**1. Introduction.** [LDF 2006.09.27.]

**2. Copyright and licenses.** [LDF 2006.10.23.]

The IWF Metadata Harvester is a package for metadata harvesting.



The author is Laurence D. Finston.



The IWF Metadata Harvester is available for downloading from the following FTP server:
`ftp://ftp.gwdg.de/pub/gnu2/iwfmdh/`

Please send bug reports to `lfinsto1@gwdg.de`

The author can be contacted at:

Laurence D. Finston
Kreuzbergring 41
D-37075 Goettingen
Germany

Email: `lfinsto1@gwdg.de`
`s246794@stud.uni-goettingen.de`

**3. Formatting commands.** [LDF 2006.09.27.]
This section contains formatting commands. They don't appear in the TEX output of CWEAVE.

**4. File Lists.** [LDF 2006.09.27.]

**5. Source Files.** [LDF 2006.09.27.]

**6. Logical and Hierarchical Order.** [LDF 2006.09.27.]

⟨ `resource.web` 12 ⟩ . . . . . . . . . . . . . . . . . . . . . . . . . . . . Preprocessor macro definitions for resources

⟨ `stdafx.web` 16 ⟩ . . . . . . . . . . . . . . . . . . . . . . . . . . . . . Main header file for ATest

⟨ `mainfrm.web` 28 ⟩ . . . . . . . . . . . . . . . . . . . . . . . . . . . . **class CMainFrame**

⟨ `atest.web` 50 ⟩ . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . **class CATestApp** and **class CAboutDlg**

⟨ `atestdoc.web` 75 ⟩ . . . . . . . . . . . . . . . . . . . . . . . . . . . **class CATestDoc**

⟨ `atstview.web` 96 ⟩ . . . . . . . . . . . . . . . . . . . . . . . . . . . **class CATestView**

⟨ `dialog1a.web` 130 ⟩ . . . . . . . . . . . . . . . . . . . . . . . . . . **class Dialog_1**

⟨ `dialog2a.web` 208 ⟩ . . . . . . . . . . . . . . . . . . . . . . . . . . **class Dialog_2**

⟨ `glblfncs.web` 294 ⟩ . . . . . . . . . . . . . . . . . . . . . . . . . . Global functions

⟨ `mtdtsrc.web` 312 ⟩ . . . . . . . . . . . . . . . . . . . . . . . . . . . **class Metadata_Source**

⟨ `selector.web` 379 ⟩ . . . . . . . . . . . . . . . . . . . . . . . . . . **class Selector**

**7. ODBC Classes.** [LDF 2006.09.27.]

⟨`records.web` 503⟩ ... **class Records**
⟨`rcrdstmp.web` 524⟩ ... **class Records_Temp**
⟨`creators.web` 545⟩ ... **class Creators**
⟨`crtrstmp.web` 566⟩ ... **class Creators_Temp**
⟨`cntrbtrs.web` 587⟩ ... **class Contributors**
⟨`cntrbtmp.web` 608⟩ ... **class Contributors_Temp**
⟨`titles.web` 629⟩ ... **class Titles**
⟨`ttlstmp.web` 650⟩ ... **class Titles_Temp**
⟨`dscrptmp.web` 671⟩ ... **class Descriptions_Temp**
⟨`subjects.web` 692⟩ ... **class Subjects**
⟨`sbjcttmp.web` 713⟩ ... **class Subjects_Temp**
⟨`identtmp.web` 734⟩ ... **class Identifiers_Temp**
⟨`langtmp.web` 755⟩ ... **class Languages_Temp**
⟨`pblshtmp.web` 776⟩ ... **class Publishers_Temp**
⟨`rghtstmp.web` 797⟩ ... **class Rights_Temp**
⟨`typestmp.web` 818⟩ ... **class Types_Temp**
⟨`tempids.web` 839⟩ ... **class Temp_IDs**
⟨`tempids1.web` 860⟩ ... **class Temp_IDs_1**

**8. Other files.** [LDF 2006.10.05.]

⟨`spchrtbl.web` 10⟩ ... Special Character Formatting

**9. Alphabetical Order.** [LDF 2006.09.27.]

⟨`atest.web` 50⟩ ... **class CATestApp** and **class CAboutDlg**
⟨`atestdoc.web` 75⟩ ... **class CATestDoc**
⟨`atstview.web` 96⟩ ... **class CATestView**
⟨`cntrbtmp.web` 608⟩ ... **class Contributors_Temp**
⟨`cntrbtrs.web` 587⟩ ... **class Contributors**
⟨`creators.web` 545⟩ ... **class Creators**
⟨`crtrstmp.web` 566⟩ ... **class Creators_Temp**
⟨`dialog1a.web` 130⟩ ... **class Dialog_1**
⟨`dialog2a.web` 208⟩ ... **class Dialog_2**
⟨`dscrptmp.web` 671⟩ ... **class Descriptions_Temp**
⟨`glblfncs.web` 294⟩ ... Global functions
⟨`identtmp.web` 734⟩ ... **class Identifiers_Temp**
⟨`langtmp.web` 755⟩ ... **class Languages_Temp**
⟨`mainfrm.web` 28⟩ ... **class CMainFrame**
⟨`mtdtsrc.web` 312⟩ ... **class Metadata_Source**
⟨`pblshtmp.web` 776⟩ ... **class Publishers_Temp**
⟨`rcrdstmp.web` 524⟩ ... **class Records_Temp**
⟨`records.web` 503⟩ ... **class Records**

