを使用して、テーブル・イメージのencapsulated PostScript(EPSF-2.0)出力を生成します。*XrtTblVaDrawPS()*は、*XtVaSetValues()*と同様に、*varargs*-styleインターフェースを使用し、`XrtTblVaDrawPS`型の属性と値のペアを使用します。*XrtTblVaDrawPS()*は、出力したページ数を返します。(0ページは、エラーを示します。)テーブルの印刷は3.18に解説されています。

```
int
XrtTblVaDrawPS(
  Widget       Table,
  FILE     *file,
  ...
  )
```

*table*はXRT/tableウィジェット。*file*は既に開かれ書き込み可能なファイルを指していなければなりません。次の表は、属性と値の一覧です。

属性	値/型	デフォルト
XRTTBL_PS_CELL_RANGE	XrtTblRange *	NULL(**テーブル全体**)
XRTTBL_PS_COLOR	XRTTBL_PS_AS_IS XRTTBL_PS_MONO XRTTBL_PS_AUTO	XRTTBL_PS_AS_IS
XRTTBL_PS_COL_PAGEBREAKS	int *	NONE (**必要時に区切る**)
XRTTBL_PS_FOOTER	string	NULL
XRTTBL_PS_FOOTER_FONT	string	"Courier 12"
XRTTBL_PS_FOOTER_MARGIN	double	**下記説明を参照**
XRTTBL_PS_FOOTER_POINT_SIZE	double	12.0
XRTTBL_PS_GRID_TYPE	XRTTBL_PS_AS_IS XRTTBL_PS_LINE XRTTBL_PS_NONE	XRTTBL_PS_AS_IS
XRTTBL_PS_HEADER	string	NULL
XRTTBL_PS_HEADER_FONT	string	"Courier 12"
XRTTBL_PS_HEADER_MARGIN	double	**下記説明を参照**
XRTTBL_PS_HEADER_POINT_SIZE	double	12.0
XRTTBL_PS_MARGIN_BOTTOM	double	**下記説明を参照**
XRTTBL_PS_MARGIN_LEFT	double	**下記説明を参照**
XRTTBL_PS_MARGIN_RIGHT	double	**下記説明を参照**
XRTTBL_PS_MARGIN_TOP	double	**下記説明を参照**
XRTTBL_PS_NUM_COPIES	int	1
XRTTBL_PS_ORIENTATION	XRTTBL_PS_PORTRAIT XRTTBL_PS_LANDSCAP E XRTTBL_PS_BEST	XRTTBL_PS_PORTRAIT
XRTTBL_PS_PAPERSIZE_WIDTH	double	8.5
XRTTBL_PS_PAPERSIZE_HEIGHT	double	11.0
XRTTBL_PS_ROW_PAGEBREAKS	int *	NONE (**必要時に区切る**)
XRTTBL_PS_SCALE	double FIT_TO_PAGE FIT_TO_PAGE_WIDTH FIT_TO_PAGE_HEIGHT	72.0
XRTTBL_PS_SHOWPAGE	Boolean	True
XRTTBL_PS_SHOW_COL_LABELS	XRTTBL_PS_EDGE_PAGES XRTTBL_PS_ALL XRTTBL_PS_NONE	XRTTBL_PS_EDGE_PAGES

属性	値/型	デフォルト
XRTTBL_PS_SHOW_FROZEN_COLS	XRTTBL_PS_EDGE_PAGES XRTTBL_PS_ALL XRTTBL_PS_NONE	XRTTBL_PS_EDGE_PAGES
XRTTBL_PS_SHOW_FROZEN_ROWS	XRTTBL_PS_EDGE_PAGES XRTTBL_PS_ALL XRTTBL_PS_NONE	XRTTBL_PS_EDGE_PAGES
XRTTBL_PS_SHOW_ROW_LABELS	XRTTBL_PS_EDGE_PAGES XRTTBL_PS_ALL XRTTBL_PS_NONE	XRTTBL_PS_EDGE_PAGES
XRTTBL_PS_SHOW_PIXMAPS	XRTTBL_PS_ALL XRTTBL_PS_NONE	XRTTBL_PS_ALL
XRTTBL_PS_SHOW_TITLES	XRTTBL_PS_EDGE_PAGES XRTTBL_PS_ALL XRTTBL_PS_NONE	XRTTBL_PS_EDGE_PAGES
XRTTBL_PS_SHOW_WIDGETS	XRTTBL_PS_ALL XRTTBL_PS_NONE	XRTTBL_PS_ALL
XRTTBL_PS_UNITS	XRTTBL_PS_INCHES XRTTBL_PS_CM	XRTTBL_PS_INCHES

XRTTBL_PS_CELL_RANGEは、出力される範囲の開始と終了の行と列を指定します。デフォルトではテーブル全体が出力されます。

XRTTBL_PS_COLORは、色を指定します。XRTTBL_PS_AS_ISのデフォルトの場合、カラーのXサーバではカラーに、モノクロのXサーバではモノクロに出力されます。カラーのPostScriptファイルは、モノクロのプリンタでは白黒の濃淡を調節して印刷されます。XRTTBL_PS_MONOの場合は、全ての前景色は黒、全ての背景色は白、セルの影は(印刷される場合は)灰色にセットされます。XRTTBL_PS_AUTOであれば、PostScriptファイルが印刷時に、カラーかモノクロのプリンタであるか判断してイメージを印刷します。(ほとんどのプリンタの場合)

XRTTBL_PS_COL_PAGEBREAKSとXRTTBL_PS_ROW_PAGEBREAKSは、各ページをどこで区切るかを指定する整数のリスト(終端は-1)です。各整数は、新しいページを開始する行または列のインデックスです。ページ区切りが指定されない場合は、自動的にページが区切られます。

XRTTBL_PS_FOOTER 、 XRTTBL_PS_FOOTER_FONT 、 XRTTBL_PS_HEADER 、 XRTTBL_PS_HEADER_FONTでは、特別な文字を組み込むことができます。デフォルトでは、ヘッダーとフッターのテキストは左揃えですが、"|"文字を使用することにより、中央または右揃えにすることができます。始めの"|"は次に続くテキストを中央揃えにし、二番目の"|"はテキストを右揃えにします。その他に、次のシンボルを使用することができます。

#	現在のページ番号
^	ページの合計数

%<X>　　現在の日付(*strftime*フォーマット)
<x>は*strftime*フォーマット・シンボル

\　　次に来る特別文字を文字として扱う

XRTTBL_PS_FOOTER_MARGINは、XRTTBL_PS_FOOTERがセットされている場合に定義できます。これは、フッターとページの端の間のマージンを指定します。デフォルトは、0.25 *units*(XRTTBL_PS_UNITSで指定)です。

XRTTBL_PS_GRID_TYPEは、セル/ラベルの罫線を指定します。XRTTBL_PS_AS_ISのデフォルトであれば、罫線はスクリーン上のイメージと同じように描かれます。XRTTBL_PS_LINEであれば、罫線は、黒く細い一重線です。XRTTBL_PS_NONEであると罫線は描かれません。

XRTTBL_PS_HEADER_MARGINは、XRTTBL_PS_HEADERがセットされている場合に定義できます。これは、ヘッダーとページの端の間のマージンを指定します。デフォルトは、0.25 *units*(XRTTBL_PS_UNITSで指定)です。

XRTTBL_PS_MARGIN_TOPとXRTTBL_PS_MARGIN_BOTTOMは、(イメージではなく)*paper*の上部と下部の印刷されない余白を指定します。デフォルトは、0.25 *units*(XRTTBL_PS_UNITSで指定)です。但し、次の場合はデフォルト値が異なります。

- XRTTBL_PS_HEADERがセットされ、XRTTBL_PS_ORIENTATIONがXRTTBL_PS_PORTRAIT(またはXRTTBL_PS_BESTによって方向が縦)になっている場合、XRTTBL_PS_MARGIN_TOPのデフォルト値は、0.75 unitsです。
- XRTTBL_PS_FOOTERがセットされ、XRTTBL_PS_ORIENTATIONがXRTTBL_PS_PORTRAIT(またはXRTTBL_PS_BESTによって方向が縦)になっている場合、XRTTBL_PS_MARGIN_BOTTOMのデフォルト値は、0.75 unitsです。

XRTTBL_PS_MARGIN_LEFTとXRTTBL_PS_MARGIN_RIGHTは、(イメージではなく)*paper*の左端と右端の印刷されない余白を指定します。デフォルトは、0.25 *units*(XRTTBL_PS_UNITSで指定)です。但し、次の場合は、デフォルト値が異なります。

- XRTTBL_PS_HEADERがセットされ、XRTTBL_PS_ORIENTATIONがXRTTBL_PS_LANDSCAPE(またはXRTTBL_PS_BESTによって方向が横)になっている場合、XRTTBL_PS_MARGIN_LEFTのデフォルト値は、0.75 *units*です。
- XRTTBL_PS_FOOTERがセットされ、XRTTBL_PS_ORIENTATIONがXRTTBL_PS_LANDSCAPE(またはXRTTBL_PS_BESTによって方向が横)になっている場合、XRTTBL_PS_MARGIN_RIGHTのデフォルト値は、0.75 *units*です。

XRTTBL_PS_NUM_COPIESは、印刷部数を指定します。この出力が、他のドキュメントへ挿入される場合は、1のままにしておいてください。

XRTTBL_PS_ORIENTATIONは、イメージの向きを指定します。XRTTBL_PS_LANDSCAPEは、テーブルのイメージを反時計方向に90度回転します。XRTTBL_PS_BESTに指定すると、*XrtTblVaDrawPS()*は出力される範囲と用紙のサイズを比較し、そのイメージに適した方向を選択します。例えば、サイズが8.5x11(inch)の用紙には、幅が高さより大きい範囲は横向きに出力されます。

XRTTBL_PS_PAPERSIZE_WIDTHとXRTTBL_PS_PAPERSIZE_HEIGHTは、使用される*paper*の幅と高さを指定します。この幅と高さは、イメージの方向によって変化することはありません。用紙は常にプリンタから同じ方向でフィードされ、イメージの向きには関係ないので、用紙の幅と高さは、イメージではなく用紙の上部と左端に対応します。ここで指定される数値は、XRTTBL_PS_UNITSで設定された単位を使用します。一般的な用紙のサイズは次の通りです。

- 8.5 x 11インチ (レター)
- 8.5 x 14インチ (リーガル)
- 21.0 x 29.7センチメートル (A4)
- 14.8 x 21.0センチメートル (A5)

XRTTBL_PS_SCALEは、インチに対するpixelで、テーブルのイメージのスケーリングを指定します。この属性をスクリーンの解像度にセットしてある場合、スクリーンのイメージと同等の印刷イメージが得られます。(デフォルトの72.0は、最も一般的なスクリーン解像度です。)サイズを200%に拡大する時はXRTTBL_PS_SCALEを36.0に、50%に縮小する時はXRTTBL_PS_SCALEを144.0にセットします。次の列挙された値は、その他のスケーリングを指定します。

- FIT_TO_PAGEは、範囲全体が1ページに収まるようにスケーリングします。
- FIT_TO_PAGE_WIDTHは、範囲がページ幅に収まるようにスケーリングします。
- FIT_TO_PAGE_HEIGHTは、範囲がページの高さに収まるようにスケーリングします。

XRTTBL_PS_SHOWPAGEは、PostScriptの*showpage*オペレータを、PostScript出力の各ページの終わりに置くかどうかを指定します。同じファイル・ポインタを使用して*XrtTblVaDrawPS()*を呼び出し、最後のコール以外はXRTTBL_PS_SHOWPAGEをFalseに設定することにより、幾つかのテーブル・イメージを1ページに印刷することができます。

XRTTBL_PS_SHOW_COL_LABELSとXRTTBL_PS_SHOW_ROW_LABELSは、どのようにラベルをPostScript出力させるか指定します。XRTTBL_PS_ALLは、行/列ラベルを各ページに置きます。XRTTBL_PS_NONEは、全てのページから行/列ラベルを取り去り、指定された範囲のみをプリントします。XRTTBL_PS_EDGE_PAGESは、複数ページの内、端のページにだけラベルを置き、大きなテーブルが複数のページに別れたようにプリントします。(1ページしかない時は、結果はXRTTBL_PS_ALLと同じになります。)

XRTTBL_PS_SHOW_FROZEN_COLSとXRTTBL_PS_SHOW_FROZEN_ROWSは、フリーズされたセルをどのようにPostScript出力させるか指定します。XRTTBL_PS_ALLは、フリーズされている行と列を各ページに出力します。XRTTBL_PS_NONEは、フリーズされている行と列を全てのページから取り去り、指定されている範囲だけプリントします。XRTTBL_PS_EDGE_PAGESの場合は、複数ページの内、端のページにだけフリーズされている行/列を置き、大きなテーブルが複数のページに別れたようにプリントされます。(1ページしかない時は、結果はXRTTBL_PS_ALLと同じになります。)

XRTTBL_PS_SHOW_PIXMAPは、どのようにpixmapをPostScript出力させるか指定します。XRTTBL_PS_ALLであると、プリントされる範囲のセル/ラベルにある全てのpixmapが出力に含まれます。XRTTBL_PS_NONEの場合は、出力から全てのpixmapが削除されます。

XRTTBL_PS_SHOW_TITLESは、どのようにテーブルのタイトルをPostScript出力させるか指定します。XRTTBL_PS_ALLであると、各ページに全てのタイトルを置きます。XRTTBL_PS_NONEでは、全てのページから全てのタイトルが削除されます。XRTTBL_PS_EDGE_PAGESの場合は、複数ページの内、端のページにだけタイトルを置き、大きなテーブルが複数のページに別れたようにプリントされます。(1ページしかない時は、結果はXRTTBL_PS_ALLと同じになります。)

XRTTBL_PS_SHOW_WIDGETSは、セル/ラベルにあるウィジェットをどのようにPostScript出力させるか指定します。XRTTBL_PS_ALLであると、プリントされる範囲のセル/ラベルにある全てのウィジェットが出力に含まれます。XRTTBL_PS_NONEの場合は、出力から全てのウィジェットが削除されます。

XRTTBL_PS_UNITSは、PAPERSIZE_とMARGIN_の属性の測定単位を指定します。

XrtTblWriteAscii() (source)

セル範囲にある値をASCIIファイルに書きます。この手続きは、各セルのXmString値をStringに変換します。*XrtTblWriteAscii()*は、エラーが起きるとFalseを返します。

```
Boolean
XrtTblWriteAscii(
  Widget       table,
  FILE     *file,
  char     delimiter,
  XrtTblRange  *range
  )
```

*table*はXRT/tableウィジェット。*file*は書き込み可能で既に開いているファイルを指します。*delimiter*は列の間に挿入する文字。*range*は出力されるセルを指定します。(NULLは全てのセル値を出力します。)

XrtTblWriteResources()

現在のテーブルのリソース値をファイルに書きます。エラーが起きると、Falseを返します。

```
Boolean
XrtTblWriteResources(
  Widget       table,
  FILE     *fp,
  String       prefix,
  Boolean      write_inherited
  )
```

*prefix*は、各リソース名の前に書かれるストリングで、これを利用して複数のテーブルのリソースを同じファイルに書き込むことができます。(通常、テーブル名にセットされます。) *prefix*がNULLであると、"*."が書かれます。*write_inherited*は、XmNwidth等の継承されたリソースもファイルに書くかどうかを指定します。

*XrtTblWriteResources()*は、XrtTblContextValueCallbackがセットされた時には、エントリ数が多くなることがあるので、context値を書きません。

XrtTblWriteSylk()

セル範囲にある値をSYLKフォーマットのファイルに書きます。この手続きは、各セルのXmString値をStringに変換します。*XrtTblWriteSylk()*は、エラーが起きるとFalseを返します。

```
Boolean
XrtTblWriteSylk(
  Widget       table,
  FILE     *file,
  XrtTblRange  *range
  )
```

*table*はXRT/tableウィジェット。*file*は書き込み可能で既に開いているファイルを指します。*range*は出力されるセルを指定します。(NULLは全てのセル値を出力します。)

XrtTblXYtoRowCol()

pixel座標に対応するセルまたはラベルを調べます。テーブル・ウィジェットの座標をルート・ウィンドウの座標に置き換えるには、*XtTranslateCoords()*を使用します。*XrtTblXYtoRowCol()*は、座標がセルまたはラベル内にない時は、Falseを返します。

```
Boolean
XrtTblXYtoRowCol(
  Widget       table,
  int      root_x, root_y,
  int      *row, *column   /* Returned */
  )
```

*root_x*と*root_y*は、ルート・ウィンドウに対応するpixel座標を指定。*row*と*column*は、返されたセルまたはラベルの位置です。

XtIsXrtTable()

ウィジェットがXRT/tableまたはサブクラスであるとTrueを返します。

```
Boolean
XtIsXrtTable(
  Widget       table
  )
```

C

マクロ

次の表は、テーブルの部分にアクセスする際に便利なマクロの一覧です。これらは、*$XRTHOME/include/Xm/XrtTable.h*に定義されています。

マクロ名	定義

`CELL_VALUE(cells, row, column)`	`CELL_VALUE_EXISTS(cells, row, column) \`
	`? cells->values[row]->list[column] \`
	`: NULL`

セル値のXmString**を返す。セルに値がなければ**NULL**を返す。** ***cells***は**XrtTblCellValues**構造体にある。***row***と***column***は一つのセルの位置を指定する整数。**	

`CELL_VALUE_EXISTS(cells, row, column)`	`cells && row >= 0 \`
	`&& row < cells->rows \`
	`&& cells->values[row] \`
	`&& column >= 0 \`
	`&& column < cells->values[row]->length`

セルが値を持つ場合はTrue**、そうでなければ**False**を返す。** ***cells***は**XrtTblCellValues**構造体にある。***row***と***column***は一つのセルの位置を指定する整数。**	
`IS_CELL(row, column)`	`row >= 0 && col >= 0`
指定したcontext**がセルであれば**True**、そうでなければ**False**を返す。** ***row***と***column***は、**context**を指定する整数。**	
`IS_COLUMNLABEL(row, column)`	`IS_LABEL(row, col) \` `&& row == XRTTBL_LABEL`
指定したcontext**が列ラベルであれば**True**、そうでなければ**False**を返す。** ***row***と***column***は、**context**を指定する整数。**	

`IS_LABEL(row, column)`	`row == XRTTBL_LABEL ? col >= 0 \`
	`: (col == XRTTBL_LABEL ? row >= 0 \`
	`: False)`

指定したcontext**がラベルであれば**True**、そうでなければ**False**を返す。** **row**と**column**は、**context**を指定する整数。**	