**10. Special character formatting.** [LDF 2006.09.28.]

**Log**

[LDF 2006.10.05.] Renamed this file from `spchrtbl.tex` to `spchrtbl.web`.

---

⟨ `spchrtbl.web` 10 ⟩ ≡ /* Empty section for use in cross-references. */

This code is cited in sections 8 and 9.

This code is used in section 881.

**11.**

See also ⟨ Define **Metadata_Source** :: *parse_record* 349 ⟩ and “UTF-8 encoding table and Unicode characters” (`http://www.utf8-chartable.de/`).

For more information about UTF-8 encoding, see
“UTF-8—Wikipedia, the free encyclopedia” (`http://en.wikipedia.org/wiki/Utf-8`).

| Hex. | Oct. | Dec. | UTF-8 | Enc. | Sym. | Unicode Name | Id. | Lit. | TeX |
|---|---|---|---|---|---|---|---|---|---|
| #80 | °*200* | 128 | #C2 80 | Â\200 | \200 | ⟨control⟩ | \200 | "\200" | \200 |
| #81 | °*201* | 129 | #C2 81 | Â\201 | \201 | ⟨control⟩ | \201 | "\201" | \201 |
| #82 | °*202* | 130 | #C2 82 | Â\202 | \202 | ⟨control⟩ | \202 | "\202" | \202 |
| #83 | °*203* | 131 | #C2 83 | Â\203 | \203 | ⟨control⟩ | \203 | "\203" | \203 |
| #84 | °*204* | 132 | #C2 84 | Â\204 | \204 | ⟨control⟩ | \204 | "\204" | \204 |
| #85 | °*205* | 133 | #C2 85 | Â\205 | \205 | ⟨control⟩ | \205 | "\205" | \205 |
| #86 | °*206* | 134 | #C2 86 | Â\206 | \206 | ⟨control⟩ | \206 | "\206" | \206 |
| #87 | °*207* | 135 | #C2 87 | Â\207 | \207 | ⟨control⟩ | \207 | "\207" | \207 |
| #88 | °*210* | 136 | #C2 88 | Â\210 | \210 | ⟨control⟩ | \210 | "\210" | \210 |
| #89 | °*211* | 137 | #C2 89 | Â\211 | \211 | ⟨control⟩ | \211 | "\211" | \211 |
| #8A | °*212* | 138 | #C2 8A | Â\212 | \212 | ⟨control⟩ | \212 | "\212" | \212 |
| #8B | °*213* | 139 | #C2 8B | Â\213 | \213 | ⟨control⟩ | \213 | "\213" | \213 |
| #8C | °*214* | 140 | #C2 8C | Â\214 | \214 | ⟨control⟩ | \214 | "\214" | \214 |
| #8D | °*215* | 141 | #C2 8D | Â\215 | \215 | ⟨control⟩ | \215 | "\215" | \215 |
| #8E | °*216* | 142 | #C2 8E | Â\216 | \216 | ⟨control⟩ | \216 | "\216" | \216 |
| #8F | °*217* | 143 | #C2 8F | Â\217 | \217 | ⟨control⟩ | \217 | "\217" | \217 |
| #93 | °*223* | 147 | #C2 93 | Â\223 | \223 | ⟨control⟩ | \223 | "\223" | \223 |
| #96 | °*226* | 150 | #C2 96 | Â\226 | \226 | ⟨control⟩ | \226 | "\226" | \226 |
| #97 | °*227* | 151 | #C2 97 | Â\227 | \227 | ⟨control⟩ | \227 | "\227" | \227 |
| #9B | °*233* | 155 | #C2 9B | Â\233 | \233 | ⟨control⟩ | \233 | "\233" | \233 |
| #9C | °*234* | 156 | #C2 9C | Â\234 | \234 | ⟨control⟩ | \234 | "\234" | \234 |
| #9F | °*237* | 159 | #C2 9F | Â\237 | \237 | ⟨control⟩ | \237 | "\237" | \237 |
| #A0 | °*240* | 160 | #C2 A0 | Â\240 | \240 | No-Break Space | \240 | "\240" | |
| #A1 | °*241* | 161 | #C2 A1 | Â¡ | ¡ | Inverted Exclamation Mark | *¡* | "¡" | ¡ |
| #A4 | °*244* | 164 | #C2 A4 | Â¤ | ¤ | Currency Sign | *¤* | "¤" | ¤ |
| #A8 | °*250* | 168 | #C2 A8 | Â¨ | ¨ | Diaeresis | ¨ | "¨" | ¨ |
| #A7 | °*247* | 167 | #C2 A7 | Â§ | § | Section Sign | *§* | "§" | § |
| #A9 | °*251* | 169 | #C2 A9 | Â© | © | Copyright Sign | *©* | "©" | © |
| #AD | °*255* | 173 | #C2 AD | Â- | - | Soft Hyphen | - | "-" | - |
| #AE | °*256* | 174 | #C2 AE | Â® | ® | Registered Sign | *®* | "®" | ® |
| #AF | °*257* | 175 | #C2 AF | Â¯ | ¯ | Macron | ¯ | "¯" | ¯ |