マクロ名	定義
IS_ROWLABEL(row, column)	IS_LABEL(row, col) \ && col == XRTTBL_LABEL
指定したcontextが行ラベルであればTrue、そうでなければFalseを返す。 *row*と*column*は、contextを指定する整数。	
LABEL_VALUE(labels, pos)	LABEL_VALUE_EXISTS(labels, pos) \ ? labels->list[pos] \ : NULL
ラベル値のXmStringを返す。ラベルに値がなければNULLを返す。 *labels*は行または列ラベル構造体のXrtTblXmStringList。*pos*は(行ラベルの)行または(列ラベルの)列を指定する整数。	
LABEL_VALUE_EXISTS(labels, pos)	labels && pos >= 0 \ && pos < labels->length
ラベルが値を持つ場合はTrue、そうでなければFalseを返す。 *labels*は行または列ラベル構造体のXrtTblXmStringList。*pos*は(行ラベルの)行または(列ラベルの)列を指定する整数。	

D

トランスレーションとアクション

デフォルトのトランスレーション
アクション・ルーチン

D.1 デフォルトのトランスレーション

XRT/tableは、XmManagerからトランスレーション(アクションとイベントの定義)を継承し、次のトランスレーションを定義します。

XRT/tableウィジェットのトランスレーション

イベント	アクション
~Alt ~Ctrl ~Meta ~Shift <Btn1Down>	ResizeCell(Start) TraverseToCell(Pointer) Select(Start)
Shift ~Alt ~Ctrl ~Meta <Btn1Down>	Select(Extend)
Ctrl ~Alt ~Meta ~Shift <Btn1Down>	Select(Add)
~Alt ~Meta <Btn1Motion>	ResizeCell(Move) Select(Extend)
<Key> osfPageDown	Scroll(PageDown)
<Key> osfPageUp	Scroll(PageUp)
~Alt ~Ctrl ~Meta ~Shift <Btn1Up>	Select(End) ResizeCell(End)
~Alt ~Meta <Btn1Up>	Select(End) ResizeCell(End)
Shift <Btn2Down>	Drag(Move)
Ctrl <Btn2Down>	Drag(Copy)
<Btn2Down>	Drag(Move, Move, Copy)

XRT/tableにインストールされる、デフォルトのXmTextトランスレーション

イベント	アクション
~Alt ~Ctrl ~Meta ~Shift <Btn1Down>	ResizeCell(Start) Select(Start) grab-focus()*
Ctrl ~Alt ~Meta ~Shift <Btn1Down>	move-destination()* Select(Add)
~Alt ~Ctrl ~Meta ~Shift <Btn1Motion>	ResizeCell(Move) Select(Extend) extend-adjust()*
<Key> osfPageDown	Scroll(PageDown)
<Key> osfPageUp	Scroll(PageUp)
Ctrl ~Alt ~Meta <Key> osfDown	TraverseToCell(Down) Select(Current)
Shift ~Alt ~Ctrl ~Meta <Key> Tab	TraverseToCell(Left) Select(Current)
Ctrl ~Alt ~Meta ~Shift <Key> osfLeft	TraverseToCell(Left) Select(Current)
~Alt ~Ctrl ~Meta <Key> Tab	TraverseToCell(Right) Select(Current)
Ctrl ~Alt ~Meta <Key> osfRight	TraverseToCell(Right) Select(Current)
Ctrl ~Alt ~Meta <Key> osfUp	TraverseToCell(Up) Select(Current)
~Alt ~Ctrl ~Meta ~Shift <Btn1Up>	Select(End) ResizeCell(End) extend-end()a
<Key> osfCancel	CancelEdit(False) Select(Cancel)
<Key> osfActivate	CommitEdit(False)
Ctrl ~Alt ~Meta <Key> Tab	TraverseNext()
Ctrl Shift ~Alt ~Meta <Key> Tab	TraversePrev()

* XmTextアクション

XRT/tableに使用されるXmText(カレント・セル)とXmScrollbar(テーブルのスクロールバー)には、デフォルトのトランスレーションがインストールされています。

次のトランスレーションは、セル/ラベル内のウィジェットに定義されます。

セル/ラベル内のウィジェットのトランスレーション

イベント	アクション
Shift ~Ctrl ~Meta ~Alt <Key>Tab	TraverseToCell(Left) Select(Current)
~Shift Ctrl ~Meta ~Alt <Key>osfLeft	TraverseToCell(Left) Select(Current)
~Ctrl ~Meta ~Alt <Key>Tab	TraverseToCell(Right) Select(Current)
Ctrl ~Meta ~Alt <Key>osfRight	TraverseToCell(Right) Select(Current)
Ctrl ~Meta ~Alt <Key>osfUp	TraverseToCell(Up) Select(Current)
Ctrl ~Meta ~Alt <Key>osfDown	TraverseToCell(Down) Select(Current)
Shift Ctrl ~Meta ~Alt <Key>Tab	TraversePrev()
Ctrl ~Meta ~Alt <Key>Tab	TraverseNext()

D.2 アクション・ルーチン

XRT/tableは次のアクション・ルーチンを定義します。

CancelEdit()

*XrtTblCancelEdit()*を呼び出して、カレント・セルの編集されたセル値を破棄し、`XrtTblCellValues`構造体にある元のセル値を復元します。このアクションは、XmTextウィジェットをアンマップするかどうかを指定する一つのパラメータを取ります。

`True`	XmTextウィジェットをアンマップする
`False`	XmTextウィジェットをアンマップしない

CommitEdit()

*XrtTblCommitEdit()*を呼び出して、編集されたセルを`XrtTblCellValues`構造体に書きます。*XrtTblCommitEdit()*はValidateCellコールバックを呼び出し、編集されたセル値を検証します。このアクションは、成功した場合、XmTextウィジェットをアンマップするかどうかを指定する一つのパラメータを取ります。

`True`	XmTextウィジェットをアンマップする
`False`	XmTextウィジェットをアンマップしない

Drag()

ドラッグ&ドロップ操作を実行します。このアクションは、一つかまたは三つのパラメータを取ります。

`Copy`	ドラッグされたデータをコピーする
`Move`	ドラッグされたデータを移動する(ドラッグ元のデータを破棄する)
`NoAction`	ドラッグ&ドロップ操作をしない

一つのパラメータのみが指定されている場合、ドラッグされたデータを移動するかコピーするかを示します。三つのパラメータが指定された場合は、次のようにドラッグ&ドロップ操作をコントロールします。

- 最初のパラメータは、ドラッグされている行全対がコピーされるか移動されるかを指定する
- 二番目のパラメータは、ドラッグされている列全体がコピーされるか移動されるかを指定する
- 三番目のパラメータは、ドラッグされているセル範囲がコピーされるか移動されるかを指定する

NoAction()

何もせずコールバックも呼び出されません。デフォルトでは、このアクション・ルーチンはどのイベントにも対応していません。このルーチンを呼び出すようにイベントをセットすることにより、効果的にトランスレーションを不可にすることができます。例えば、キーボードによるスクロールをオフにするには、<Key>osfPageDownと<Key>osfPageUpが*NoAction()*を呼び出すように、新しいトランスレーションを設定します。

ResizeCell()

行/列をリサイズします。*ResizeCell()*は、リサイズの開始、新しいサイズの選択、リサイズの終了、というインタラクティブなリサイズの各段階を処理します。このアクションは、XmNxrtTblResizeCellCallbackで指定されたコールバックを呼び出し、XrtTblResizeCellCallbackStruct構造体のポインタを各コールバックに渡します。インタラクティブなリサイズは、XmNxrtTblPixelHeightと(またはXmNxrtTblPixelWidthリソースの値を変更します。このアクションは、インタラクティブなリサイズの段階を指定する一つのパラメータを取ります。

Start	リサイズを開始する。*reason*がXRTTBL_REASON_RESIZE_BEGINにセットされたResizeCellコールバックが呼び出される。
Move	新しい高さ/幅を選択する。点線がアップデートされる。
End	リサイズを終了します。*reason*がXRTTBL_REASON_RESIZE_ENDにセットされたResizeCellコールバックが呼び出される。

Scroll()

テーブルをスクロールします。*Scroll()*は、XmNxrtTblLeftColumnとXmNxrtTblTopRowリソースの新しい値を計算し、それをウィジェットに適用します。このアクションは、スクロールの方向を決める一つのパラメータを取ります。

PageLeft	1ページ左にスクロールする
PageRight	1ページ右にスクロールする
PageUp	1ページ上にスクロールする
PageDown	1ページ下にスクロールする

Select()

一つかまたは複数のセル範囲を選択します。*Select()*は、選択の開始、選択の拡張、選択の終了、というインタラクティブな選択の各段階を処理します。このアクションは、XmNxrtTblSelectCallbackで指定されたコールバックを呼び出し、XrtTblSelectCallbackStruct構造体のポインタを各コールバックに渡します。インタラクティブな選択は、XmNxrtTblSelectedCellListリソースの値を変更します。このアクションは、どのようにセルが選択されたかを決める一つのパラメータを取ります。(ラベルは選択できません。)

Start	XmNxrtTblSelectionPolicyがXRTTBL_SELECT_NONEでなければ、セルを選択する。
Current	XmNxrtTblSelectionPolicyがXRTTBL_SELECT_NONEでなければ、カレント・セルを選択する。
Extend	XmNxrtTblSelectionPolicyがXRTTBL_SELECT_RANGEまたはXRTTBL_SELECT_MULTIRANGEであれば、選択の範囲を拡張する。
Add	XmNxrtTblSelectionPolicyがXRTTBL_SELECT_MULTIRANGEであれば、セルを選択する。

Cancel	現在の選択をキャンセルする。
End	選択を終了する。

TraverseNext()

テーブルの外にトラバースし、tabグループ内の次のウィジェットに移動します。

TraversePrev()

テーブルの外にトラバースし、tabグループ内の前のウィジェットに移動します。

TraverseToCell()

カレント・セルをトラバースし、必要であればテーブルをスクロールします。*TraverseToCell()*は、編集されたセルの検証、トラバース先の決定、そのセルへ入るといった、インタラクティブなトラバースの各段階を処理します。このアクションは始めに*CommitEdit()*を呼び出します。成功すると、次にトラバース先を決定し、XmNxrtTblTraverseCellCallbackで指定されているコールバックを呼び出し、XrtTblTraverseCellCallbackStruct構造体のポインタを各コールバックに渡します。その後、新しいセルを表示するようテーブルをスクロールし、(そのセルに入る前に)XmNxrtTblEnterCellCallbackで指定されているコールバックを呼び出し、XrtTblEnterCellCallbackStruct構造体のポインタを渡します。*TraverseToCell()*は、トラバースの方向を指定する一つのパラメータを取ります。

Pointer	ユーザによってクリックされたセルへトラバース。
Left	トラバース可能な最初のセルに左へトラバース。
Right	トラバース可能な最初のセルに右へトラバース。
Up	トラバース可能な最初のセルに上へトラバース。
Down	トラバース可能な最初のセルに下へトラバース。

E

データ型

XRT/tableのデータ型のアルファベット順のリスト。Cの構造体の定義も提供されています。

XrtTblAction

XmNxrtTblContextValueCallbackに使用され、アクションを指定する列挙法です。

```
XRTTBL_ACTION_CALCULATE
XRTTBL_ACTION_DRAW
```

XrtTblAlignment

XmNxrtTblAlignmentContextとXmNxrtTblAlignmentSeriesに使用され、セルやラベルのテキストの行揃えを指定する列挙法です。

```
XRTTBL_ALIGNMENT_TOPBEGIN
XRTTBL_ALIGNMENT_TOPCENTER
XRTTBL_ALIGNMENT_TOPEND
XRTTBL_ALIGNMENT_MIDDLEBEGIN
XRTTBL_ALIGNMENT_MIDDLECENTER
XRTTBL_ALIGNMENT_MIDDLEEND
XRTTBL_ALIGNMENT_BOTTOMBEGIN
XRTTBL_ALIGNMENT_BOTTOMCENTER
XRTTBL_ALIGNMENT_BOTTOMEND
```

XrtTblAnyValue

全てのXrtTblValue型構造体のユニオンです。

```
typedef union {
  XrtTblArgListValue   arg;
  XrtTblBooleanValue   b;
  XrtTblColorValue color;
  XrtTblFontListValue  f;
  XrtTblIntValue   i;
  XrtTblPickListValue  pl;
  XrtTblPixmapValue    pm;
  XrtTblPointerValue   p;
  XrtTblStringValue    s;
  XrtTblWidgetValue    w;
} XrtTblAnyValue;
```

XrtTblArgListValue

ArgList値を指定するseriesのメンバです。

```
typedef struct {
  XrtTblContext    context;
  ArgList      value;
  int      num_args;
} XrtTblArgListValue;
```

XrtTblAttachment

XmNxrtTblHorizScrollBarLeftAttachment, XmNxrtTblHorizScrollBarRightAttachment,XmNxrtTblVertScrollBarTop-Attachment,XmNxrtTblVertScrollBarBottomAttachmentに使用され、どのようにスクロールバーの端をテーブルに接続するか指定する列挙法です。

```
XRTTBL_ATTACH_CELLS
XRTTBL_ATTACH_SIDE
```

XrtTblBooleanValue

Boolean値を指定するseriesのメンバです。

```
typedef struct {
  XrtTblContextcontext;
  Boolean      value;
} XrtTblBooleanValue;
```

XrtTblBorderType

XmNxrtTblBorderTypeContextとXmNxrtTblBorderTypeSeriesに使用され、セルとラベルの罫線の種類を指定する列挙法です。

```
XRTTBL_BORDER_NONE
XRTTBL_BORDER_ETCHED_IN
XRTTBL_BORDER_ETCHED_OUT
XRTTBL_BORDER_FRAME_IN
XRTTBL_BORDER_FRAME_OUT
XRTTBL_BORDER_IN
XRTTBL_BORDER_OUT
XRTTBL_BORDER_PLAIN
```

XrtTblBorderSides

XmNxrtTblBorderSidesContextとXmNxrtTblBorderSidesSeriesに使用され、セルとラベルの罫線を描く場所を指定する列挙法です。

```
XRTTBL_BORDERSIDE_NONE
XRTTBL_BORDERSIDE_ALL
XRTTBL_BORDERSIDE_LEFT
XRTTBL_BORDERSIDE_RIGHT
XRTTBL_BORDERSIDE_TOP
XRTTBL_BORDERSIDE_BOTTOM
```

XrtTblCase

XmNxrtTblCaseContextとXmNxrtTblCaseSeriesに使用され、セルの編集の大文字･小文字変換を指定する列挙法です。

```
XRTTBL_CASE_UPPER
XRTTBL_CASE_LOWER
XRTTBL_CASE_AS_IS
```

XrtTblCellValueCallbackStruct

XmNxrtTblCellValueCallbackリストのコールバックに渡される情報を定義する構造体です。

```
typedef struct {
  XrtTblReason reason;
  XEvent       *event;
  int      row, column;
  XmString     string;
  Boolean      store;
  Boolean      free_string;
} XrtTblCellValueCallbackStruct;
```

XrtTblCellValues

セル値を含む構造体です。CELL_VALUEマクロでアクセスできます。

```
typedef struct {
  int          rows;
  XrtTblXmStringSet set_mode;
  XrtTblXmStringList    **values;
} XrtTblCellValues;
```

XrtTblColorValue

色値を指定するseriesのメンバです。

```
typedef struct {
  XrtTblContext         context;
  struct {
       String       name;
       Pixel        pixel;
       Pixel        top_shadow;
       Pixel        bottom_shadow;
  } value;
} XrtTblColorValue;
```

XrtTblColumnPlacement

XmNxrtTblFrozenColumnPlacement, XmNxrtTblRowLabelPlacement, XmNxrtTblVertScrollBarPlacementに使用され、テーブルの縦方向の領域の場所を指定する列挙型です。

```
XRTTBL_PLACE_LEFT
XRTTBL_PLACE_RIGHT
```

XrtTblCompareStruct

*XrtTblSortTable()*に使用される構造体です。

```
typedef struct {
  Widget        table;
  int       row, column;
  String        string;
} XrtTblCompareStruct;
```

XrtTblContext

セルまたはラベルの位置の定義に使用される構造体です。行や列(ラベルも含む)や、テーブル全体も指定できます。詳細は2.5を参照してください。

```
typedef struct {
  int       row, column;
} XrtTblContext;
```

XrtTblContextReason

XmNxrtTblContextValueCallbackに使用され、reasonを指定する列挙法です。

```
XRTTBL_CONTEXT_ALIGNMENT,
XRTTBL_CONTEXT_ARGLIST,
XRTTBL_CONTEXT_BACKGROUND,
XRTTBL_CONTEXT_BORDERSIDES,
XRTTBL_CONTEXT_BORDERTYPE,
XRTTBL_CONTEXT_BOTTOMSHADOW,
XRTTBL_CONTEXT_CASE,
XRTTBL_CONTEXT_CHAR_HEIGHT,
XRTTBL_CONTEXT_CHAR_WIDTH,
XRTTBL_CONTEXT_CLIP_PIXMAP,
XRTTBL_CONTEXT_CURRENCYPICKLIST,
XRTTBL_CONTEXT_DATATYPE,
XRTTBL_CONTEXT_DATEPICKLIST,
XRTTBL_CONTEXT_EDITABLE,
XRTTBL_CONTEXT_FIELD_ARGLIST,
XRTTBL_CONTEXT_FLOATPICKLIST,
XRTTBL_CONTEXT_FOREGROUND,
XRTTBL_CONTEXT_INTPICKLIST,
XRTTBL_CONTEXT_MAXLENGTH,
XRTTBL_CONTEXT_MULTILINE,
XRTTBL_CONTEXT_PIXEL_HEIGHT,
XRTTBL_CONTEXT_PIXEL_WIDTH,
XRTTBL_CONTEXT_PIXMAP,
XRTTBL_CONTEXT_PIXMAP_LAYOUT,
XRTTBL_CONTEXT_PIXMAP_SHADOW,
XRTTBL_CONTEXT_PS_FONTLIST,
XRTTBL_CONTEXT_SENSITIVE,
XRTTBL_CONTEXT_STRINGPICKLIST,
XRTTBL_CONTEXT_TOP_SHADOW,
XRTTBL_CONTEXT_USERDATA,
XRTTBL_CONTEXT_WIDGET_SHADOW,
XRTTBL_CONTEXT_WORDWRAP
```

XrtTblContextValue

XmNxrtTblContextValueCallbackリストのコールバックに渡されるcontext値の情報を持つユニオンです。

```
typedef union {
  XrtTblAlignment        alignment;
  ArgList            arglist;
  Pixel              background;
  XrtTblBorderSides      border_sides;
  XrtTblBorderType       border_type;
  Pixel              bottom_shadow;
  int            char_height;
  int            char_width;
  Pixmap             clip_pixmap;
  XrtTblDataType         datatype;
  Boolean            editable;
  ArgList            field_arglist;
  Pixel              foreground;
  int            maxlength;
  Boolean            multiline;
  int            pixel_height;
  int            pixel_width;
  Pixmap             pixmap;
  XrtTblPixmapLayout pixmap_layout;
  Boolean            pixmap_shadow;
  String             ps_fontlist;
      Boolean            sensitive;
      XrtTblCase         string_case;
      Pixel              top_shadow;
      XtPointer      userdata;
  Boolean            widget_shadow;
  Boolean            wordwrap;
} XrtTblContextValue;
```

XrtTblContextValueCallbackStruct

XmNxrtTblContextValueCallbackリストのコールバックに渡される情報を定義する構造体です。

```
typedef struct {
  int            reason;
  int            row, column;
  XEvent             *event;
  XrtTblAction       action;
  XrtTblContextReason    context_reason;
  XrtTblContextValue     value;
} XrtTblContextValueCallbackStruct;
```

XrtTblCreateWidgetCallbackStruct

XmNxrtTblCreateWidgetCallbackリストのコールバックに渡される情報を定義する構造体です。

```
typedef struct {
  XrtTblReason  reason;
  XEvent        *event;
  int       row, column;
  Widget        parent;
  WidgetClass   widget_class;
  String        class_name, name;
  Widget        clone_source;
  Widget        new_widget;
} XrtTblCreateWidgetCallbackStruct;
```

XrtTblDataType

XmNxrtTblDataTypeContextとXmNxrtTblDataTypeSeriesに使用され、セルのデータ型を指定する列挙法です。

```
XRTTBL_TYPE_FIELD_CURRENCY = 0,    /* XrtCurrencyField */
XRTTBL_TYPE_FIELD_DATE,            /* XrtDateField */
XRTTBL_TYPE_FIELD_FLOAT,           /* XrtFloatField */
XRTTBL_TYPE_FIELD_INT,             /* XrtIntField */
XRTTBL_TYPE_FIELD_NONE,
XRTTBL_TYPE_FIELD_STRING,          /* XrtStringField */
XRTTBL_TYPE_FIELD_COMBO_CURRENCY,  /* XrtComboBox with ... */
XRTTBL_TYPE_FIELD_COMBO_DATE,
XRTTBL_TYPE_FIELD_COMBO_FLOAT,
XRTTBL_TYPE_FIELD_COMBO_INT,
XRTTBL_TYPE_FIELD_COMBO_STRING,
XRTTBL_TYPE_FIELD_SPIN_CURRENCY,   /* XrtSpinBox with ... */
XRTTBL_TYPE_FIELD_SPIN_DATE,
XRTTBL_TYPE_FIELD_SPIN_FLOAT,
XRTTBL_TYPE_FIELD_SPIN_INT,
XRTTBL_TYPE_FIELD_SPIN_STRING,
XRTTBL_TYPE_BOOLEAN,               /* Native table types */
XRTTBL_TYPE_DOUBLE,
XRTTBL_TYPE_FLOAT,
XRTTBL_TYPE_INT,
XRTTBL_TYPE_STRING,
XRTTBL_TYPE_LAST = 100             /* User defined after this */
```

XrtTblDataTypeRoutine

XmNxrtTblDataTypeListに使用され、データ型の検証ルーチンを定義する構造体です。

```
typedef struct {
  XrtTblDataType        type;
  XtCallbackProc        routine;
} XrtTblDataTypeRoutine;
```

XrtTblDisplayWidgetCallbackStruct

XmNxrtTblDisplayWidgetCallbackリストのコールバックに渡される情報を定義する構造体です。

```
typedef struct {
  XrtTblReason reason;
  XEvent       *event;
  int      row, column;
  Widget       clone_source;
  Widget       widget;
} XrtTblDisplayWidgetCallbackStruct;
```

XrtTblDispScrollBar

XmNxrtTblDisplayHorizScrollBarとXmNxrtTblDisplayVertScrollBarに使用され、いつスクロールバーを表示するか指定する列挙法です。

```
XRTTBL_DISPSB_AS_NEEDED
XRTTBL_DISPSB_ALWAYS
XRTTBL_DISPSB_NEVER
```

XrtTblDoubleBuffer

XmNxrtTblDoubleBufferに使用され、ダブル・バッファリングをコントロールする列挙法です。

```
XRTTBL_DOUBLEBUFFER_NONE
XRTTBL_DOUBLEBUFFER_CELL
XRTTBL_DOUBLEBUFFER_WINDOW
```

XrtTblDrawPSarg

*XrtTblDrawPS()*と*XrtTblVaDrawPS()*に使用され、引数を指定する列挙法です。

```
XRTTBL_PS_ALL
XRTTBL_PS_AS_IS
XRTTBL_PS_AUTO
XRTTBL_PS_BEST
XRTTBL_PS_CELL_RANGE
XRTTBL_PS_CM
XRTTBL_PS_COLOR
XRTTBL_PS_COL_PAGEBREAKS
XRTTBL_PS_EDGE_PAGES
XRTTBL_PS_GRID_TYPE
XRTTBL_PS_INCHES
XRTTBL_PS_LANDSCAPE
XRTTBL_PS_LINE
XRTTBL_PS_MARGIN_BOTTOM
XRTTBL_PS_MARGIN_LEFT
XRTTBL_PS_MARGIN_RIGHT
XRTTBL_PS_MARGIN_TOP
XRTTBL_PS_MONO
XRTTBL_PS_NONE
XRTTBL_PS_ORIENTATION
XRTTBL_PS_PAPERSIZE_HEIGHT
XRTTBL_PS_PAPERSIZE_WIDTH
XRTTBL_PS_PORTRAIT
XRTTBL_PS_ROW_PAGEBREAKS
XRTTBL_PS_SCALE
XRTTBL_PS_SHADOW
XRTTBL_PS_SHOW_COL_LABELS
XRTTBL_PS_SHOW_FROZEN_CELLS
XRTTBL_PS_SHOWPAGE
XRTTBL_PS_SHOW_ROW_LABELS
XRTTBL_PS_UNITS
XRTTBL_PS_SHOW_FROZEN_ROWS
XRTTBL_PS_SHOW_FROZEN_COLS
XRTTBL_PS_SHOW_PIXMAPS
XRTTBL_PS_SHOW_TITLES
XRTTBL_PS_SHOW_WIDGETS
XRTTBL_PS_NUM_COPIES
XRTTBL_PS_FOOTER
XRTTBL_PS_FOOTER_FONT
XRTTBL_PS_FOOTER_MARGIN
XRTTBL_PS_FOOTER_POINT_SIZE
XRTTBL_PS_HEADER
XRTTBL_PS_HEADER_FONT
XRTTBL_PS_HEADER_MARGIN
XRTTBL_PS_HEADER_POINT_SIZE
```

XrtTblEnterCellCallbackStruct

XmNxrtTblEnterCellCallbackリストのコールバックに渡される情報を定義する構造体です。

```
typedef struct {
  XrtTblReason reason;
  XEvent       *event;
  String       *params;
  Cardinal     num_params;
  int      row, column;
  Boolean      doit;
} XrtTblEnterCellCallbackStruct;
```

XrtTblExportCallbackStruct

XmNxrtTblExportCallbackリストのコールバックに渡される情報を定義する構造体です。

```
typedef struct {
  XrtTblReason reason;
  XEvent       *event;
  int      row, column;
  String       target;
  Boolean      internal;
  XrtTblRange  range;
  unsigned charoperation;
  Boolean      doit;
  Boolean      bell;
} XrtTblExportCallbackStruct;
```

XrtTblExposeCallbackStruct

XmNxrtTblExposeCallbackリストのコールバックに渡される情報を定義する構造体です。

```
typedef struct {
  XrtTblReason reason;
  XEvent       *event;
  XrtTblRange  range;
} XrtTblExposeCallbackStruct;
```

XrtTblFontListValue

フォント・リストの値を指定するseriesのメンバです。

```
typedef struct {
  XrtTblContext         context;
  struct {
      XmFontList   fontList;
      Dimension    width, height;
               XmStringCharSet *tags;
               XmFontListEntry *entries;
               int      *num_entries;
          } value;
      } XrtTblFontListValue;
```

XrtTblImportCallbackStruct

XmNxrtTblImportCallbackリストのコールバックに渡される情報を定義する構造体です。

```
typedef struct {
  XrtTblReason reason;
  XEvent       *event;
  int      row, column;
  String       target;
  Boolean      internal;
  unsigned charoperation;
  unsigned chardropSiteStatus;
  Boolean      insert;
  Boolean      doit;
  Boolean      bell;
} XrtTblImportCallbackStruct;
```

XrtTblImportExportStruct

XmNxrtTblExportTargetsとXmNxrtTblImportTargetsに使用され、ドラッグ&ドロップのターゲットと変換プログラムを指定する構造体です。

```
typedef struct {
  String                target;
  XrtTblImportExportConvertProc converter;
} XrtTblImportExportStruct;
```

XrtTblIntValue

整数値を指定するseriesのメンバです。

```
typedef struct {
  XrtTblContextcontext;
  int      value;
} XrtTblIntValue;
```

XrtTblJumpScroll

XmNxrtTblJumpScrollに使用され、スクロールの方向を指定する列挙型です。

```
XRTTBL_JUMP_NONE
XRTTBL_JUMP_HORIZONTAL
XRTTBL_JUMP_VERTICAL
XRTTBL_JUMP_ALL
```

XrtTblLabelCreateCallbackStruct

XmNxrtTblLabelCreateCallbackリストのコールバックに渡される情報を定義する構造体です。

```
typedef struct {
  XrtTblReason reason;
  XEvent       *event;
  int      row, column;
  XmString     string;
  Boolean      store;
  Boolean      free_string;
} XrtTblLabelCreateCallbackStruct;
```

XrtTblLocation

XmNxrtTblLocationに使用され、タイトルの位置を指定する列挙法です。

```
XRTTBL_LOCATION_TOP
XRTTBL_LOCATION_LEFT
XRTTBL_LOCATION_BOTTOM
XRTTBL_LOCATION_RIGHT
XRTTBL_LOCATION_NONE
```

XrtTblMode

XmNxrtTblModeに使用され、表示モードを指定する列挙法です。

```
XRTTBL_MODE_LIST
XRTTBL_MODE_TABLE
```

XrtTblOverflow

XmNxrtTblOverflowContext と XmNxrtTblOverflowSeries に使用され、セルの overflow(隣のセルにまたがる表示)を指定する列挙法です。

```
XRTTBL_OVERFLOW_NONE
XRTTBL_OVERFLOW_HORIZONTAL
```

XrtTblPickListValue

PickList値を指定するseriesのメンバです。

```
typedef struct {
 XrtTblContextcontext;
      union {
      struct tm **d;
      double   *f;
      double   *c;
      int *i;
      String   *s;
 } value;
 XrtTblDataType type;
} XrtTblPickListValue;
```

XrtTblPixmapLayout

XmNxrtTblPixmapLayoutContextとXmNxrtTblPixmapLayoutSeriesに使用され、セルのpixmapの位置を指定する列挙法です。

```
XRTTBL_PIXMAP_TOPLEFT
XRTTBL_PIXMAP_TOPCENTER
XRTTBL_PIXMAP_TOPRIGHT
XRTTBL_PIXMAP_MIDDLELEFT
XRTTBL_PIXMAP_MIDDLECENTER
XRTTBL_PIXMAP_MIDDLERIGHT
XRTTBL_PIXMAP_BOTTOMLEFT
XRTTBL_PIXMAP_BOTTOMCENTER
XRTTBL_PIXMAP_BOTTOMRIGHT
XRTTBL_PIXMAP_TILE
```

XrtTblPixmapValue

pixmap値を指定するseriesのメンバです。

```
typedef struct {
  XrtTblContext    context;
  int      width, height;
  int      depth;
  Pixmap       value;
} XrtTblPixmapValue;
```

XrtTblPointerValue

ポインタ値を指定するseriesのメンバです。

```
typedef struct {
  XrtTblContextcontext;
  XtArgList    value;
}XrtTblPointerValue;
```

XrtTblPreviewScroll

プレビュー・スクロール・ウィンドウの定義に使用される列挙法です。

```
XRTTBL_PREVIEW_NONE
XRTTBL_PREVIEW_VERTICAL
XRTTBL_PREVIEW_HORIZONTAL
XRTTBL_PREVIEW_BOTH
```

XrtTblPrintCallbackStruct

XmNxrtTblPrintCallbackリストのコールバックに渡される情報を定義する構造体です。

```
typedef struct {
  int      reason;
  XEvent       *event;
  XrtTblRange  **range;
  int      current_page;
  int      total_pages;
  int      across_pages;
  int      down_pages;
  int      page_number;
  String       header, footer;
  FILE     *fp;
  Boolean      write_file_header;
  Boolean      write_file_trailer;
  Boolean      doit;
} XrtTblPrintCallbackStruct;
```

XrtTblPrintWidgetCallbackStruct

XmNxrtTblPrintWidgetCallbackリストのコールバックに渡される情報を定義する構造体です。

```
typedef struct {
  int      reason;
  XEvent       *event;
  Widget       widget;
  int      row, column;
  FILE     *fp;
  int      current_page;
  int      page_number;
  Boolean      doit;
} XrtTblPrintWidgetCallbackStruct;
```

XrtTblPSarg

*XrtTblDrawPS()*の引数の定義に使用される構造体です。

```
typedef struct {
  XrtTblDrawPSarg       name;
  XtArgVal          value;
} XrtTblPSarg;
```

XrtTblPSFontMap

XmNxrtTblPSFontMapListに使用され、Xフォントに対応するPostScriptフォントを定義する構造体です。

```
typedef struct {
  String       font_name, tag;
  String       PS_font;
  double       point_size;
} XrtTblPSFontMap;
```

XrtTblRange

セル範囲の定義に使用される構造体です。

```
typedef struct {
  int  start_row, start_column;
  int  end_row, end_column;
} XrtTblRange;
```

XrtTblReason

コールバックのreasonの定義に使用される列挙法です。

```
XRTTBL_REASON_CELLVALUE
XRTTBL_REASON_CONTEXTVALUE
XRTTBL_REASON_CREATE_WIDGET_BEGIN
XRTTBL_REASON_CREATE_WIDGET_END
XRTTBL_REASON_CURSOR_EDGE
XRTTBL_REASON_CURSOR_INSIDE
XRTTBL_REASON_CURSOR_LEAVE
XRTTBL_REASON_DISPLAY_WIDGET
XRTTBL_REASON_ENTER_CELL_BEGIN
XRTTBL_REASON_ENTER_CELL_END
XRTTBL_REASON_EXPORT_DRAG_BEGIN
XRRTBL_REASON_EXPORT_TARGET
XRTTBL_REASON_EXPORT_DRAG_END
XRTTBL_REASON_EXPOSE_BEGIN
XRTTBL_REASON_EXPOSE_END
XRTTBL_REASON_IMPORT_DRAG_MOTION
XRTTBL_REASON_IMPORT_BEGIN
XRTTBL_REASON_IMPORT_TARGET
XRTTBL_REASON_IMPORT_DROP_END
XRTTBL_REASON_LABELCREATE
XRTTBL_REASON_PRINTWIDGET_BEGIN
XRTTBL_REASON_PRINTWIDGET_END
XRTTBL_REASON_PRINT_BEGIN
XRTTBL_REASON_PRINT_END
XRTTBL_REASON_RESIZE_BEGIN
XRTTBL_REASON_RESIZE_DONE
XRTTBL_REASON_RESIZE_END
XRTTBL_REASON_SCROLL_BEGIN
XRTTBL_REASON_SCROLL_END
XRTTBL_REASON_SELECT_BEGIN
XRTTBL_REASON_SELECT_END
XRTTBL_REASON_SELECT_EXTEND
XRTTBL_REASON_SETVALUE_BEGIN
XRTTBL_REASON_SETVALUE_END
XRTTBL_REASON_SORT
XRTTBL_REASON_TRAVERSE_CELL
XRTTBL_REASON_UNSELECT
XRTTBL_REASON_VALIDATE_CELL_BEGIN
XRTTBL_REASON_VALIDATE_CELL_END
```

XrtTblRepaint

XmNxrtTblRepaintに使用され、テーブルの再描画をコントロールする列挙法です。

```
XRTTBL_REPAINT_ON
XRTTBL_REPAINT_OFF
XRTTBL_REPAINT_NOCALC
```

XrtTblResize

XmNxrtTblAllowResizeとXmNxrtTblAllowCellResizeに使用され、リサイズをコントロールする列挙法です。

```
XRTTBL_RESIZE_NONE
XRTTBL_RESIZE_HORIZONTAL
XRTTBL_RESIZE_VERTICAL
XRTTBL_RESIZE_ALL
```

XrtTblResizeCallbackStruct

XmNxrtTblResizeCallbackリストのコールバックに渡される情報を定義する構造体です。

```
typedef struct {
  XrtTblReason reason;
  XEvent       *event;
  Dimension    width, height;
  Dimension    visible_width;
  Dimension    visible_height;
} XrtTblResizeCallbackStruct;
```

XrtTblResizeCellCallbackStruct

XmNxrtTblResizeCellCallbackリストのコールバックに渡される情報を定義する構造体です。

```
typedef struct {
  XrtTblReason reason;
  XEvent       *event;
  String       *params;
  Cardinal     num_params;
  int      row, column;
  int      current_row_height,
  int      current_column_width,
  int      new_row_height,
  int      new_column_width,
  Boolean      doit;
} XrtTblResizeCellCallbackStruct;
```

XrtTblRowPlacement

XmNxrtTblFrozenRowPlacement 、 XmNxrtTblColumnLabelPlacement 、 XmNxrtTblHorizScrollBarPlacementに使用され、テーブルの横方向の領域の場所を指定する列挙法です。

```
XRTTBL_PLACE_TOP
XRTTBL_PLACE_BOTTOM
```

XrtTblSBPosition

XmNxrtTblHorizScrollBarPosition、XmNxrtTblVertScrollBarPositionに使用され、スクロールバーがどのようにテーブルに接続されるかを指定する列挙法です。

```
XRTTBL_SBPOSITION_CELLS
XRTTBL_SBPOSITION_SIDE
```

XrtTblScrollCallbackStruct

XmNxrtTblScrollCallbackリストのコールバックに渡される情報を定義する構造体です。

```
typedef struct {
  int          reason;
  XEvent           *event;
  unsigned char    direction;
  int          new_value;
  String           preview_text;
  XmScrollBarCallbackStruct *sb_cbs;
} XrtTblScrollCallbackStruct;
```

XrtTblSelectCallbackStruct

XmNxrtTblSelectCallbackリストのコールバックに渡される情報を定義する構造体です。

```
typedef struct {
  XrtTblReason reason;
  XEvent       *event;
  String       *params;
  Cardinal     num_params;
  int      range_index;
  XrtTblRange  *range;
  int      row, column;
  int      click_count;
  Boolean      doit;
} XrtTblSelectCallbackStruct;
```

XrtTblSelectionPolicy

XmNxrtTblSelectionPolicyに使用され、選択の決まり(policy)を定義する列挙法です。

```
XRTTBL_SELECT_NONE
XRTTBL_SELECT_SINGLE
XRTTBL_SELECT_RANGE
XRTTBL_SELECT_MULTIRANGE
```