(cont.)

| Hex. | Oct. | Dec. | UTF-8 | Enc. | Sym. | Unicode Name | Id. | Lit. | TEX |
|---|---|---|---|---|---|---|---|---|---|
| #B0 | °260 | 176 | #C2 B0 | Â° | ° | Degree Sign | ° | "°" | ° |
| #B1 | °261 | 177 | #C2 B1 | Â± | ± | Plus-Minus Sign | ± | "±" | ± |
| #B3 | °263 | 179 | #C2 B3 | Â³ | ³ | Superscript Three | ³ | "³" | ³ |
| #B4 | °264 | 180 | #C2 B4 | Â´ | ´ | Acute Accent | ´ | "´" | ´ |
| #B5 | °265 | 181 | #C2 B5 | Âµ | µ | Micro Sign | µ | "µ" | µ |
| #B6 | °266 | 182 | #C2 B6 | Â¶ | ¶ | Pilcrow Sign | ¶ | "¶" | ¶ |
| #B7 | °267 | 183 | #C2 B7 | Â· | · | Middle Dot | · | "·" | · |
| #B8 | °270 | 184 | #C2 B8 | Â¸ | ¸ | Cedilla | ¸ | "¸" | ¸ |
| #BC | °274 | 188 | #C2 BC | Â¼ | ¼ | Vulgar Fraction One Quarter | ¼ | "¼" | ¼ |
| #BD | °275 | 189 | #C2 BD | Â½ | ½ | Vulgar Fraction One Half | ½ | "½" | ½ |
| #BF | °277 | 191 | #C2 BF | Â¿ | ¿ | Inverted Question Mark | ¿ | "¿" | ¿ |
| #C0 | °300 | 192 | #C3 80 | Ã\200 | À | Latin Capital Letter A With Grave | À | "À" | À |
| #C1 | °301 | 193 | #C3 81 | Ã\201 | Á | Latin Capital Letter A With Acute | Á | "Á" | Á |
| #C2 | °302 | 194 | #C3 82 | Ã\202 | Â | Latin Capital Letter A With Circumflex | Â | "Â" | Â |
| #C3 | °303 | 195 | #C3 83 | Ã\203 | Ã | Latin Capital Letter A With Tilde | Ã | "Ã" | Ã |
| #C4 | °304 | 196 | #C3 84 | Ã\204 | Ä | Latin Capital Letter A With Diaeresis | Ä | "Ä" | Ä |
| #C5 | °305 | 197 | #C3 85 | Ã\205 | Å | Latin Capital Letter A With Ring Above | Å | "Å" | Å |
| #C6 | °306 | 198 | #C3 86 | Ã\206 | Æ | Latin Capital Letter AE | Æ | "Æ" | Æ |
| #C7 | °307 | 199 | #C3 87 | Ã\207 | Ç | Latin Capital Letter C With Cedilla | Ç | "Ç" | Ç |
| #C8 | °310 | 200 | #C3 88 | Ã\210 | È | Latin Capital Letter E With Grave | È | "È" | È |
| #C9 | °311 | 201 | #C3 89 | Ã\211 | É | Latin Capital Letter E With Acute | É | "É" | É |
| #CA | °312 | 202 | #C3 8A | Ã\212 | Ê | Latin Capital Letter E With Circumflex | Ê | "Ê" | Ê |
| #CB | °313 | 203 | #C3 8B | Ã\213 | Ë | Latin Capital Letter E With Diaeresis | Ë | "Ë" | Ë |
| #CE | °316 | 206 | #C3 8E | Ã\216 | Î | Latin Capital Letter I With Circumflex | Î | "Î" | Î |
| #D6 | °326 | 214 | #C3 96 | Ã\226 | Ö | Latin Capital Letter O With Diaeresis | Ö | "Ö" | Ö |
| #D7 | °327 | 215 | #C3 97 | Ã\227 | × | Multiplication Sign | × | "×" | × |
| #DB | °333 | 219 | #C3 9B | Ã\233 | Û | Latin Capital Letter U With Circumflex | Û | "Û" | Û |
| #DC | °334 | 220 | #C3 9C | Ã\234 | Ü | Latin Capital Letter U With Diaeresis | Ü | "Ü" | Ü |
| #DF | °337 | 223 | #C3 9F | Ã\237 | ß | Latin Small Letter Sharp S | ß | "ß" | ß |