XrtTblSeries

XrtTblAnyValueインスタンスのポインタのリストを定義する構造体です。

```
typedef struct {
  XrtTblValueType  type;
  String       name;
  int      last_index;
  int      length;
  XrtTblAnyValue   **list;
} XrtTblSeries;
```

XrtTblSortCallbackStruct

XmNxrtTblSortCallbackリストのコールバックに渡される情報を定義する構造体です。

```
typedef struct {
 int          reason;
 XEvent           *event;
 int          column;
 XrtTblCompareStruct   *data;
 int          num_rows;
} XrtTblSortCallbackStruct;
```

XrtTblStringValue

String値を指定するseriesのメンバです。

```
typedef struct {
      XrtTblContext    context;
      String       value;
} XrtTblStringValue;
```

XrtTblTrackCursorCallbackStruct

XmNxrtTblTrackCursorCallbackリストのコールバックに渡される情報を定義する構造体です。

```
typedef struct {
 XrtTblReason     reason;
 XEvent           *event;
 int          row, column;
 XrtTblBorderSides sides;
 Boolean          enter_cell;
 Cursor           old_cursor;
 Cursor           new_cursor;
} XrtTblTrackCursorCallbackStruct;
```

XrtTblTraverseCellCallbackStruct

XmNxrtTblTraverseCellCallbackリストのコールバックに渡される情報を定義する構造体です。

```
typedef struct {
 XrtTblReason reason;
 XEvent       *event;
 String       *params;
 Cardinal     num_params;
 int      row, column;
 int      next_row;
 int      next_column;
} XrtTblTraverseCellCallbackStruct;
```

XrtTblValidateCellCallbackStruct

XmNxrtTblValidateCellCallbackリストのコールバックに渡される情報を定義する構造体です。

```
typedef struct {
  XrtTblReason reason;
  XEvent       *event;
  int          row, column;
  Boolean      value_changed;
  XrtTblDataType   datatype;
  XtArgVal     *converted_value;
  XtPointer    user_data;
  String       value;
  Boolean      doit;
  Boolean      bell;
} XrtTblValidateCellCallbackStruct;
```

XrtTblValidatePolicy

セル値の検証の決まり(policy)を定義する列挙法です。

```
XRTTBL_VALIDATE_USER_EDIT
XRTTBL_VALIDATE_SET
XRTTBL_VALIDATE_NEVER
XRTTBL_VALIDATE_ALWAYS
```

XrtTblValueType

*XrtTblSeries()*手続きの引数として使用される列挙法です。

```
XRTTBL_VALUE_ARGLIST = 1,
XRTTBL_VALUE_BOOLEAN,
XRTTBL_VALUE_COLOR,
XRTTBL_VALUE_CURRENCY_PICKLIST,
XRTTBL_VALUE_DATE_PICKLIST,
XRTTBL_VALUE_FONTLIST,
XRTTBL_VALUE_FLOAT_PICKLIST,
XRTTBL_VALUE_INT_PICKLIST,
XRTTBL_VALUE_INT,
XRTTBL_VALUE_PIXMAP,
XRTTBL_VALUE_POINTER,
XRTTBL_VALUE_STRING,
XRTTBL_VALUE_STRING_PICKLIST,
XRTTBL_VALUE_WIDGET
XRTTBL_VALUE_ARGLIST = 1,
XRTTBL_VALUE_BOOLEAN,
XRTTBL_VALUE_COLOR,
XRTTBL_VALUE_CURRENCY_PICKLIST,
XRTTBL_VALUE_DATE_PICKLIST,
XRTTBL_VALUE_FONTLIST,
XRTTBL_VALUE_FLOAT_PICKLIST,
XRTTBL_VALUE_INT_PICKLIST,
XRTTBL_VALUE_INT,
XRTTBL_VALUE_PIXMAP,
XRTTBL_VALUE_POINTER,
XRTTBL_VALUE_STRING,
XRTTBL_VALUE_STRING_PICKLIST,
XRTTBL_VALUE_WIDGET
```

XrtTblWidgetClass

XmNxrtTblWidgetClassListに使用され、ウィジェットのクラスを定義する構造体です。

```
typedef struct {
  String       name;
  WidgetClass  *widget_class;
} XrtTblWidgetClass;
```

XrtTblWidgetPos

ウィジェットの位置の記憶に使用される、XrtTblWidgetValueのサブ構造体です。

```
typedef struct {
  Widget       widget;
  int      row, column;
  Window       parent_window;
       Boolean      moved;
} XrtTblWidgetPos;
```

XrtTblWidgetSetValueCallbackStruct

XmNxrtTblWidgetSetValueCallbackリストのコールバックに渡される情報を定義する構造体です。

```
typedef struct {
  int      reason;
  XEvent       *event;
  int      row, column;
  Widget       widget;
  Arg      *arglist;
  int      num_arglist;
  Boolean      doit;
} XrtTblWidgetSetValueCallbackStruct;
```

XrtTblWidgetValue

ウィジェット値を指定するseriesのメンバです。

```
typedef struct {
  XrtTblContext         context;
  Window            parent_window;
  struct {
       Widget       widget;
       int      width, height;
       XrtTblWidgetPos   *list;
       int      num_list;
       String       resource_string;
  } value;
} XrtTblWidgetValue;
```

XrtTblXmStringGet

XmNxrtTblXmStringGetModeに使用され、どのようにXmStringリソースが取り出されるかを指定する列挙法です。

```
XRTTBL_GET_STRING
XRTTBL_GET_XMSTRING
```

XrtTblXmStringList

行ラベル、列ラベル、または1行分のセル値を含む構造体です。

```
typedef struct {
  int      length;
  XmString     *list;
} XrtTblXmStringList;
```

テーブルがStringsを必要としている場合は、それらをStringListに割り当てる前に、XmStringとしてcastしてください。

XrtTblXmStringSet

XmNxrtTblXmStringSetModeに使用され、どのようにXmStringリソースがセットされるかを指定する列挙法です。

```
XRTTBL_SET_EXAMINE
XRTTBL_SET_STRING
XRTTBL_SET_XMSTRING
```

XrtTblXmStringProc

XmNxrtTblCellValueFuncとXmNxrtTblLabelCreateFuncに使用される手続きのプロトタイプです。

```
typedef XmString (*XrtTblXmStringProc) ();
```

F

リソースの型とクラス

リソースの型とクラスのアルファベット順のリストです。

Name	Class	Type
XmNxrtTblAlignmentContext	XmCxrtTblAlignment	XmRxrtTblAlignment
XmNxrtTblAlignmentSeries	XmCxrtTblAlignmentSeries	XmRxrtTblAlignmentSeries
XmNxrtTblAllowCellResize	XmCxrtTblAllowCellResize	XmRxrtTblResize
XmNxrtTblAllowResize	XmCxrtTblAllowResize	XmRxrtTblResize
XmNxrtTblArgListContext	XmCxrtTblArgList	XmRxrtTblArgList
XmNxrtTblArgListSeries	XmCxrtTblArgListSeries	XmRxrtTblArgListSeries
XmNxrtTblBackgroundContext	XmCBackground	XmRString
XmNxrtTblBackgroundPixelContext	XmCBackground	XmRPixel
XmNxrtTblBackgroundSeries	XmCxrtTblColorSeries	XmRxrtTblColorSeries
XmNxrtTblBorderSidesContext	XmCxrtTblBorderSides	XmRxrtTblBorderSides
XmNxrtTblBorderSidesSeries	XmCxrtTblBorderSidesSeries	XmRxrtTblBorderSidesSeries
XmNxrtTblBorderTypeContext	XmCxrtTblBorderType	XmRxrtTblBorderType
XmNxrtTblBorderTypeSeries	XmCxrtTblBorderTypeSeries	XmRxrtTblBorderTypeSeries
XmNxrtTblBottomShadowContext	XmCBackground	XmRPixel
XmNxrtTblCaseContext	XmCxrtTblCase	XmRxrtTblCase
XmNxrtTblCaseSeries	XmCxrtTblCaseSeries	XmRxrtTblCaseSeries
XmNxrtTblCellValueCallback	XmCCallback	XmRCallback
XmNxrtTblCellValueContext	XmCxrtTblCellValues	XmRXmString
XmNxrtTblCellValueFunc	XmCCallback	XtRFunction

Name	Class	Type
XmNxrtTblCellValues	XmCxrtTblCellValues	XmRxrtTblCellValues
XmNxrtTblCharHeightContext	XmCxrtTblHeight	XmRxrtTblCellSize
XmNxrtTblCharHeightSeries	XmCxrtTblSizeSeries	XmRxrtTblCellSizeSeries
XmNxrtTblCharWidthContext	XmCxrtTblWidth	XmRxrtTblCellSize
XmNxrtTblCharWidthSeries	XmCxrtTblSizeSeries	XmRxrtTblCellSizeSeries
XmNxrtTblClipChildren	XmCReadOnly	XmRWidgetList
XmNxrtTblClipChildrenPersistent	XmCBoolean	XmRBoolean
XmNxrtTblClipPixmapContext	XmCPixmap	XmRxrtTblClipPixmap
XmNxrtTblClipPixmapSeries	XmCxrtTblPixmapSeries	XmRxrtTblClipPixmapSeries
XmNxrtTblColumnLabelOffset	XmCxrtTblOffset	XmRInt
XmNxrtTblColumnLabelPlacement	XmCxrtTblRowPlacement	XmRxrtTblRowPlacement
XmNxrtTblColumnLabels	XmCxrtTblXmStringList	XmRxrtTblXmStringList
XmNxrtTblContext	XmCxrtTblContext	XmRxrtTblContext
XmNxrtTblContextValueCallback	XmCCallback	XmRCallback
XmNxrtTblCreateWidgetCallback	XmCCallback	XmRCallback
XmNxrtTblCurrencyPickListContext	XmCxrtTblCurrencyPickList	XmRxrtTblCurrencyPickList
XmNxrtTblCurrencyPickListSeries	XmCxrtTblCurrencyPickListSeries	XmRxrtTblCurrencyPickListSeries
XmNxrtTblCursorCorner	XmCCursor	XmRCursor
XmNxrtTblCursorHorizontal	XmCCursor	XmRCursor
XmNxrtTblCursorNontraversable	XmCCursor	XmRCursor
XmNxrtTblCursorTraversable	XmCCursor	XmRCursor
XmNxrtTblCursorVertical	XmCCursor	XmRCursor
XmNxrtTblDataTypeContext	XmCxrtTblDataType	XmRxrtTblDataType
XmNxrtTblDataTypeList	XmCxrtTblDataTypeList	XmRxrtTblDataTypeList
XmNxrtTblDataTypeSeries	XmCxrtTblDataTypeSeries	XmRxrtTblDataTypeSeries
XmNxrtTblDatePickListContext	XmCxrtTblDatePickList	XmRxrtTblDatePickList
XmNxrtTblDatePickListSeries	XmCxrtTblDatePickListSeries	XmRxrtTblDatePickListSeries
XmNxrtTblDebug	XmCxrtTblDebug	XmRBoolean
XmNxrtTblDisplayClipArrows	XmCxrtTblDisplayClipArrows	XmRBoolean
XmNxrtTblDisplayHorizScrollBar	XmCxrtTblDispScrollBar	XmRxrtTblDispScrollBar
XmNxrtTblDisplayVertScrollBar	XmCxrtTblDispScrollBar	XmRxrtTblDispScrollBar
XmNxrtTblDisplayWidgetCallback	XmCCallback	XmRCallback

Name	Class	Type
XmNxrtTblDoubleBuffer	XmCxrtTblDoubleBuffer	XmRxrtTblDoubleBuffer
XmNxrtTblEditableContext	XmCxrtTblEditable	XmRBoolean
XmNxrtTblEditableSeries	XmCxrtTblEditabilitySeries	XmRxrtTblBooleanSeries
XmNxrtTblEnterCellCallback	XmCCallback	XmRCallback
XmNxrtTblExportCallback	XmCCallback	XmRCallback
XmNxrtTblExportDelimiter	XmCxrtTblDelimiter	XmRString
XmNxrtTblExportTargets	XmCxrtTblTargets	XmRxrtTblTargets
XmNxrtTblExposeCallback	XmCCallback	XmRCallback
XmNxrtTblFieldArgListContext	XmCxrtTblArgList	XmRxrtTblArgList
XmNxrtTblFieldArgListSeries	XmCxrtTblFieldArgListSeries	XmRxrtTblArgListSeries
XmNxrtTblFloatPickListContext	XmCxrtTblFloatPickList	XmRxrtTblFloatPickList
XmNxrtTblFloatPickListSeries	XmCxrtTblFloatPickListSeries	XmRxrtTblFloatPickListSeries
XmNxrtTblFontListContext	XmCFontList	XmRFontList
XmNxrtTblFontListSeries	XmCxrtTblFontListSeries	XmRxrtTblFontListSeries
XmNxrtTblForegroundContext	XmCForeground	XmRString
XmNxrtTblForegroundPixelContext	XmCForeground	XmRPixel
XmNxrtTblForegroundSeries	XmCxrtTblColorSeries	XmRxrtTblColorSeries
XmNxrtTblFrameBorderType	XmCxrtTblBorderType	XmRxrtTblBorderType
XmNxrtTblFrameShadowThickness	XmCShadowThickness	XmRDimension
XmNxrtTblFrozenColumnPlacement	XmCxrtTblColumnPlacement	XmRxrtTblColumnPlacement
XmNxrtTblFrozenColumns	XmCxrtTblFrozenColumns	XmRInt
XmNxrtTblFrozenRowPlacement	XmCxrtTblRowPlacement	XmRxrtTblRowPlacement
XmNxrtTblFrozenRows	XmCxrtTblFrozenRows	XmRInt
XmNxrtTblHorizontalScrollBar	XmCHorizontalScrollBar	XmRWidget
XmNxrtTblHorizScrollBarLeftAttachment	XmCxrtTblScrollBarAttachment	XmRxrtTblAttachment
XmNxrtTblHorizScrollBarPlacement	XmCxrtTblRowPlacement	XmRxrtTblRowPlacement
XmNxrtTblHorizScrollBarPosition	XmCxrtTblScrollBarPosition	XmRxrtTblScrollBarPosition
XmNxrtTblHorizScrollBarRightAttachment	XmCxrtTblScrollBarAttachment	XmRxrtTblAttachment
XmNxrtTblImportCallback	XmCCallback	XmRCallback
XmNxrtTblImportDelimiter	XmCxrtTblDelimiter	XmRString
XmNxrtTblImportTargets	XmCxrtTblTargets	XmRxrtTblTargets
XmNxrtTblIntPickListContext	XmCxrtTblIntPickList	XmRxrtTblIntPickList

Name	Class	Type
XmNxrtTblIntPickListSeries	XmCxrtTblIntPickListSeries	XmRxrtTblIntPickListSeries
XmNxrtTblJumpScroll	XmCxrtTblJumpScroll	XmRxrtTblJumpScroll
XmNxrtTblLabelContext	XmCLabel	XmRXmString
XmNxrtTblLabelCreateCallback	XmCCallback	XmRCallback
XmNxrtTblLabelCreateFunc	XmCCallback	XtRFunction
XmNxrtTblLeftColumn	XmCxrtTblLeftColumn	XmRInt
XmNxrtTblMarginHeight	XmCMarginHeight	XmRDimension
XmNxrtTblMarginWidth	XmCMarginWidth	XmRDimension
XmNxrtTblMaxLengthContext	XmCMaxLength	XmRxrtTblInt
XmNxrtTblMaxLengthSeries	XmCxrtTblMaxLengthSeries	XmRxrtTblIntSeries
XmNxrtTblMinCellVisibility	XmCxrtTblMinCellVisibility	XmRInt
XmNxrtTblMinFlexibleHeight	XmCxrtTblMinFlexibleHeight	XmRDimension
XmNxrtTblMinFlexibleWidth	XmCxrtTblMinFlexibleWidth	XmRDimension
XmNxrtTblMode	XmCxrtTblMode	XmRxrtTblMode
XmNxrtTblMultilineContext	XmCxrtTblMultiline	XmRBoolean
XmNxrtTblMultilineSeries	XmCxrtTblEditabilitySeries	XmRxrtTblBooleanSeries
XmNxrtTblNumColumns	XmCxrtTblNumColumns	XmRInt
XmNxrtTblNumRows	XmCxrtTblNumRows	XmRInt
XmNxrtTblOverflowContext	XmCxrtTblOverflow	XmRxrtTblOverflow
XmNxrtTblOverflowSeries	XmCxrtTblOverflowSeries	XmRxrtTblOverflowSeries
XmNxrtTblPixelHeightContext	XmCxrtTblHeight	XmRxrtTblCellSize
XmNxrtTblPixelHeightSeries	XmCxrtTblSizeSeries	XmRxrtTblCellSizeSeries
XmNxrtTblPixelWidthContext	XmCxrtTblWidth	XmRxrtTblCellSize
XmNxrtTblPixelWidthSeries	XmCxrtTblSizeSeries	XmRxrtTblCellSizeSeries
XmNxrtTblPixmapContext	XmCPixmap	XmRPixmap
XmNxrtTblPixmapLayoutContext	XmCxrtTblPixmapLayout	XmRxrtTblPixmapLayout
XmNxrtTblPixmapLayoutSeries	XmCxrtTblPixmapLayoutSeries	XmRxrtTblPixmapLayoutSeries
XmNxrtTblPixmapSeries	XmCxrtTblPixmapSeries	XmRxrtTblPixmapSeries
XmNxrtTblPixmapShadowContext	XmCxrtTblPixmapShadow	XmRBoolean
XmNxrtTblPixmapShadowSeries	XmCxrtTblPixmapShadowSeries	XmRxrtTblBooleanSeries
XmNxrtTblPreviewScrolling	XmCxrtTblPreviewScrolling	XmRxrtTblPreviewScrolling
XmNxrtTblPrintCallback	XmCCallback	XmRCallback