(cont.)

| Hex. | Oct. | Dec. | UTF-8 | Enc. | Sym. | Unicode Name | Id. | Lit. | TEX |
|---|---|---|---|---|---|---|---|---|---|
| #E0 | °*340* | 224 | #C3 A0 | Ã\240 | à | Latin Small Letter A With Grave | *à* | "à" | à |
| #E1 | °*341* | 225 | #C3 A1 | Ã¡ | á | Latin Small Letter A With Acute | *á* | "á" | á |
| #E4 | °*344* | 228 | #C3 A4 | Ã¤ | ä | Latin Small Letter A With Diaeresis | *ä* | "ä" | ä |
| #E7 | °*347* | 231 | #C3 A7 | Ã§ | ç | Latin Small Letter C With Cedilla | *ç* | "ç" | ç |
| #E8 | °*350* | 232 | #C3 A8 | Ã¨ | è | Latin Small Letter E With Grave | *è* | "è" | è |
| #E9 | °*351* | 233 | #C3 A9 | Ã© | é | Latin Small Letter E With Acute | *é* | "é" | é |
| #ED | °*355* | 237 | #C3 AD | Ã- | í | Latin Small Letter I With Acute | *í* | "í" | í |
| #EE | °*356* | 238 | #C3 AE | Ã® | î | Latin Small Letter I With Circumflex | *î* | "î" | î |
| #EF | °*357* | 239 | #C3 AF | Ã¯ | ï | Latin Small Letter I With Diaeresis | *ï* | "ï" | ï |
| #F1 | °*361* | 241 | #C3 B1 | Ã± | ñ | Latin Small Letter N With Tilde | *ñ* | "ñ" | ñ |
| #F3 | °*363* | 243 | #C3 B3 | Ã³ | ó | Latin Small Letter O With Acute | *ó* | "ó" | ó |
| #F6 | °*366* | 246 | #C3 B6 | Ã¶ | ö | Latin Small Letter O With Diaeresis | *ö* | "ö" | ö |
| #FC | °*374* | 252 | #C3 BC | Ã¼ | ü | Latin Small Letter U With Diaeresis | *ü* | "ü" | ü |
| #FD | °*375* | 253 | #C3 BD | Ã½ | ý | Latin Small Letter Y With Acute | *ý* | "ý" | ý |
| #FF | °*377* | 255 | #C3 BF | Ã¿ | ÿ | Latin Small Letter Y With Diaeresis | *ÿ* | "ÿ" | ÿ |

⟨Special character formatting 11⟩ ≡ /* Empty section. Used for cross-references. */

This code is cited in section 373.

This code is used in section 881.

**12. Preprocessor macro definitions for resources (`resource.web`).** [LDF 2006.10.12.]

**Log**

[LDF 2006.10.12.] Created this file. It contains code formerly in `resource.h`.

---

⟨`resource.web` 12⟩ ≡ /* Empty section for use in cross-references. */

This code is cited in sections 6 and 9.

This code is used in section 881.

**13. Putting `resource.web` together.**

**14.** This is what `resource.h` contains. It's generated automatically by Microsoft Visual Studio. The code must be inserted here by hand, because the first comment doesn't obey the CWEB's syntax rules. [LDF 2006.10.17.]

**Log**

[LDF 2006.10.17.] No longer using CTANGLE to generate `resource.h`.