Name	Class	Type
XmNxrtTblPrintWidgetCallback	XmCCallback	XmRCallback
XmNxrtTblPSFontListContext	XmCString	XmRString
XmNxrtTblPSFontListSeries	XmCxrtTblStringSeries	XmRxrtTblStringSeries
XmNxrtTblPSFontMapList	XmCxrtTblPSFontMapList	XmRxrtTblPSFontMapList
XmNxrtTblRangeContext	XmCxrtTblRangeContext	XmRxrtTblRange
XmNxrtTblRepaint	XmCxrtTblRepaint	XmRxrtTblRepaint
XmNxrtTblRepeatBackgroundColors	XmCxrtTblStringTable	XmRStringTable
XmNxrtTblRepeatForegroundColors	XmCxrtTblStringTable	XmRStringTable
XmNxrtTblResizeCallback	XmCCallback	XmRCallback
XmNxrtTblResizeCellCallback	XmCCallback	XmRCallback
XmNxrtTblRowLabelOffset	XmCxrtTblOffset	XmRInt
XmNxrtTblRowLabelPlacement	XmCxrtTblColumnPlacement	XmRxrtTblColumnPlacement
XmNxrtTblRowLabels	XmCxrtTblXmStringList	XmRxrtTblXmStringList
XmNxrtTblScrollCallback	XmCCallback	XmRCallback
XmNxrtTblSelectCallback	XmCCallback	XmRCallback
XmNxrtTblSelectedBackground	XmCxrtTblSelectedBackground	XmRString
XmNxrtTblSelectedCellList	XmCxrtTblRangeList	XmRxrtTblRangeList
XmNxrtTblSelectedForeground	XmCxrtTblSelectedForeground	XmRString
XmNxrtTblSelectionPolicy	XmCSelectionPolicy	XmRxrtTblSelectionPolicy
XmNxrtTblSensitiveContext	XmCxrtTblSensitive	XmRBoolean
XmNxrtTblSensitiveSeries	XmCxrtTblEditabilitySeries	XmRxrtTblBooleanSeries
XmNxrtTblSetTextCursorPos	XmCxrtTblSetTextCursorPos	XmRBoolean
XmNxrtTblShadowThickness	XmCShadowThickness	XmRDimension
XmNxrtTblSortCallback	XmCCallback	XmRCallback
XmNxrtTblSpace	XmCSpace	XmRDimension
XmNxrtTblSpanList	XmCxrtTblRangeList	XmRxrtTblRangeList
XmNxrtTblStringPickListContext	XmCxrtTblStringPickList	XmRxrtTblStringPickList
XmNxrtTblStringPickListSeries	XmCxrtTblStringPickListSeries	XmRxrtTblStringPickListSeries
XmNxrtTblText	XmCxrtTblText	XmRWidget
XmNxrtTblTextInitSelect	XmCxrtTblTextInitSelect	XmRBoolean
XmNxrtTblTopRow	XmCxrtTblTopRow	XmRInt
XmNxrtTblTopShadowContext	XmCBackground	XmRPixel

Name	Class	Type
XmNxrtTblTrackCursor	XmCxrtTblTrackCursor	XmRBoolean
XmNxrtTblTrackCursorCallback	XmCCallback	XmRCallback
XmNxrtTblTraversableContext	XmCxrtTblTraversable	XmRBoolean
XmNxrtTblTraversableSeries	XmCxrtTblEditabilitySeries	XmRxrtTblBooleanSeries
XmNxrtTblTraverseCellCallback	XmCCallback	XmRCallback
XmNxrtTblUserDataContext	XmCUserData	XmRPointer
XmNxrtTblUserDataSeries	XmCUserData	XmRxrtTblPointerSeries
XmNxrtTblValidateCellCallback	XmCCallback	XmRCallback
XmNxrtTblValidatePolicy	XmCxrtTblValidatePolicy	XmRxrtTblValidatePolicy
XmNxrtTblVerticalScrollBar	XmCVerticalScrollBar	XmRWidget
XmNxrtTblVertScrollBarBottomAttachment	XmCxrtTblScrollBarAttachment	XmRxrtTblAttachment
XmNxrtTblVertScrollBarPlacement	XmCxrtTblColumnPlacement	XmRxrtTblColumnPlacement
XmNxrtTblVertScrollBarPosition	XmCxrtTblScrollBarPosition	XmRxrtTblScrollBarPosition
XmNxrtTblVertScrollBarTopAttachment	XmCxrtTblScrollBarAttachment	XmRxrtTblAttachment
XmNxrtTblVisibleColumns	XmCxrtTblVisibleColumns	XmRInt
XmNxrtTblVisibleRows	XmCxrtTblVisibleRows	XmRInt
XmNxrtTblWidgetClassList	XmCxrtTblWidgetClassList	XmRxrtTblWidgetClassList
XmNxrtTblWidgetContext	XmCxrtTblWidget	XmRxrtTblWidget
XmNxrtTblWidgetSeries	XmCxrtTblWidgetSeries	XmRxrtTblWidgetSeries
XmNxrtTblWidgetSetValueCallback	XmCCallback	XmRCallback
XmNxrtTblWidgetShadowContext	XmCxrtTblWidgetShadow	XmRBoolean
XmNxrtTblWidgetShadowSeries	XmCxrtTblWidgetShadowSeries	XmRxrtTblBooleanSeries
XmNxrtTblWordWrapContext	XmCxrtTblWordWrap	XmRBoolean
XmNxrtTblWordWrapSeries	XmCxrtTblWordWrapSeries	XmRxrtTblBooleanSeries
XmNxrtTblXmStringGetMode	XmCxrtTblXmStringGetMode	XmRxrtTblXmStringGetMode
XmNxrtTblXmStringSetMode	XmCxrtTblXmStringSetMode	XmRxrtTblXmStringSetMode

Constraint Resources		
Name	**Class**	**Type**
XmNxrtTblBottomOffset	XmCxrtTblOffset	XmRxrtTblInt
XmNxrtTblLeftOffset	XmCxrtTblOffset	XmRxrtTblInt
XmNxrtTblLocation	XmCxrtTblLocation	XmRxrtTblLocation
XmNxrtTblResizeToCell	XmCBoolean	XmRBoolean
XmNxrtTblRightOffset	XmCxrtTblOffset	XmRxrtTblInt
XmNxrtTblTopOffset	XmCxrtTblOffset	XmRxrtTblInt
XmNxrtTblWidgetLocation	XmCxrtTblContext	XmRxrtTblContext

G

リソース･ファイルの使用

構文

リソースファイルの例

XRT/tableのリソース変換プログラムが認識できる、リソース･ファイルの構文の一覧です。この章の最後には、サンプルのリソース･ファイルが提供されています。XRT/tableは、XtとMotifに定義されていないデータ型のリソース変換のみを定義します。

手続きを使用して値をセットするリソース以外は、全てリソース･ファイルでセットできます。テーブルの部分の属性を指定する時は、Seriesリソースを使用してください。

G.1 構文

コメント

(!)の後のテキストは無視されます。

継続

行の一番最後をバック･スラッシュ(\)にすると、次の行に継続します。

Boolean

大文字、小文字、大･小混合文字の"True"または"False"。

列挙型

列挙型を用いるリソースは、列挙された値に基づく一貫した命名規則を使用します。全てのリソースでは、先頭のXRTTBL_がありません。リソース値は、大文字、小文字または大小混合の文字で入力できます。例えば、次の各リソースの指定は、どれもXRTTBL_BORDER_ETCHED_OUTがXmNxrtTblBorderTypeに使用されるようにしています。

```
*.XtXrtTable.xrtTblBorderType: BORDER_ETCHED_OUT
*.XtXrtTable.xrtTblBorderType: border_etched_out
*.XtXrtTable.xrtTblBorderType: BorDer_Etched_out
```

Seriesリソース

seriesのリソース値は、スペースで区切られているメンバのリストとして指定できます。(各メンバは括弧に囲まれます。) メンバの内、最初の二つのアイテムは行と列(context)で、三番目はリソース値です。値の一部として左または右の括弧を指定する時は、二つのバック・スラッシュを先に置いてください(\\( )。行と列(context)のインデックスは次の通りです。

`<integer>`	行または列の番号
`label`	行または列ラベル
`all`	全ての行または列、ラベルも含む
`allcells`	行または列にある全てのセル
`<integer> - <integer>`	行または列の範囲
`<integer>, <integer>`	セル/ラベル範囲の開始/終了位置

下記は、seriesメンバの例です。

```
*.XtXrtTable.xrtTblForegroundSeries: (all all red) \
 (all label blue) (2 3 blue) (1-3 1 green) \
 (5,5 7,7 green)
```

Picklist Seriesリソース

Picklist seriesリソースは、上記に記述されている他のseriesリソースと同じフォーマットを使用します。この場合、フォーマットの最初に来る二つのエントリは行と列で、その他のエントリは、全てカンマで区切ってください。

下記は、picklist seriesリソースの例です。

```
*.XtXrtTable.xrtTblIntPickListSeries: \
   (0 0-2 1,3,5,7,9,11) \
   (0 3-5 2,4,6,8,10,12,14,16,18,20,24,26) \
   (2 ALLCELLS 2,4,6,8,10,12,14,16,18,20,24,26)
```

上記の例では、行0の列0、1、2にpicklist *1,3,5,7,9,11*、行0の列3、4、5にpicklist *2,4,6,8,10,12,14,16,18,20,24,26*、行2の(ラベル以外の)全ての列にpicklist *2,4,6,8,10,12,14,16,18,20,24,26*を割り当てています。整数のpicklistは標準の整数の他に、二進数(0b)と16進数(0x)をサポートしています。

```
*.XtXrtTable.xrtTblStringPickListSeries: \
   (1 ALLCELLS John,Paul,George,Ringo) \
   (4 ALLCELLS Here are,some more,resources,from a,resource,file) \
   (6 0-2 These,values,"have been"\\,"loaded", from a,resource,file)
```

上記の例では、行1の全てのセルにpicklist *John,Paul,George,Ringo*、行4の全てのセルに*Here are,some more,resources,from a,resource,file*、行6の列0、1、2に*These,values,"have been"*\\,*"loaded",from a,resource,file*が割り当てられています。また、最後のPickListの例では、エスケープ・カンマと引用符が含まれています。

```
*XtXrtTable.xrtTblCurrencyPickListSeries: \
   (3 ALLCELLS 0,10.0,11.0,12.0,13.0-12.231,+45.5543e3)

*XtXrtTable.xrtTblFloatPickListSeries: \
   (5 ALLCELLS 10.0,11.0,12.0,13.0,-100,2e5,+45.5543e3)
```

前ページの例では、浮動小数点と通貨のpicklistを作成しています。この場合、シンタックスとフォーマットは全く同じです。浮動小数点と通貨のpicklistでは、標準の浮動小数点と指数をサポートしています。

```
*XtXrtTable.xrtTblDatePickListSeries: \
      (7 ALLCELLS NOW, TODAY, 1999-01-01 01:01:01, \
              1999-Apr-01 01:01:01, 01:01:01, NOW)
```

上記のリストは、行7の全てのセルに日付のpicklistを作成しています。日付のpicklistは次のフォーマットをサポートしています。

```
YYYY-M-D h:m:s
YYYY-MMM-D h:m:s
YYYY-M-D
YYYY-MMM-D
h:m:s
TODAY
NOW
```

TODAYは現在の日付けでの深夜、NOWは現在の日時です。

セル値

XmNxrtTblCellValuesは、直接またはファイルを使用して指定できます。ファイル名を指定する場合、リソース変換プログラムはそのファイルを開き、*XrtTblReadAscii()*を呼び出します。ファイルの指定は、環境変数や、(そのファイルで列区切りを示す)区切り文字を含むことができます。例:

```
*.XtXrtTable.xrtTblCellValues: $MYDIR/values.dat |
```

リソース・ファイルに直接セル値を提供する場合、このリソースは、カンマで区切られた値のリスト(一つのリストが一つの行)として指定され、各リストは括弧で囲まれます。各行が同じ数の列の値を持つ必要はありません。カンマや括弧を値として指定する場合は、二つのバック・スラッシュ(\\ )を先に置いてください。値を省略する時は、間にスペースを入れないでカンマを二つ(,,)入力してください。複数行を指定する時は、newline(\n)を挿入します。例:

```
*.XtXrtTable.xrtTblCellValues: (a,b,c) \
  (e,f\\,and g) \
  (Our number:\n\\(416\\) 594-1026) \
```

ラベル

XmRxrtTblXmStringList型のリソースは、カンマで区切られた値として指定されます。スペースも値の一部とみなされます。カンマを値として指定する場合は、二つのバック・スラッシュ(\\,)を先に置いてください。値を省略する時は、間にスペースを入れないでカンマを二つ(,,)入力してください。複数行を指定する時は、newline(\n)を挿入します。例:

```
*.XtXrtTable.xrtTblRowLabels: first,second, \
  and\\,a third,fourth,,sixth
```

セル/ラベルのpixmap

XmNxrtTblPixmapSeriesはファイルを使用して指定されます。リソース変換

プログラムはファイルを開き、ファイルを読み込む手続きを呼び出します。pixmapはXBMまたはXPMフォーマットでなければなりません。指定には環境変数を含むことができます。contextにpixmapが現れないように指定するには、NONEまたはUNSPECIFIED_PIXMAPを使用してください。例:

```
*.XtXrtTable.xrtTblPixmapSeries: \
  (label all none) (allcells 0 $MYDIR/myimage.xpm) \
  (all label xlogo32)
```

セル/ラベルのウィジェット

XmNxrtTblWidgetSeriesは、ウィジェット・クラスとオプションのウィジェット名で指定されます。リソース値は、大文字、小文字、大小混合文字で入力できます。contextにウィジェットが現れないように指定するには、NONEを使用します。詳細は、6.1.2を参照してください。

```
*.XtXrtTable.xrtTblWidgetSeries: \
  (label all XmLabel) (1 3 XmPushbutton.my_button) \
  (all label none) (0 0 XtXrtTable.inner_table) \
```

罫線の場所

XmNxrtTblBorderSidesSeriesは、一つか複数の有効なXrtTblBorderSides列挙値で指定され、間にはスペースを入れずに(+)を挿入します。

```
*.XtXrtTable.xrtTblBorderSidesSeries: \
  (label all borderside_top+borderside_bottom) \
  (allcells allcells borderside_all)
```

Span(結合された)リストと選択されたセル・リスト

XmRxrtTblRangeList型のリソースは、(各範囲が括弧で囲まれる)範囲のリストとして指定されます。始めのアイテムは範囲の開始と終了の行、二番目のアイテムは開始と終了の列です。例: (0-1 2-5) または、範囲の左上のセルと右下のセルの位置で指定することもできます。例: (0,0 3,3) 範囲リストのメンバ間にはスペースを入れることができます。行または列にある全てのセルを参照させるには、ALLまたはALLCELLSを使用します。例:

```
*.XtXrtTable.xrtTblSelectedCellList: (1-5 6-9) \
  (10,13 12,20) (ALL 21)
*.XtXrtTable.xrtTblSpanList: (0,0 1,1) (allcells 25)
```

マウスポインタ

XmRCursor型のリソースは、カーソルのシンボル名で、接頭部のXC_を省略した形で指定されます。例:

```
*.XtXrtTable.xrtTblCursorNontraversable: left_ptr
```

背景と影の色

XmNxrtTblBackgroundSeriesの各seriesメンバは、context、背景色、(場合によっては)上部と下部の影の色で指定されます。上部または下部の影の色が省略されている時は、XRT/tableが自動的に色を計算します。上部の影の色は下部より先に指定され、それぞれの色はカンマで区切られます。

次の例では、背景色を"white"、上部の影の色を"grey90"、下部の影の色を"grey95"にセットしています。

```
*.XtXrtTable.xrtTblBackgroundSeries: \
 (1 3 white,grey90,grey95)
```

PostScriptフォント・マップのリスト

XmNxrtTblPSFontMapListは、カンマで区切られた値のリストで指定され、各リストは括弧で囲まれ、それぞれ各フォントの定義に対応します。

```
*.XtXrtTable.xrtTblPSFontMapList: \
 (-*-helvetica-medium-r-*-*-*-*-*-*-*-iso8859-1,,\
 Helvetica-Narrow, 0)\
 (-*-helvetica-bold-r-*-*-*-*-*-*-*-iso8859-1,,\
 Helvetica-Narrow-Bold, 0)\
 (-*-helvetica-medium-o-*-*-*-*-*-*-*-iso8859-1,,\
 Helvetica-Narrow-Oblique, 0)\
 (-*-helvetica-bold-o-*-*-*-*-*-*-*-iso8859-1,,\
 Helvetica-Narrow-BoldOblique, 0)
```

繰り返される色

XmNxrtTblRepeatBackgroundColorsとXmNxrtTblRepeatForegroundColorsは、カンマで区切られた色のリストで指定されます。

```
*.XtXrtTable.xrtTblRepeatBackgroundColors: \
 red,white,blue
```

XmString変換プログラムの設定

*XrtTblSetXmStringConverter()*は、XmRXmStringリソース変換プログラムをセットし、セルとラベルのXmString値を作成します。この手続きには、引数は渡されず、値も返しません。また、この手続きは、必ずテーブルが作成される前に呼び出して下さい。

XmRXmStringリソース変換プログラムは、次のシンタックスを使用しフォントを指定しています。

@font@	XmNfontListのフォントのタグ
@P@	前のフォントに戻る
\@	"@"を文字として扱う

例えば、下記のストリングがリソース・ファイルで指定され、

```
For a @bold@free evaluation@P@ of any product
```

boldフォントがフォント・リストで次のように定義されている場合は、

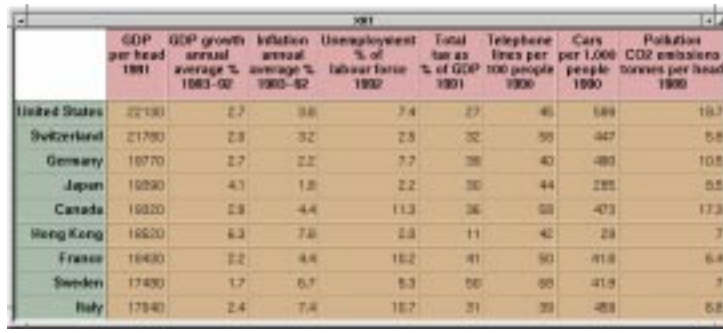
```
fixed,-*-times-bold-r-normal-*-140-*=bold
```

変換されたXmString値は、次のように表示されます。

For a **free evaluation** of any product

G.2 リソースファイルの例

次のテーブルは、下にあるリソース・ファイルで定義されています。



	GDP per head 1991	GDP growth annual average % 1983-92	Inflation annual average % 1983-92	Unemployment % of labour force 1992	Total tax as % of GDP 1991	Telephone lines per 100 people 1990	Cars per 1,000 people 1990	Pollution CO2 emissions tonnes per head 1989
United States	22130	2.7	3.6	7.4	27	46	589	19.7
Switzerland	21780	2.8	3.2	2.5	32	59	447	6.8
Germany	18070	2.7	2.2	7.7	38	40	480	10.5
Japan	19250	4.1	1.8	2.2	30	44	285	8.5
Canada	19320	2.8	4.4	11.3	36	58	473	17.3
Hong Kong	16520	6.3	7.8	2.3	11	42	28	3
France	18480	2.2	4.4	10.2	41	50	418	6.4
Sweden	17490	1.7	6.7	5.3	50	68	419	7
Italy	17840	2.4	7.4	10.7	31	39	488	6.8

```
! Sample resource file similar to $XRTHOME/src/table/samples/econ.res
*xrtTblAlignmentSeries:                (ALL ALL ALIGNMENT_TOPEND)\
(LABEL ALL ALIGNMENT_TOPCENTER)
*xrtTblAllowCellResize:                RESIZE_ALL
*xrtTblAllowResize:                    RESIZE_NONE
*xrtTblBackgroundSeries:               (ALL ALL tan)(LABEL ALL Yellow Green)\
(ALL LABEL Dark Green)(ALLCELLS ALLCELLS Light Steel Blue)
*xrtTblBackingStore:                   False
*xrtTblBorderTypeSeries:               (ALL ALL BORDER_IN)
*xrtTblCaseSeries:                     (ALL ALL CASE_AS_IS)
*xrtTblCellValues:                     $XRTHOME/src/table/data/economics.ascii
*xrtTblCharHeightSeries:               (ALL ALL 1)(LABEL ALL 4)
*xrtTblCharWidthSeries:                (ALL ALL 10)
*xrtTblClipChildrenPersistent:         False
*xrtTblColumnLabelOffset:              0
*xrtTblColumnLabelPlacement:           PLACE_TOP
*xrtTblColumnLabels:                   GDP\\nper head\\n1991,\
GDP growth\\nannual\\naverage %\\n1983-92,\
Inflation\\nannual\\naverage %\\n1983-92,\
Unemployment\\n% of\\nlabour force\\n1992,\
Total\\ntax as\\n% of GDP\\n1991,\
Telephone\\nlines per\\n100 people\\n1990,\
Cars\\nper 1\\,000\\npeople\\n1990,\
Pollution\\nCO2 emissions\\ntonnes per head\\n1989
*xrtTblDataTypeSeries:                 (ALL ALL TYPE_STRING)
*xrtTblDisplayClipArrows:              True
*xrtTblDisplayHorizScrollBar:          DISPSB_AS_NEEDED
*xrtTblDisplayVertScrollBar:           DISPSB_AS_NEEDED
*xrtTblDoubleBuffer:                   DOUBLEBUFFER_CELL
*xrtTblEditableSeries:                 (ALL ALL True)
*xrtTblExportDelimiter:
*xrtTblFontListSeries:\
(ALL ALL -Adobe-Helvetica-Medium-R-Normal--14-140-75-75-P-77-ISO8859-1)\
(LABEL ALL -Adobe-Helvetica-Bold-R-Normal--14-140-75-75-P-82-ISO8859-1)\
(ALL LABEL -Adobe-Helvetica-Bold-R-Normal--14-140-75-75-P-82-ISO8859-1)
*xrtTblForegroundSeries:               (ALL ALL black)(ALL LABEL Yellow)
```

```
*xrtTblFrameBorderType:               BORDER_NONE
*xrtTblFrameShadowThickness:          0
*xrtTblFrozenColumnPlacement:         PLACE_LEFT
*xrtTblFrozenColumns:                 0
*xrtTblFrozenRowPlacement:            PLACE_TOP
*xrtTblFrozenRows:                     0
*xrtTblHorizScrollBarLeftAttachment: ATTACH_CELLS
*xrtTblHorizScrollBarPlacement:       PLACE_BOTTOM
*xrtTblHorizScrollBarPosition:        SBPOSITION_CELLS
*xrtTblHorizScrollBarRightAttachment: ATTACH_CELLS
*xrtTblImportDelimiter:
*xrtTblJumpScroll:                    JUMP_NONE
*xrtTblLeftColumn:                    0
*xrtTblMarginHeight:                  4
*xrtTblMarginWidth:                   2
*xrtTblMaxLengthSeries:               (ALL ALL MAXINT)
*xrtTblMinCellVisibility:             100
*xrtTblMinFlexibleHeight:             5
*xrtTblMinFlexibleWidth:              5
*xrtTblMode:MODE_TABLE
*xrtTblMultilineSeries:               (ALL ALL False)
*xrtTblNumColumns:                    8
*xrtTblNumRows:                       9
*xrtTblOverflowSeries:                (ALL ALL OVERFLOW_NONE)
*xrtTblPSFontListSeries:              (ALL ALL null)
*xrtTblPixelHeightSeries:             (ALL ALL VARIABLE)
*xrtTblPixelWidthSeries:              (ALL ALL VARIABLE)
*xrtTblRepeatBackgroundColors:        Inherited
*xrtTblRepeatForegroundColors:        Inherited
*xrtTblRowLabelOffset:                0
*xrtTblRowLabelPlacement:             PLACE_LEFT
*xrtTblRowLabels:                     United States,Switzerland,Germany,Japan,\
Canada,Hong Kong,France,Sweden,Italy
*xrtTblSelectedBackground:            Black
*xrtTblSelectedForeground:            White
*xrtTblSelectionPolicy:               SELECT_NONE
*xrtTblSetTextCursorPos:              False
*xrtTblShadowThickness:               1
*xrtTblSpace:                         6
*xrtTblTextInitSelect:                False
*xrtTblTopRow:                        0
*xrtTblTrackCursor:                   True
*xrtTblTraversableSeries:             (ALL ALL False)
*xrtTblValidatePolicy:                VALIDATE_USER_EDIT
*xrtTblVertScrollBarBottomAttachment: ATTACH_CELLS
*xrtTblVertScrollBarPlacement:        PLACE_RIGHT
*xrtTblVertScrollBarPosition:         SBPOSITION_CELLS
*xrtTblVertScrollBarTopAttachment:    ATTACH_CELLS
*xrtTblVisibleColumns:                8
*xrtTblVisibleRows:                   9
*xrtTblWidgetSeries:                  (ALL ALL NONE)
*xrtTblWidgetShadowSeries:            (ALL ALL False)
*xrtTblWordWrapSeries:                (ALL ALL False)
*xrtTblXmStringGetMode:               GET_XMSTRING
*xrtTblXmStringSetMode:               SET_EXAMINE
```

H

UIL引数

XRT/tableのUIL引数とデータ型のアルファベット順のリストです。二番目の表は、XRT/tableのUILのreasonsです。UILは、6.14に解説されています。

名前の最後に"Context"が付く引数は、テーブル全体に適用されます。尚、UILは、標準のMotifウィジェットで使用されるデータ型のみをサポートしていますので、UILでPickListリソースを指定することはできません。

UIL Argument Name	Argument Type
XmNxrtTblAlignmentContext	integer
XmNxrtTblAllowCellResize	integer
XmNxrtTblAllowResize	integer
XmNxrtTblArgListContext	string
XmNxrtTblBackgroundContext	string
XmNxrtTblBackgroundPixelContext	color
XmNxrtTblBorderSidesContext	integer
XmNxrtTblBorderTypeContext	integer
XmNxrtTblBottomOffset	integer
XmNxrtTblBottomShadowContext	color
XmNxrtTblCaseContext	integer
XmNxrtTblCellValueContext	compound_string
XmNxrtTblCharHeightContext	integer
XmNxrtTblCharWidthContext	integer
XmNxrtTblClipChildrenPersistent	boolean
XmNxrtTblClipPixmapContext	icon

UIL Argument Name	Argument Type
XmNxrtTblColumnLabelOffset	integer
XmNxrtTblColumnLabelPlacement	integer
XmNxrtTblDataTypeContext	integer
XmNxrtTblDebug	boolean
XmNxrtTblDisplayClipArrows	boolean
XmNxrtTblDisplayHorizScrollBar	integer
XmNxrtTblDisplayVertScrollBar	integer
XmNxrtTblDoubleBuffer	integer
XmNxrtTblEditableContext	boolean
XmNxrtTblExportDelimiter	string
XmNxrtTblFieldArgListContext	string
XmNxrtTblFontListContext	font_table
XmNxrtTblForegroundContext	string
XmNxrtTblFrameBorderType	integer
XmNxrtTblFrameShadowThickness	integer
XmNxrtTblFrozenColumnPlacement	integer
XmNxrtTblFrozenColumns	integer
XmNxrtTblFrozenRowPlacement	integer
XmNxrtTblFrozenRows	integer
XmNxrtTblHorizScrollBarLeftAtt	integer
XmNxrtTblHorizScrollBarPlace	integer
XmNxrtTblHorizScrollBarPosition	integer
XmNxrtTblHorizScrollBarRightAtt	integer
XmNxrtTblImportDelimiter	string
XmNxrtTblJumpScroll	integer
XmNxrtTblLeftColumn	integer
XmNxrtTblLeftOffset	integer
XmNxrtTblLocation	integer
XmNxrtTblMarginHeight	integer
XmNxrtTblMarginWidth	integer
XmNxrtTblMaxLengthContext	integer
XmNxrtTblMinCellVisibility	integer

UIL Argument Name	Argument Type
XmNxrtTblMinFlexibleHeight	integer
XmNxrtTblMinFlexibleWidth	integer
XmNxrtTblMode	integer
XmNxrtTblMultilineContext	boolean
XmNxrtTblNumColumns	integer
XmNxrtTblNumRows	integer
XmNxrtTblOverflowContext	integer
XmNxrtTblPixelHeightContext	integer
XmNxrtTblPixelWidthContext	integer
XmNxrtTblPixmapContext	icon
XmNxrtTblPixmapLayoutContext	integer
XmNxrtTblPixmapShadowContext	boolean
XmNxrtTblPreviewScrolling	integer
XmNxrtTblPSFontListContext	string
XmNxrtTblRepaint	integer
XmNxrtTblRepeatBackgroundColors	asciz_string_table
XmNxrtTblRepeatForegroundColors	asciz_string_table
XmNxrtTblRightOffset	integer
XmNxrtTblRowLabelOffset	integer
XmNxrtTblRowLabelPlacement	integer
XmNxrtTblSelectedBackground	string
XmNxrtTblSelectedForeground	string
XmNxrtTblSelectionPolicy	integer
XmNxrtTblSensitiveContext	Boolean
XmNxrtTblSetTextCursorPos	Boolean
XmNxrtTblShadowThickness	Integer
XmNxrtTblSpace	Integer
XmNxrtTblTextInitSelect	Boolean
XmNxrtTblTopOffset	Integer
XmNxrtTblTopRow	Integer
XmNxrtTblTopShadowContext	Color
XmNxrtTblTrackCursor	Boolean
XmNxrtTblTraversableContext	Boolean
XmNxrtTblValidatePolicy	Integer

UIL Argument Name	Argument Type
XmNxrtTblVertScrollBarBottomAtt	Integer
XmNxrtTblVertScrollBarPlacement	Integer
XmNxrtTblVertScrollBarPosition	Integer
XmNxrtTblVertScrollBarTopAttach	Integer
XmNxrtTblVisibleColumns	Integer
XmNxrtTblVisibleRows	Integer
XmNxrtTblWidgetShadowContext	Boolean
XmNxrtTblWordWrapContext	Boolean
XmNxrtTblXmStringGetMode	Integer
XmNxrtTblXmStringSetMode	Integer

Reasons
XmNxrtTblCellValueCallback
XmNxrtTblContextValueCallback
XmNxrtTblCreateWidgetCallback
XmNxrtTblDisplayWidgetCallback
XmNxrtTblEnterCellCallback
XmNxrtTblExportCallback
XmNxrtTblExposeCallback
XmNxrtTblImportCallback
XmNxrtTblLabelCreateCallback
XmNxrtTblPrintCallback
XmNxrtTblPrintWidgetCallback
XmNxrtTblResizeCallback
XmNxrtTblResizeCellCallback
XmNxrtTblScrollCallback
XmNxrtTblSelectCallback
XmNxrtTblSortCallback

Reasons
XmNxrtTblTrackCursorCallback
XmNxrtTblTraverseCellCallback
XmNxrtTblValidateCellCallback
XmNxrtTblWidgetSetValueCallback

I

XrtLabelウィジェット

ウィジェットの概要 ■ XrtLabelについて
XrtLabelを使用する ■ XRT/tableと一緒に使用する
リソース ■ ソースの型とクラス

XrtLabelウィジェット[1] は、MotifのXmLabelウィジェットと似ているラベルです。XrtLabelは、テーブルのタイトル用に設計されていますので、アプリケーションの他の部分には使用しないでください。XrtLabelは、標準のラベルでは提供されていない、回転、影付き罫線、印刷(テーブルで使用された場合)等をサポートしています。

I.1 ウィジェットの概要

インクルード・ファイル	*<Xm/XrtLabel.h>*
クラス名	XtXrtLabel
クラス階層	Core → XmPrimitive → XtXrtLabel
クラス・ポインタ	xtXrtLabelWidgetClass
インスタンスの生成	`widget = XtCreateWidget(name, xtXrtLabelWidgetClass ...)`
関数/マクロ	*XmCreateXrtLabel(), IsXrtLabel()*

I.2 XrtLabelについて

XrtLabelはMotifに新しいウィジェットを追加します。このウィジェットは、Xtのコールで簡単に操作できます。

[1] 以前のバージョンでは、XrtLabelはXRT/labelという名前でした。

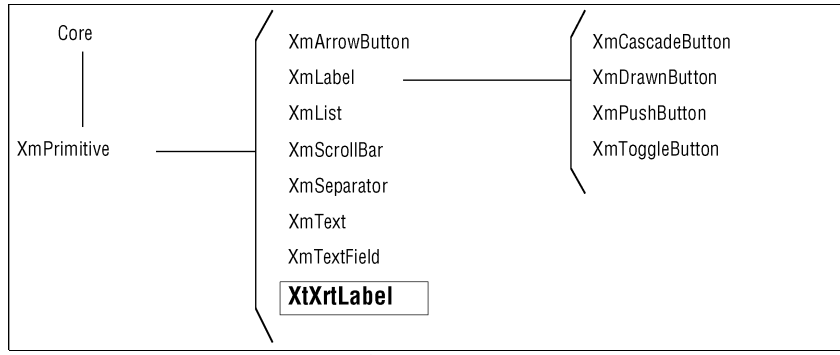


図65　Motifクラス階層内のXrtLabelの位置

このウィジェットは、標準のXmLabelのほとんどのリソースを持っています。図66は、このウィジェットの構成です。

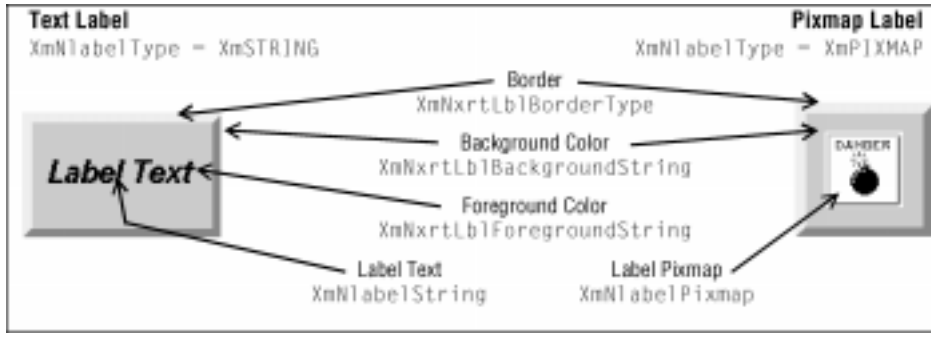


図66　XrtLabelウィジェットの構成

I.3　XrtLabelを使用する

ラベルの作成

ラベルを作成するには、*<Xm/XrtLabel.h>*をインクルードし、*XtVaCreateMangedWidget()*を呼び出します。(下の例参照)

```
Widget label;
label = XtVaCreateManagedWidget("label",
  xtXrtLabelWidgetClass, table,
  ...
  NULL);
```

ラベルの型

XrtLabelは図66にあるように、ラベル内にテキスト・ストリングまたはpixmapを表示することができます。`XmNlabelType`リソースは、ラベルの型を指定します。テキスト・ラベルを表示するには、ラベルの型を`XmSTRING`にセットしてください。pixmapを表示する場合は、`XmPIXMAP`にセットします。

テキスト・ラベルは、XmNlabelStringの値を表示し、pixmapラベルは、XmNlabelPixmapで指定されるpixmapを表示します。

テキスト・ラベル

ラベルに表示されるテキストは、XmNlabelStringリソースを使用して設定したり変更したりすることができます。Motif 1.2を使用している場合は、このリソースはStringまたはXmStringでセットできます。Motif 2.x使用している場合は、XmStringでセットしてください。XmNalignmentは、テキストの行揃えをコントロールします。次の例は、ラベルに新しいテキストをセットしています。

```
XtVaSetValues(label,
  XmNlabelString, "Label Text",
  XmNalignment, XmALIGNMENT_CENTER,
  NULL);
```

pixmapラベル

ラベルに表示されるpixmapは、XmNlabelPixmapリソースを使用して設定したり変更したりすることができます。このリソースは、pixmapを値として取ります。XrtLabelでは、XBMまたはXPMフォーマットがサポートされています。pixmapは使用後必ず解放してください。XRT/tableはセル/ラベル内のpixmapをコピーしません。pixmapはラベルのマージンの内側に表示され、XmNalignmentが位置をコントロールします。

CコードにXBM/XPMイメージを直接入れることもできますが、これらのイメージをファイルで保持し、必要時に読み込む方が簡単です。ファイルに保管されているイメージを指定するには、Motifのリソース変換プログラムを用いて、ファイル名のストリングをXmNlabelPixmapが必要とするデータ型のpixmapに変換してください。

```
XtVaSetValues(label,
  XmNlabelType, XmPIXMAP,
  XtVaTypedArg, XmNlabelPixmap,
  XmRString, "my_image.xpm",
  strlen("my_image.xpm") + 1,
  NULL);
```

ラベルからpixmapを取り去るには、XmNlabelPixmapをXmUNSPECIFIED_PIXMAPにセットしてください。

罫線の種類

XmNxrtLblBorderTypeはラベルの周囲に描かれる罫線の種類を指定します。使用できる罫線のスタイルは、図67にある通りです。罫線の幅は、XmNshadowThicknessを使用してセットします。

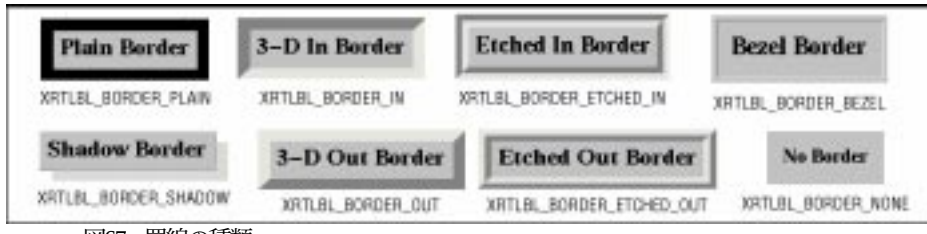


図67　罫線の種類

回転

ラベルの内容は、XmNxrtLblRotationリソースを使用して回転させることができます。テキストまたはpixmapを反時計方向に90度回転するには、このリソースをXRTLBL_ROTATE_90にセットします。XRTLBL_ROTATE_270にすると270度回転させることができます。