---

⟨ `resource.h` contents 14 ⟩ ≡

```
//{{NO_DEPENDENCIES}}   /* Microsoft Visual C++ generated include file. */
  /* Used by ATest.rc */   /* */
#define IDD_ABOUTBOX 100
#define IDP_OLE_INIT_FAILED 100
#define IDR_MAINFRAME 128
#define IDR_ATestTYPE 129
#define IDD_DIALOG1 130
#define IDD_DIALOG2 131
#define IDC_MESSAGES 1000
#define IDC_TODAY 1002
#define IDC_YESTERDAY 1003
#define IDC_THIS_WEEK 1004
#define IDC_LAST_WEEK 1005
#define IDC_THIS_MONTH 1006
#define IDC_LAST_MONTH 1007
#define IDC_LAST_6_MONTHS 1008
#define IDC_THIS_YEAR 1009
#define IDC_LAST_YEAR 1010
#define IDC_LAST_2_YEARS 1011
#define IDC_LAST_5_YEARS 1012
#define IDC_LAST_10_YEARS 1013
#define IDC_LAST_20_YEARS 1014
#define IDC_ALL_RECORDS 1015
#define IDC_DOWNLOAD 1018
#define IDC_RESULTS 1024
#define IDC_SEARCH 1025
#define IDC_SEARCH_STRING 1026
#define IDC_SELECT 1027
#define IDC_BEG_OR_WHOLE_WORD 1030
#define IDC_WHOLE_WORD_ONLY 1031
#define IDC_WHOLE_OR_PARTIAL_WORD 1032
#define IDC_CASE_IGNORE 1033
#define IDC_DELETE_OLD_RECORDS 1034
#define IDC_EXACT_MATCH 1036
#define IDC_LIST_RECORDS 1037
#define IDC_SORT_BY 1038
#define IDC_ALL_DATES 1044
#define IDC_SINCE_LAST_YEAR 1045
#define IDC_DESCENDING 1051
#define IDC_USE_DC_DATE 1055
#define IDC_USE_HEADER_DATESTAMP 1056
#define IDC_NO_DUPLICATES 1057
#define IDC_LIST_TITLES 1058
```

```
#define IDC_CREATORS 1059
#define IDC_SUBJECTS 1060
#define IDC_CONTRIBUTORS 1063
#define IDC_TIMMS 1064
#define IDC_DBT 1065
#define IDC_RADIO3 1066
#define IDC_RADIO4 1067
#define IDC_HOTKEY1 1068
#define IDC_CHECK1 1069    /* Next default values for new objects */    /* */
#ifdef APSTUDIO_INVOKED
#ifndef APSTUDIO_READONLY_SYMBOLS
#define _APS_NEXT_RESOURCE_VALUE 132
#define _APS_NEXT_COMMAND_VALUE 32771
#define _APS_NEXT_CONTROL_VALUE 1070
#define _APS_NEXT_SYMED_VALUE 101
#endif
#endif
```

This code is cited in section 15.

This code is used in section 15.

**15.** ⟨ `resource.h` contents 14 ⟩ is included here to prevent CTANGLE and CWEAVE from issuing warnings. [LDF 2006.10.17.]

⟨ Dummy sections 15 ⟩ ≡

```
#if 0
  ⟨ resource.h contents 14 ⟩
#endif
```

See also section 881.

This code is used in section 884.

**16. stdafx (`stdafx.web`).** [LDF 2006.10.05.]

**Log**

[LDF 2006.10.05.] Created this file. It contains code formerly in `stdafx.h` and `stdafx.cpp`.

---

⟨ `stdafx.web` 16 ⟩ ≡ /* Empty section for use in cross-references. */

This code is cited in sections 6 and 9.

This code is used in section 881.

**17. Preprocessor macro calls.**

⟨ Preprocessor macro calls 17 ⟩ ≡

```
#pragma once
```

See also sections 30, 52, 77, 98, 132, 210, 314, 381, 504, 525, 546, 567, 588, 609, 630, 651, 672, 693, 714, 735, 756, 777, 798, 819, 840, and 861.

This code is used in sections 27, 49, 74, 95, 129, 207, 293, 378, 502, 522, 543, 564, 585, 606, 627, 648, 669, 690, 711, 732, 753, 774, 795, 816, 837, 858, and 879.

**18. Preprocessor macro definitions.**

⟨ Preprocessor macro definitions 18 ⟩ ≡

```
#ifndef VC_EXTRALEAN
#define VC_EXTRALEAN
#endif
#ifndef WINVER
#define WINVER #0400
#endif
#ifndef _WIN32_WINNT
#define _WIN32_WINNT #0400
#endif
#ifndef _WIN32_WINDOWS
#define _WIN32_WINDOWS #0410
#endif
#ifndef _WIN32_IE
#define _WIN32_IE #0400
#endif
#define _ATL_CSTRING_EXPLICIT_CONSTRUCTORS
#define _AFX_ALL_WARNINGS
```

See also sections 31, 53, 78, 99, 211, and 382.

This code is used in sections 27, 48, 73, 128, 292, and 501.

**19. Include files in `stdafx.h`.**

**20.** Windows and Visual Studio headers. [LDF Undated.]

```
⟨Include files 20⟩ ≡
#include <afxwin.h>
#include <afxext.h>
#include <afxdisp.h>
#include <afxdtctl.h>
#include <afxinet.h>
#ifndef _AFX_NO_AFXCMN_SUPPORT
#include <afxcmn.h>
#endif
#include <afxdb.h>
#include <afxmt.h>
```

See also sections 21, 23, 29, 51, 76, 97, 131, 209, 295, 313, 380, 506, 527, 548, 569, 590, 611, 632, 653, 674, 695, 716, 737, 758, 779, 800, 821, 842, and 863.

This code is used in sections 27, 48, 73, 94, 128, 206, 292, 310, 377, 501, 523, 544, 565, 586, 607, 628, 649, 670, 691, 712, 733, 754, 775, 796, 817, 838, 859, and 880.

**21.** C++ standard library headers. [LDF Undated.]

```
⟨Include files 20⟩ +≡
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ios>
#include <iomanip>
#include <iostream>
#include <fstream>
#include <sstream>
#include <strstream>
#include <map>
#include <set>
```

**22.** Header files for types declared in ATest. [LDF Undated.]

**23.** Headers for classes derived from **CRecordset**. [LDF Undated.]

**Log**

[LDF 2006.10.23.] Now including `private.h`.

[LDF 2006.10.27.] Now including `glblfncs.h`.

⟨Include files 20⟩ +≡

```
#include "private.h"
#include "glblfncs.h"
#include ".\records.h"
#include ".\rcrdstmp.h"
#include ".\tempids.h"
#include ".\tempids1.h"
#include ".\cntrbtmp.h"
#include ".\crtrstmp.h"
#include ".\dscrptmp.h"
#include ".\identtmp.h"
#include ".\langtmp.h"
#include ".\pblshtmp.h"
#include ".\rghtstmp.h"
#include ".\sbjcttmp.h"
#include ".\ttlstmp.h"
#include ".\typestmp.h"
#include ".\cntrbtrs.h"
#include ".\creators.h"
#include ".\subjects.h"
#include ".\titles.h"
#include ".\mtdtsrc.h"
#include ".\selector.h"
```

**24. Global variables.** [LDF Undated.]

**Log**

[LDF 2006.10.27.] Added the **extern** declarations for **CMutex** *special_char_encodings_map_mutex* and **map⟨string, char⟩** *special_char_encodings_map*.

[LDF 2006.12.11.] Added the **extern** declaration of **string** *copyright_html_str*.

⟨Global variables 24⟩ ≡

```
extern CMutex log_file_mutex;
extern CMutex special_char_encodings_map_mutex;
extern map<string, char> special_char_encodings_map;
extern string copyright_html_str;
```

This code is used in section 27.

**25. Putting `stdafx.web` together.**

**26.** This is what's compiled. [LDF Undated.]

```
#include "stdafx.h"
```

**27.** This is what's written to `stdafx.h`.

⟨ `stdafx.h` 27 ⟩ ≡
  ⟨ Preprocessor macro calls 17 ⟩
  ⟨ Preprocessor macro definitions 18 ⟩
  ⟨ Include files 20 ⟩
  ⟨ Global variables 24 ⟩

**28. CMainFrame (`mainfrm.web`).** [LDF 2006.10.12.]

**Log**

[LDF 2006.10.12.] Created this file. It contains code formerly in `MainFrm.h` and `MainFrm.cpp`.

---

⟨ `mainfrm.web` 28 ⟩ ≡ /* Empty section for use in cross-references. */

This code is cited in sections 6 and 9.

This code is used in section 881.

**29. Include files.**

⟨ Include files 20 ⟩ +≡
**#include** "stdafx.h"
**#include** "ATest.h"
**#include** "MainFrm.h"

**30. Preprocessor macro calls.**

⟨ Preprocessor macro calls 17 ⟩ +≡
**#pragma** *once*

**31. Preprocessor macro definitions.**

⟨ Preprocessor macro definitions 18 ⟩ +≡
**#ifdef** _DEBUG
**#define new** DEBUG_NEW
**#endif**

**32. Declare class CMainFrame.** [LDF Undated.]

⟨ Declare **class CMainFrame** 32 ⟩ ≡
  **class CMainFrame** : **public CFrameWnd** {
  **protected**: ⟨ Declare **protected CMainFrame** functions 35 ⟩
  **public**: ⟨ Declare **public CMainFrame** functions 37 ⟩
  **protected**: **CStatusBar** *m_wndStatusBar*;
    **CToolBar** *m_wndToolBar*;
    DECLARE_MESSAGE_MAP( )
  };

This code is used in section 49.

**33. Declare staticindicators array.** [LDF Undated.]

⟨ Declare **static** *indicators* array 33 ⟩ ≡
  **static UINT** *indicators*[ ] = {ID_SEPARATOR, ID_INDICATOR_CAPS, ID_INDICATOR_NUM,
    ID_INDICATOR_SCRL, };

This code is used in section 48.

**34. Functions.** [LDF 2006.10.12.]

**35. Constructor.** [LDF Undated.]

⟨ Declare **protected CMainFrame** functions 35 ⟩ ≡
  **CMainFrame**( ); `DECLARE_DYNCREATE`(**CMainFrame**)

See also section 45.
This code is used in section 32.