図68　ラベルの回転

色

XrtLabelには、継承されたXmNbackgroundとXmNforegroundリソースの他に、ラベルの背景/前景色を指定するもう一つの方法が提供されています。XmNxrtLblBackgroundStringとXmNxrtLblForegroundStringを用いると、テキスト・ストリングで色を指定することができます。これは、Xサーバのカラー・データベースの色(“LemonChiffon”等)や、RGB値(“#228B22”等)を指定する場合に便利です。尚、色の指定に関する詳細は、2.8を参照してください。

一つの色のリソースをセットすると、もう一方もアップデートします。例えば、XmNforegroundをpixel値にセットすると、XmNxrtLblForegroundStringのRGB値もその値にアップデートします。

高さと幅の設定

デフォルトでは、XrtLabelがタイトルとして使用されると、ラベルに表示されているテキストまたはpixmapの高さと幅に合わせてサイズがセットされます。高さと幅を別に設定する場合は、XmNrecomputeSizeをFalseにし、XmNheightとXmNwidthを使用して高さと幅をそれぞれセットしてください。

XmNrecomputeSizeがデフォルト値のTrueになっていると、XmNheightとXmNwidthはセットできません。

その他の属性

XrtLabelによって継承されるXmLabelのリソースは、XmLabelの場合と同様に動作します。例えば、ラベルに使用されるフォント・リストの設定にはXmNfontListが使用できます。Appendix I.5には、XrtLabelによって継承されるリソースのリストがあります。(これらのリソースの詳細は、Motifプログラミングに関する資料を参照してください。)

I.4 XRT/tableと一緒に使用する

XrtLabelは、XRT/tableのテーブルにタイトルを提供するために設計されています。ここでは、XRT/tableとの使用法を解説します。

タイトルの追加

テーブルにタイトルを追加するには、テーブルの子としてウィジェットを作成します。次の例では、ラベルをテーブルのタイトルとして作成しています。

```
title = XtVaCreateWidget("title",
  xtXrtLabelWidgetClass, table,
  XmNlabelString, "A Table Title",
  XmNxrtTblLocation, XRTTBL_LOCATION_TOP,
  NULL);
```

タイトルの位置

XRT/tableのconstraintリソースは、各タイトルの位置(とサイズ)をコントロールします。constraintリソースは、テーブルではなく、ラベル自身にセットしてください。テーブルへタイトルを追加する方法は、詳細が3.6に解説されています。

セル/ラベルへの配置

XrtLabelウィジェットは、テーブルのセルやラベル内に配置することもできます。このウィジェットは回転がサポートされているので、回転させたいセルがある時にXrtLabelを使用できます。セルやラベルにウィジェットを配置する方法は、6.1に解説されています。

次の例は、XrtLabelウィジェットを全ての列ラベルに追加しています。

```
label = XtVaCreateWidget("label",
  xtXrtLabelWidgetClass, table,
  XmNxrtTblWidgetLocation,
      XrtTblSetContext(XRTTBL_LABEL, XRTTBL_ALL),
  NULL);
```

印刷

XRT/tableのタイトルとして使用されている時、XRT/tableのPostScript出力ルーチン(*XrtTblDrawPS()*と*XrtTblVaDrawPS()*)が、PostScriptフォントとデバイスに依存しないPostScriptオペレータを使用してラベルを印刷します。

I.5 リソース

XrtLabelの全てのリソースがアルファベット順に一覧になっています。リソース名の後は、データ型、デフォルト値、そしてリソースと共に使用される手続き(**C**reate、**S**et、**G**et)のリストです。

XmNlabelPixmap　Pixmap　XmUNSPECIFIED_PIXMAP　CSG

継承された`XmNlabelPixmap`リソースを置き換え、リソース・ファイルでセットできるようにしたり、XPMフォーマットのpixmapをサポートするようにします。このリソースは、`XmNlabelType`が`XmPIXMAP`の時に表示されるpixmapを指定します。

XmNlabelString　XmString　NULL　CSG

継承された`XmNlabelString`リソースを置き換え、Stringとしてセットできるようにします(Motif 1.2を使用している場合)。このリソースは、`XmNlabelType`が`XmSTRING`の時に表示されるラベルのテキスト値を指定します。Stringとして指定される時は、XrtLabelは、内部でXmStringに変換します。

XmNshadowThickness　Dimension　5　CSG

Coreから継承されたリソースのデフォルト値(2)を置き換えます。

XmNxrtLblBackgroundString　String　“Inherited”　CSG

ラベルの背景色を指定します。このリソースは、値としてStringを取ることを除けば、Coreから継承された`XmNbackground`リソースと同様です。値は、Xカラー・データベースにある色名か、RGB値です。“Inherited”にセットされている時は、色は`XmNbackground`から継承されます。このリソースがセットされると、`XmNbackground`リソースが副作用でアップデートします。

モノクロのサーバでは、黒以外の色にセットすると白に表示されます。

XmNxrtLblBorderType　XrtLblBorderType　XRTLBL_BORDER_IN　CSG

ラベルの罫線に使用されるスタイルを指定します。有効な値は、`XRTLBL_BORDER_PLAIN`、`XRTLBL_BORDER_SHADOW`、`XRTLBL_BORDER_IN`、`XRTLBL_BORDER_OUT`、`XRTLBL_BORDER_ETCHED_IN`、`XRTLBL_BORDER_ETCHED_OUT`、`XRTLBL_BORDER_NONE`です。

罫線の幅は、`XmNshadowThickness`で指定します。

XmNxrtLblForegroundString　String　“Inherited”　CSG

ラベルの前景色を指定します。このリソースは、値としてStringを取ることを除けば、Coreから継承された`XmNforeground`リソースと同様です。値は、Xカラー・データベースにある色名か、RGB値です。“Inherited”にセットされている時は、色は`XmNforeground`から継承されます。このリソースがセットされると、`XmNforeground`リソースが副作用でアップデートします。

モノクロのサーバでは、白以外の色にセットすると黒に表示されます。

XmNxrtLblRotation	XrtLblRotation	XRTLBL_ROTATE_NONE	CSG

ラベルの内容(テキストまたはpixmap)の回転を指定します。有効な値は、XRTLBL_ROTATE_NONE、XRTLBL_ROTATE_90、XRTLBL_ROTATE_270です。回転の方向は、時計と反対方向です。

継承されたリソース

XrtLabelは、XmLabelとその親のクラスからリソースを継承します。次の表は、これらのリソースとそれを定義する親のクラスのリストです。詳細は、MotifやXtの資料をご覧ください。

XmLabel リソース

XmNalignment	XmNlabelString	XmNrecomputeSize
XmNfontList	XmNlabelType	XmNstringDirection
XmNlabelInsensitivePixmap	XmNmarginHeight	
XmNlabelPixmap	XmNmarginWidth	

I.6 リソースの型とクラス

Resource	Inherited From
XmNaccelerators	Core
XmNancestorSensitive	Core
XmNbackground	Core
XmNbackgroundPixmap	Core
XmNborderColor	Core
XmNborderPixmap	Core
XmNborderWidth	Core
XmNbottomShadowColor	Primitive
XmNbottomShadowPixmap	Primitive
XmNcolormap	Core
XmNdepth	Core
XmNdestroyCallback	Core
XmNforeground	Primitive
XmNheight	Core
XmNhelpCallback	Primitive
XmNhighlightColor	Primitive
XmNhighlightOnEnter	Primitive

Resource	Inherited From
XmNhighlightPixmap	Primitive
XmNhighlightThickness	Primitive
XmNinitialResourcesPersistent	Core
XmNmappedWhenManaged	Core
XmNnavigationType	Primitive
XmNscreen	Core
XmNsensitive	Core
XmNshadowThickness	Primitive
XmNtopShadowColor	Primitive
XmNtopShadowPixmap	Primitive
XmNtranslations	Core
XmNtraversalOn	Primitive
XmNuserData	Primitive
XmNwidth	Core
XmNx	Core
XmNy	Core

Name	Class	Type
XmNlabelPixmap	XmCLabelPixmap	XmRPixmap
XmNlabelString	XmCXmString	XmRXmString
XmNshadowThickness	XmCShadowThickness	XmRHorizontalDimension
XmNxrtLblBackgroundString	XmCBackground	XmRString
XmNxrtLblBorderType	XmCxrtLblBorderType	XmRxrtLblBorderType
XmNxrtLblForegroundString	XmCForeground	XmRString
XmNxrtLblRotation	XmCxrtLblRotation	XmRxrtLblRotation

J

サンプル・プログラム

例プログラム
デモ・プログラム
simple.c

XRT/tableには幾つかの例やデモのプログラムが提供されています。例は、特定な機能を使用した短いプログラムです。デモは、複数の機能を使用した、アプリケーションです。XRT/tableの基本的なプログラミングの大半が、これらの例やデモで紹介されています。

J.1 例プログラム

例プログラムは、*$XRTHOME/src/table/examples*にあります。

- *simple.c*は、簡単なテーブルを表示する、XRT/tableの基本操作習得に便利なプログラムです。第1章に詳細が解説されています。
- *styles*は、ボタンをクリックすることによりテーブルの表示を切り替える例を示しています。*styles_c.c*は全てのリソースをCコードでセットし、*styles_res.c*は全てのリソースをリソース・ファイル(*styles.res*)から読み込み、*styles_uil.c*はUILを使用して全てのリソースをセット(*styles_uil.uil*)しています。
- *multiply.c*は、`XmNxrtTblCellValueCallback`を使用し、要求に従って各セルの値を指定する例です。これは、大きなテーブルを扱う場合に便利です。
- *list.c*は、`XmNxrtTblMode`をセットし、XmScrolledListと同様な動作をするテーブルを作成しています。ソートや検索の例も示しています。
- *entry.c*は、TraverseCellとEnterCellコールバックを使用し、トラバースやテキストの入力をコントロールしています。セル内でボタンやオプション・メニューも使用しています。
- *simple_cpp.cxx*は、*simple.c*でC++を使用しています。

J.2 デモ・プログラム

デモ・アプリケーションは、*$XRTHOME/src/table/demos*にあります。各デモごとにディレクトリがあります。

- *widgets*は、セル内でウィジェットを管理する例を示しています。一つのウィンドウのテーブルは、“ドラッグ元”になるウィジェットをパレット形式に表示し、もう一つのウィンドウは空の“ドロップ先”のテーブルになっています。ドラッグ元のパレットからウィジェットをドラッグし、ドロップ先のセルにそのウィジェットを表示させることができます。
- *form*は、二つのテーブルを使用するデータ入力フォームです。一つのテーブルは従業員のリストを表示しており、セルをクリックすると、二つめのテーブルが現れ、各従業員のインフォメーションの詳細を入力、変更することができます。セルのスパン(結合)や、セル内のウィジェット、pixmapの例も示しています。
- *play*を利用して、XRT/tableの全てのリソースにアクセスすることができます。ユーザはどのリソース値も変更することができ、その結果を直ちに見ることができます。*play*は、テーブルの記述をリソース・ファイルに保管しますので、これをアプリケーションで使用することもできます。
- *bombhunt*は、XRT/tableを簡単なゲームで使用した例です。セルの下に隠されている地雷を探し当ててください。
- *stocks*では、色をランタイム時にセットしています。

複数のXRT製品を使用したデモ

- *graph_editor*は、一つのアプリケーションでXRT/tableとXRT/graphを両方使用しています。テーブルは、XRT/graphでグラフ化されているデータを表示します。テーブルのデータを編集すると、グラフはそれに応じてアップデートします。
- *mvc*-XRT/graph、XRT/table、XRT/3dを使用するMVCのデモです。
- *spreadsheet*は、スプレッドシート(表計算)形式のアプリケーションで、セルにデータをロードまたは入力することができます。このデモは、XRT/tableのほとんどの機能を使用し、行/列の移動、削除や、表示の変更(色、罫線、行揃え、etc)、セルの編集やトラバースの可・不可、セル範囲の印刷等、テーブルの一般的な操作を実行することができます。
- *surface_editor*は、一つのアプリケーションでXRT/tableとXRT/3dの両方を使用しています。テーブルは、3Dグラフで表示されているZ値を表示しています。ユーザがZ値を編集すると、グラフはそれに応じてアップデートします。
- *table_entry*は、XRT/tableとXRT/fieldを一緒に使用しています。XRT/fieldのサブクラスを幾つか使用し、ユーザによる編集を検証します。(例: integer(整数)フィールドが、電話番号の列を編集する。)

J.3 simple.c

このプログラムは、1.3に解説されています。実行されると、図2が表示されます。

```
#include <Xm/XrtTable.h>
#include <stdlib.h>

int
main(argc, argv)
int argc;
char *argv[];
{
  Widget toplevel, table, title;
  XrtTblCellValues *values;
  XrtTblXmStringList labels;
  XmString s;
  static String label_list[] =
    { "Name", "Address", "Phone Number", "Position", "Salary" };
  FILE *fp;

  /* Open data file */
  if (!(fp = fopen("simple.dat", "r"))) {
    perror("simple.dat");
    exit(0);
  }

  /* Initialize Intrinsics */
  toplevel = XtInitialize(argv[0], "Simple", NULL, 0, &argc, argv);

  /* Read data values, and load into table */
  values = XrtTblReadAscii(toplevel, fp, ',');
  labels.length = XtNumber(label_list);
  labels.list = (XmString *)label_list;
  table = XtVaCreateManagedWidget("table",
    xtXrtTableWidgetClass,    toplevel,
    XmNxrtTblNumRows,         7,
    XmNxrtTblNumColumns,      5,
    XmNxrtTblCellValues,      values,
    XmNxrtTblColumnLabels,    &labels,
    XmNxrtTblBackgroundContext,   "honeydew1",
    XmNxrtTblForegroundContext,   "blue4",
    XmNxrtTblPixelWidthContext,   XRTTBL_VARIABLE,
    XmNxrtTblPixelHeightContext, XRTTBL_VARIABLE,
    XmNxrtTblXmStringSetMode,     XRTTBL_SET_XMSTRING,/* Motif 2.x only */
    NULL);
  XrtTblDestroyCellValues(values);
  fclose(fp);

  /* Format Column Labels */
  XtVaSetValues(table,
    XmNxrtTblContext, XrtTblSetContext(XRTTBL_LABEL, XRTTBL_ALL),
    XmNxrtTblBackgroundContext,   "blue",
    XmNxrtTblForegroundContext,   "white",
    XmNxrtTblAlignmentContext,    XRTTBL_ALIGNMENT_CENTER,
    NULL);
```

```
  /* Change color of President's Salary */
  XtVaSetValues(table,
    XmNxrtTblContext, XrtTblSetContext(0, 4),
    XmNxrtTblForegroundContext, "red",
    NULL);

  /* Add a title */
  title = XtVaCreateWidget("title",
    xtXrtLabelWidgetClass,    table,
    XmNxrtLblBorderType, XRTLBL_BORDER_SHADOW,
    XmNlabelString,
      (s = XmStringCreateSimple("Employee Listing")),
    XmNxrtTblLocation,        XRTTBL_LOCATION_TOP,
    XmNxrtTblTopOffset,  10,
    XmNxrtTblBottomOffset,    20,
    XtVaTypedArg,         XmNfontList,
    XmRString, "-*-times-bold-r-*-*-*-180-*-*-*-*-iso8859-1",
    strlen("-*-times-bold-r-*-*-*-180-*-*-*-*-iso8859-1") +1,
    NULL);

  XmStringFree(s);
  XtRealizeWidget(toplevel);

  XtMainLoop();
}
```

索引

C

C++, 13, 142
clipping arrow, 17, 48, 60
clipマスク, 60
Combo Box, 97
Compound String(複合ストリング参照), 124
context, 18, 25, 26, 139
Contextリソース, 26

M

Motif 2.xの制限, 29
Motifクラス階層, 22, 272

P

Picklist, 258
pixmap, 58
　削除, 59
　指定, 58
　フォーマット, 58
　ラベル, 273
　リソース・ファイル, 259
　レイアウト, 59

S

series, 26
Seriesリソース, 28, 126, 258
Spin Box, 97

U

UIL, 142
UIL引数, 265

X

XmCreateXrtTable(), 191
XmNhighlightThickness, 52
XmNlabelPixmap, 276
XmNlabelString, 276
XmNshadowThickness, 276
XmNstringDirection, 55
XmNxrtLblBackgroundString, 276
XmNxrtLblBorderType, 276
XmNxrtLblForegroundString, 276
XmNxrtLblRotation, 48, 277
XmNxrtTblAlignmentContext, 55, 148
XmNxrtTblAlignmentSeries, 148
XmNxrtTblAllowCellResize, 148
XmNxrtTblAllowResize, 148
XmNxrtTblArgListContext, 111, 113, 114, 148
XmNxrtTblArgListSeries, 148
XmNxrtTblBackgroundContext, 53, 149
XmNxrtTblBackgroundPixelContext, 53, 149
XmNxrtTblBackgroundSeries, 149, 260
XmNxrtTblBorderSidesContext, 57, 149
XmNxrtTblBorderSidesSeries, 149, 260
XmNxrtTblBorderTypeContext, 56, 150
XmNxrtTblBorderTypeSeries, 150
XmNxrtTblBottomOffset, 47, 189
XmNxrtTblBottomShadowContext, 54, 150
XmNxrtTblCaseContext, 96, 150
XmNxrtTblCaseSeries, 150
XmNxrtTblCellValueCallback, 82, 151
XmNxrtTblCellValueContext, 76, 77, 151
XmNxrtTblCellValueFunc, 151
XmNxrtTblCellValues, 73, 81, 152, 259
XmNxrtTblCharHeightContext, 49, 152
XmNxrtTblCharHeightSeries, 152
XmNxrtTblCharWidthContext, 49, 152
XmNxrtTblCharWidthSeries, 152
XmNxrtTblClipChildren, 87, 152
XmNxrtTblClipChildrenPersistent, 152
XmNxrtTblClipPixmapContext, 60, 153
XmNxrtTblClipPixmapSeries, 153
XmNxrtTblColumnLabelOffset, 44, 153
XmNxrtTblColumnLabelPlacement, 44, 153
XmNxrtTblColumnLabels, 16, 42, 43, 153
XmNxrtTblContext, 18, 25, 26, 153
XmNxrtTblContextValueCallback, 154
XmNxrtTblCreateWidgetCallback, 155
XmNxrtTblCurrencyPickListContext, 156
XmNxrtTblCurrencyPickListSeries, 156
XmNxrtTblCursorCorner, 156
XmNxrtTblCursorHorizontal, 156
XmNxrtTblCursorNontraversable, 156