**36.**

⟨ Define **CMainFrame** functions 36 ⟩ ≡
```
CMainFrame::CMainFrame()
{
  return;
}
```
See also sections 38, 40, 42, 44, and 46.
This code is used in section 48.

**37. Destructor.** [LDF Undated.]

⟨ Declare **public CMainFrame** functions 37 ⟩ ≡
```
virtual ~CMainFrame();
```
See also sections 39, 41, and 43.
This code is used in section 32.

**38.**

⟨ Define **CMainFrame** functions 36 ⟩ +≡
```
CMainFrame::~CMainFrame()
{
  return;
}
```

**39. Pre-Create Window.** [LDF Undated.]

⟨ Declare **public CMainFrame** functions 37 ⟩ +≡
```
virtual BOOL PreCreateWindow(CREATESTRUCT &cs);
```

**40.**

⟨ Define **CMainFrame** functions 36 ⟩ +≡
```
BOOL CMainFrame::PreCreateWindow(CREATESTRUCT &cs)
{
  if (¬CFrameWnd::PreCreateWindow(cs)) return FALSE;
  return TRUE;
}
```

**41. Assert Valid.** [LDF Undated.]

⟨ Declare **public CMainFrame** functions 37 ⟩ +≡
```
#ifdef _DEBUG
  virtual void AssertValid() const;
#endif
```

**42.**

⟨ Define **CMainFrame** functions 36 ⟩ +≡
```
#ifdef _DEBUG
  void CMainFrame::AssertValid() const
  {
    CFrameWnd::AssertValid();
  }
#endif
```

**43. Dump.** [LDF Undated.]

⟨ Declare **public CMainFrame** functions 37 ⟩ +≡
```
#ifdef _DEBUG
  virtual void Dump(CDumpContext &dc) const;
#endif
```

**44.**

⟨ Define **CMainFrame** functions 36 ⟩ +≡
```
#ifdef _DEBUG
  void CMainFrame::Dump(CDumpContext &dc) const
  {
    CFrameWnd::Dump(dc);
  }
#endif
```

**45. On Create.** [LDF Undated.]

⟨ Declare **protected CMainFrame** functions 35 ⟩ +≡
```
  afx_msg int OnCreate(LPCREATESTRUCT lpCreateStruct);
```

**46.**

⟨ Define **CMainFrame** functions 36 ⟩ +≡
```
  int CMainFrame::OnCreate(LPCREATESTRUCT lpCreateStruct)
  {
    if (CFrameWnd::OnCreate(lpCreateStruct) ≡ −1) return −1;
    if (¬m_wndToolBar.CreateEx(this,
          TBSTYLE_FLAT, WS_CHILD | WS_VISIBLE | CBRS_TOP | CBRS_GRIPPER | CBRS_TOOLTIPS |
          CBRS_FLYBY | CBRS_SIZE_DYNAMIC) ∨ ¬m_wndToolBar.LoadToolBar(IDR_MAINFRAME)) {
      TRACE0("Symbolleiste␣konnte␣nicht␣erstellt␣werden\n");
      return −1;    /* Fehler bei Erstellung */
    }
    if (¬m_wndStatusBar.Create(this) ∨ ¬m_wndStatusBar.SetIndicators(indicators, sizeof
          (indicators)/sizeof(UINT))) {
      TRACE0("Statusleiste␣konnte␣nicht␣erstellt␣werden\n");
      return −1;    /* Fehler bei Erstellung */
    }
    m_wndToolBar.EnableDocking(CBRS_ALIGN_ANY);
    EnableDocking(CBRS_ALIGN_ANY);
    DockControlBar(&m_wndToolBar);
    return 0;
  }
```

**47. Putting CMainFrame together.**

**48.** This is what's compiled.

⟨ Include files 20 ⟩
⟨ Preprocessor macro definitions 18 ⟩
IMPLEMENT_DYNCREATE(**CMainFrame**, **CFrameWnd**)
BEGIN_MESSAGE_MAP(**CMainFrame**, **CFrameWnd**)
ON_WM_CREATE( )
END_MESSAGE_MAP( )
⟨ Declare **static** *indicators* array 33 ⟩
⟨ Define **CMainFrame** functions 36 ⟩

**49.** This is what's written to `mainfrm.h`.

⟨ `mainfrm.h` 49 ⟩ ≡
  ⟨ Preprocessor macro calls 17 ⟩
  ⟨ Declare **class CMainFrame** 32 ⟩

**50. CATestApp and CABoutDlg (`atest.web`).** [LDF 2006.09.27.]

⟨ `atest.web` 50 ⟩ ≡ /* Empty section for use in cross-references. */

This code is cited in sections 6 and 9.

This code is used in section 881.