XmNxrtTblCursorTraversable, 156
XmNxrtTblCursorVertical, 156
XmNxrtTblDataTypeContext, 116, 119, 156
XmNxrtTblDataTypeList, 117, 157
XmNxrtTblDataTypeSeries, 158
XmNxrtTblDatePickListContext, 158
XmNxrtTblDatePickListSeries, 159
XmNxrtTblDebug, 30, 159
XmNxrtTblDisplayClipArrows, 60, 159
XmNxrtTblDisplayHorizScrollBar, 159
XmNxrtTblDisplayVertScrollBar, 159
XmNxrtTblDisplayWidgetCallback, 111, 114, 159
XmNxrtTblDoubleBuffer, 138, 160
XmNxrtTblEditableContext, 96, 108, 160
XmNxrtTblEditableSeries, 160
XmNxrtTblEnterCellCallback, 98, 160
XmNxrtTblExportCallback, 133, 161
XmNxrtTblExportDelimiter, 162
XmNxrtTblExportTargets, 130, 162
XmNxrtTblExposeCallback, 123, 163
XmNxrtTblFieldArgListContext, 120, 164
XmNxrtTblFieldArgListSeries, 164
XmNxrtTblFloatPickListContext, 164
XmNxrtTblFloatPickListSeries, 164
XmNxrtTblFontListContext, 55, 164
XmNxrtTblFontListSeries, 164
XmNxrtTblForegroundContext, 164
XmNxrtTblForegroundPixelContext, 53, 165
XmNxrtTblForegroundSeries, 165
XmNxrtTblFrameBorderType, 57, 165
XmNxrtTblFrameShadowThickness, 58, 165
XmNxrtTblFrozenColumnPlacement, 42, 165
XmNxrtTblFrozenColumns, 41, 166
XmNxrtTblFrozenRowPlacement, 42, 166
XmNxrtTblFrozenRows, 41, 166
XmNxrtTblHorizontalScrollBar, 166
XmNxrtTblHorizScrollBarLeftAttachment, 89, 166
XmNxrtTblHorizScrollBarPlacement, 166
XmNxrtTblHorizScrollBarPosition, 89, 166
XmNxrtTblHorizScrollBarRightAttachment, 89, 166
XmNxrtTblImportCallback, 134, 166
XmNxrtTblImportDelimiter, 132, 167
XmNxrtTblImportTargets, 130, 168
XmNxrtTblImportTragets, 107
XmNxrtTblIntPickListContext, 169
XmNxrtTblIntPickListSeries, 169
XmNxrtTblJumpScroll, 90, 169
XmNxrtTblLabelContext, 43, 169
XmNxrtTblLabelCreateCallback, 45, 170
XmNxrtTblLabelCreateFunc, 170
XmNxrtTblLeftColumn, 89, 170
XmNxrtTblLeftOffset, 47, 189
XmNxrtTblLocation, 46, 48, 189
XmNxrtTblMarginHeight, 52, 170
XmNxrtTblMarginWidth, 52, 59, 171
XmNxrtTblMaxLengthContext, 97, 171
XmNxrtTblMaxLengthSeries, 171
XmNxrtTblMinCellVisibility, 95, 171
XmNxrtTblMinFlexibleHeight, 51, 171
XmNxrtTblMinFlexibleWidth, 51, 171
XmNxrtTblMode, 171
XmNxrtTblMultilineContext, 97, 172
XmNxrtTblMultilineSeries, 172
XmNxrtTblNumColumns, 16, 40, 172
XmNxrtTblNumRows, 16, 40, 172
XmNxrtTblOverflowContext, 60, 172
XmNxrtTblOverflowSeries, 172
XmNxrtTblPixelHeight, 51, 92
XmNxrtTblPixelHeightContext, 50, 172
XmNxrtTblPixelHeightSeries, 173
XmNxrtTblPixelWidth, 51, 92
XmNxrtTblPixelWidthContext, 50, 173
XmNxrtTblPixelWidthSeries, 173
XmNxrtTblPixmapContext, 58, 173
XmNxrtTblPixmapLayoutContext, 59, 174
XmNxrtTblPixmapLayoutSeries, 174
XmNxrtTblPixmapSeries, 174, 259
XmNxrtTblPixmapShadowContext, 59, 174
XmNxrtTblPixmapShadowSeries, 174
XmNxrtTblPreviewScrolling, 90, 174
XmNxrtTblPrintCallback, 69, 175
XmNxrtTblPrintWidgetCallback, 71, 176
XmNxrtTblPSFontMapList, 176, 261
XmNxrtTblRangeContext, 26, 176
XmNxrtTblRepaint, 51, 177
XmNxrtTblRepeatBackgroundColors, 54, 177, 261
XmNxrtTblRepeatForegroundColors, 54, 177
XmNxrtTblResizeCallback, 123
XmNxrtTblResizeCellCallback, 177
XmNxrtTblResizeToCell, 189
XmNxrtTblRightOffset, 47, 189
XmNxrtTblRowLabelOffset, 44, 178
XmNxrtTblRowLabelPlacement, 44, 178
XmNxrtTblRowLabels, 43, 178
XmNxrtTblScrollCallback, 92, 179
XmNxrtTblSelectCallback, 179
XmNxrtTblSelectedBackground, 53, 101, 180
XmNxrtTblSelectedCellList, 100, 180
XmNxrtTblSelectedForeground, 53, 101, 180
XmNxrtTblSelectionPolicy, 94, 99, 181
XmNxrtTblSensitiveContext, 104, 181
XmNxrtTblSensitiveSeries, 181
XmNxrtTblSetTextCursorPos, 96, 181
XmNxrtTblShadowThickness, 56, 181

XmNxrtTblSortCallback, 182
XmNxrtTblSpace, 90, 182
XmNxrtTblSpanList, 62, 63, 182
XmNxrtTblStringPickListContext, 182
XmNxrtTblStringPickListSeries, 182
XmNxrtTblText, 182
XmNxrtTblTextInitSelect, 183
XmNxrtTblTopOffset, 47, 189
XmNxrtTblTopRow, 89, 183
XmNxrtTblTopShadowContext, 54, 183
XmNxrtTblTrackCursor, 105, 138, 183
XmNxrtTblTrackCursorCallback, 183
XmNxrtTblTraversableContext, 184
XmNxrtTblTraversableSeries, 184
XmNxrtTblTraverseCellCallback, 95, 138, 184
XmNxrtTblUserDataContext, 65, 184
XmNxrtTblUserDataSeries, 184
XmNxrtTblUseXmGetColors, 185
XmNxrtTblValidateCellCallback, 118, 185
XmNxrtTblValidatePolicy, 116, 185
XmNxrtTblVerticalScrollBar, 186
XmNxrtTblVertScrollBarBottomAttachment, 89, 186
XmNxrtTblVertScrollBarPlacement, 186
XmNxrtTblVertScrollBarPosition, 89, 186
XmNxrtTblVertScrollBarTopAttachment, 89, 186
XmNxrtTblVisibleColumns, 41, 186, 187
XmNxrtTblVisibleRows, 41, 186
XmNxrtTblWidgetClassList, 115, 187
XmNxrtTblWidgetContext, 110, 113
XmNxrtTblWidgetLocation, 110, 189
XmNxrtTblWidgetSeries, 187, 260
XmNxrtTblWidgetSetValueCallback, 114, 187
XmNxrtTblWidgetShadowContext, 115, 187
XmNxrtTblWidgetShadowSeries, 187
XmNxrtTblWordWrapContext, 61, 187
XmNxrtTblWordWrapSeries, 188
XmNxrtTblXmStringGetMode, 29, 44, 78, 188
XmNxrtTblXmStringSetMode, 14, 29, 45, 77, 188
XmRxrtTblRangeList, 260
XmRxrtTblXmStringList, 259
XRT/3dの使用, 130
XRT/fieldの使用, 97, 119
XRT/graphの使用, 130
XRT/table 3.0
  新しい機能, 2
  変更点, 3
  用語の解説, 23
xrt_auth, 12
XRTHOME, 12
XrtLabel, 19, 271
  XRT/tableとの使用, 275
  色, 274
  回転, 274
  罫線, 273
  構成, 272
  作成, 272
  高さと幅, 274
  ラベルの型, 272
  リソース, 276
XrtTblAction, 229
XrtTblAddCallback(), 191
XrtTblAddColumns(), 79, 192
XrtTblAddRows(), 79, 192
XrtTblAlignment, 229
XrtTblAnyValue, 229
XrtTblArgListValue, 230
XrtTblAttachment, 230
XrtTblBooleanValue, 230
XrtTblBorderSide, 230
XrtTblBorderType, 230
XrtTblCancelEdit(), 193
XrtTblCase, 231
XrtTblCellValueCallbackStruct, 231
XrtTblCellValues, 13, 15, 23, 40, 73, 76, 81, 128, 231
XrtTblCellValuesCopy(), 193
XrtTblCheckDispatcher(), 193
XrtTblCheckIsBoolean(), 193
XrtTblCheckIsDouble(), 194
XrtTblCheckIsEmpty(), 194
XrtTblCheckIsFloat(), 194
XrtTblCheckIsInt(), 194
XrtTblColorValue, 231
XrtTblColumnPlacement, 231
XrtTblCommitEdit(), 194
XrtTblCompareStruct, 232
XrtTblContext, 232
XrtTblContextReason, 232
XrtTblContextValue, 233
XrtTblContextValueCallbackStruct, 233
XrtTblCreateSpreadsheetLabel(), 195
XrtTblCreateWidgetCallbackStruct, 234
XrtTblCvtPixelToString(), 195
XrtTblCvtStringToXmString(), 195
XrtTblCvtXmStringToString(), 196
XrtTblDataType, 234
XrtTblDataTypeRoutine, 234
XrtTblDeleteColumns(), 80, 196
XrtTblDeleteRows(), 80, 196
XrtTblDestroyCellvalues(), 196
XrtTblDisplayWidgetCallbackStruct, 235
XrtTblDispScrollBar, 235

XrtTblDoubleBuffer, 235
XrtTblDrawPS(), 67, 197
XrtTblDrawPSarg, 236
XrtTblEnableXrtField(), 119, 197
XrtTblEnterCellCallbackStruct, 237
XrtTblEventToRowCol(), 88, 197
XrtTblExportCallbackStruct, 237
XrtTblExposeCallbackStruct, 237
XrtTblFlash(), 63, 198
XrtTblFontListValue, 237
XrtTblGet3dData(), 130, 198
XrtTblGetCell(), 78, 198
XrtTblGetCellDimensions(), 50, 51, 199
XrtTblGetCellValues(), 199
XrtTblGetCellValuesSize(), 199
XrtTblGetCurrentCell(), 200
XrtTblGetGraphData(), 130, 200
XrtTblGetPreviewProperties(), 200
XrtTblGetRowColByWidget(), 113, 200
XrtTblGetRowColLocation(), 201
XrtTblGetSelectedRange(), 101, 201
XrtTblGetSpanByRowCol(), 63, 201
XrtTblGetWidgetByRowCol(), 202
XrtTblGetXrtFieldByType(), 121, 202
XrtTblImportCallbackStruct, 238
XrtTblImportExportListAdd(), 202
XrtTblImportExportListCopy(), 202
XrtTblImportExportListDelete(), 202
XrtTblImportExportListDestroy(), 203
XrtTblImportExportStruct, 238
XrtTblIntValue, 238
XrtTblIsCellSelected(), 101, 203
XrtTblIsCellTraversable(), 203
XrtTblIsCellVisible(), 91, 203
XrtTblJumpScroll, 238
XrtTblLabelCreateCallbackStruct, 238
XrtTblLocation, 239
XrtTblMakeCellVisible(), 92, 204
XrtTblMode, 239
XrtTblMoveColumns(), 42, 80, 204
XrtTblMoveRows(), 42, 80, 204
XrtTblMrmInitialize(), 143, 204
XrtTblOverflow, 239
XrtTblPickListValue, 239
XrtTblPixmapLayout, 239
XrtTblPixmapValue, 240
XrtTblPointerValue, 240
XrtTblPreviewScroll, 240
XrtTblPrintCallbackStruct, 240
XrtTblPrintWidgetCallbackStruct, 241
XrtTblPSarg, 241
XrtTblPSFontMap, 241
XrtTblRange, 62, 100, 241
XrtTblRangeListAdd(), 63, 101, 205
XrtTblRangeListCopy(), 205
XrtTblRangeListDelete(), 102, 205
XrtTblRangeListFind(), 205
XrtTblRangeListFindRange(), 206
XrtTblRangeListFree(), 206
XrtTblReadAscii(), 15, 76, 206
XrtTblReadResources(), 144, 206
XrtTblReason, 242
XrtTblRemoveAllCallbacks(), 207
XrtTblRemoveCallback(), 207
XrtTblRepaint, 242
XrtTblRepaintRange(), 207
XrtTblResize, 243
XrtTblResizeCallbackStruct, 243
XrtTblResizeCellCallbackStruct, 243
XrtTblRestoreAllCursors(), 208
XrtTblRowPlacement, 243
XrtTblSBPosition, 243
XrtTblScrollCallbackStruct, 244
XrtTblSelectCallbackStruct, 244
XrtTblSelectionPolicy, 244
XrtTblSeries, 126, 244
XrtTblSeriesAdd(), 208
XrtTblSeriesAddValue(), 208
XrtTblSeriesCopy(), 208
XrtTblSeriesCreate(), 208
XrtTblSeriesDelete(), 209
XrtTblSeriesDestroy(), 209
XrtTblSeriesFind(), 128, 209
XrtTblSeriesIterate(), 127, 209
XrtTblSeriesSetValue(), 210
XrtTblSetAllCursors(), 210
XrtTblSetCell(, 210
XrtTblSetCell(), 77
XrtTblSetCellValues(), 211
XrtTblSetContext(), 27, 211
XrtTblSetPreviewProperties(), 58, 211
XrtTblSetPSarg(), 212
XrtTblSetRange(), 212
XrtTblSetReverseConverters(), 212
XrtTblSetXmStringConverter(), 212, 261
XrtTblSortCallbackStruct, 245
XrtTblSortTable(), 213
XrtTblStringValue, 245
XrtTblSyncScrolling(), 213
XrtTblTrackCursorCallbackStruct, 245
XrtTblTraverseCellCallbackStruct, 245
XrtTblTraverseToCell(), 95, 214
XrtTblVaDrawPS(), 66, 214
XrtTblValidateCellCallbackStruct, 118, 246

XrtTblValidatePolicy, 246
XrtTblValueType, 246
XrtTblWidgetClass, 247
XrtTblWidgetPos, 247
XrtTblWidgetSetValueCallbackStruct, 247
XrtTblWidgetValue, 247
XrtTblWriteAscii(), 83, 219
XrtTblWriteResources(), 143, 219
XrtTblWriteSylk(), 83, 220
XrtTblXmStringGet, 247
XrtTblXmStringList, 17, 23, 248
XrtTblXmStringProc, 248
XrtTblXmStringSet, 248
XrtTblXYtoRowCol(), 220
XtIsXrtTable(), 220

あ

アクション・ルーチン
  CancelEdit(), 225
  CommitEdit(), 225
  Drag(), 225
  NoAction(), 87, 107, 225
  ResizeCell(), 92, 226
  Scroll(), 89, 226
  Select(), 103, 226
  TraverseNext(), 227
  TraversePrev(), 227
  TraverseToCell(), 94, 227
アプリケーションの許可, 12
色
  XrtLabel, 274
  影(shadow), 54
  カラーマップのフラッシュ, 30
  繰り返すパターン, 54
  子ウィジェット, 53
  指定, 29, 53
  前景色, 53
  選択されたセル, 53, 101
  背景色, 53
  リソース・ファイル, 260
印刷, 65
  PrintWidgetコールバック, 71
  Printコールバック, 69
エラー・メッセージ, 141

か

カーソルの位置, 96
カレント・セル, 23, 98
環境変数, 12
行揃え, 55
行と列
  移動, 80
  削除, 80
  追加, 79
  非表示, 51
行の高さ, 48
  character指定, 49
  pixel指定, 50
  可変(variable), 50
  フレキシブル, 51
警告, 141
罫線, 56
  種類, 56
  場所, 57
  幅, 56
  リソース・ファイル, 260
子ウィジェット, 24
  XmScrollBar, 24, 91
  XmText, 24, 98
  色, 53
コールバック
  CreateWidget, 113, 155, 187
  DisplayWidget, 114, 148, 159
  EnterCell, 98, 160
  Export, 133
  Expose, 123, 163
  Import, 134
  Resize, 123
  ResizeCell, 93, 177
  Scroll, 92
  Select, 88, 94, 102, 179
  TrackCursor, 105, 183
  TraverseCell, 95, 184
  ValidateCell, 118, 128, 156, 185
コンテキスト(context参照), 18

さ

サンプル・プログラム, 279
  simple.c, 13, 281
  table1.c, 13
スクロール
  強制, 91
  不可, 91
スクロールバー, 89

ジャンプスクロール, 90
配置, 90
表示, 90
セル値, 23
CellValueコールバック, 82
clip, 60
overflow, 60
pixmap, 58
spanning(結合), 62
型, 78
クリア, 81
構造体, 73
指定, 77
取得, 78
複数行, 97
変更, 81
編集, 77, 96
文字数, 97
リソース・ファイル, 259
ワードラップ(wordwrap), 61
セルの選択, 99
セル/ラベル
検査, 117
データ型, 116
セル/ラベル内のウィジェット, 109
クローン, 110
配置, 110
破棄, 113
非管理, 113
リソース設定, 111
リソース・ファイル, 115, 260
センシティビティ, 104

た

タイトル, 19, 46, 271
XrtLabel, 46
位置, 46, 48
回転, 48
削除, 48
追加, 46
ダブルクリック, 88
ダブル・バッファリング, 138
データ, 73
エクスポート(export), 83
簡易マクロ, 74
ロード, 15, 76
データ型, 229
テーブルの構成, 23
テーブルのサイズ, 16, 40, 122
ドラッグ&ドロップ, 106, 130
区切り文字, 132
ターゲット, 130
データ転送フォーマット, 136
独自のターゲット, 134
トラバース, 94
強制, 95
禁止, 94
コントロール, 95
トランスレーション
設定, 87
デフォルト, 223
不可, 87

は

パフォーマンス, 127, 137
フォント, 55, 124, 137, 261
PostScript, 68
複合ストリング, 124
複数のテーブル, 128
フラッシュ, 63
作成, 64
停止, 64
フリーズ, 41
行/列の位置, 42
プレビュー・ウィンドウ, 58
プレビュー・スクロール, 56, 90, 138
フレーム, 57

ま

マクロ, 221
マージン, 52

や

ユーザ・インタラクション, 14, 85
カスタマイズ, 87
デフォルト, 85
ポインタの形, 104

ら

ラベル, 23, 42
  LabelCreateコールバック, 45
  位置, 44
  型, 44
  間隔, 44
  取得, 43
  設定, 16, 43
  リソース・ファイル, 259
リサイズ, 17, 122
  コントロール, 93
  不可, 92
リスト表示モード, 88
リソース
  Context, 26
  Fallback, 127
  Series, 28
  XmString, 29
  XrtLabel, 276
  一覧, 33, 147
  色, 53
  型とクラス, 249
  ポインタ, 28
リソース・ファイル, 31, 143, 257
  構文, 257
列幅, 48
  character指定, 49
  pixel指定, 50
  可変(variable), 50
  フレキシブル, 51