**51. Include files.**

⟨ Include files 20 ⟩ +≡
**#include** "stdafx.h"
**#include** "ATest.h"
**#include** "MainFrm.h"
**#include** "ATestDoc.h"
**#include** "atstview.h"
**#include** "dialog1a.h"
**#include** "dialog2a.h"
**#ifdef** _DEBUG
**#define new** DEBUG_NEW
**#endif**

**52. Preprocessor macro calls.**

⟨ Preprocessor macro calls 17 ⟩ +≡
**#pragma** *once*

**53. Preprocessor macro definitions.**

⟨ Preprocessor macro definitions 18 ⟩ +≡

**54. class CATestApp declaration.**

⟨ **class CATestApp** declaration 54 ⟩ ≡
  **class CATestApp** : **public CWinApp** {
  **public**: ⟨ Declare **CATestApp** functions 59 ⟩
    DECLARE_MESSAGE_MAP( )
  };

This code is used in section 74.

**55. Global variable declarations.** [LDF Undated.]

**Log**

[LDF 2006.10.27.] Added the declarations of **CMutex** *special_char_encodings_map_mutex* and **map**⟨**string**, **char**⟩ *special_char_encodings_map*.

[LDF 2006.12.11.] Added the declaration of **string** *copyright_html_str*.

---

⟨ Global variable declarations 55 ⟩ ≡
  **CMutex** *log_file_mutex*;
  **CATestApp** *theApp*;
  **CMutex** *special_char_encodings_map_mutex*;

**map**⟨**string**, **char**⟩ *special_char_encodings_map*;
**string** *copyright_html_str*;

See also section 213.

This code is used in sections 73 and 292.

**56. extern global variable declarations.** [LDF Undated.]

⟨ **extern** global variable declarations 56 ⟩ ≡
**extern CATestApp** *theApp*;

See also section 214.

This code is used in sections 74 and 293.

**57. CATestApp message map.** [LDF Undated.]

⟨ **CATestApp** message map 57 ⟩ ≡

```
BEGIN_MESSAGE_MAP(CATestApp, CWinApp)
ON_COMMAND(ID_APP_ABOUT, OnAppAbout)
ON_COMMAND(ID_FILE_NEW, CWinApp::OnFileNew)
ON_COMMAND(ID_FILE_OPEN, CWinApp::OnFileOpen)
ON_COMMAND(ID_FILE_PRINT_SETUP, CWinApp::OnFilePrintSetup)
END_MESSAGE_MAP()
```

This code is used in section 73.

**58. Functions.** [LDF Undated.]

**59. Constructor.** [LDF Undated.]

⟨ Declare **CATestApp** functions 59 ⟩ ≡
**CATestApp**( );

See also sections 61 and 63.

This code is used in section 54.

**60.**

⟨ Define **CATestApp** functions 60 ⟩ ≡

```
CATestApp::CATestApp(void)
{
  return;
}
```

See also sections 62 and 64.

This code is used in section 73.

**61. Initialize instance.** [LDF Undated.]

**Log**

[LDF 2006.10.27.] Now calling the global function *init_special_char_encodings_map*.

[LDF 2006.12.11.] Now calling the global function *init_copyright_strings*.

---

⟨ Declare **CATestApp** functions 59 ⟩ +≡
**virtual BOOL** *InitInstance*( );

**62.**

```
⟨ Define CATestApp functions 60 ⟩ +≡
  BOOL CATestApp::InitInstance()
  {
    InitCommonControls();
    CWinApp::InitInstance();
    if (¬AfxOleInit()) {
      AfxMessageBox(IDP_OLE_INIT_FAILED);
      return FALSE;
    }
    AfxEnableControlContainer();
    SetRegistryKey(_T("Vom␣lokalen␣Anwendungs-Assistenten␣generierte␣Anwendungen"));
    LoadStdProfileSettings(4);
    CSingleDocTemplate *pDocTemplate;
    pDocTemplate = new CSingleDocTemplate(IDR_MAINFRAME, RUNTIME_CLASS(CATestDoc),
        RUNTIME_CLASS(CMainFrame), RUNTIME_CLASS(CATestView));
    if (¬pDocTemplate) return FALSE;
    AddDocTemplate(pDocTemplate);
    CCommandLineInfo cmdInfo;
    ParseCommandLine(cmdInfo);
    if (¬ProcessShellCommand(cmdInfo)) return FALSE;
    m_pMainWnd→ShowWindow(SW_SHOW);
    m_pMainWnd→UpdateWindow();
    init_copyright_strings();
    init_special_char_encodings_map();
#if 1    /* 0 */
    Dialog_1 dlg_1;
    dlg_1.DoModal();
#endif
#if 1    /* 0 */
    Dialog_2 dlg_2;
    dlg_2.DoModal();
#endif
    return TRUE;
  }    /* End of CATestApp::InitInstance definition. */
```

**63. OnAppAbout.** [LDF Undated.]

```
⟨ Declare CATestApp functions 59 ⟩ +≡
  afx_msg void OnAppAbout();
```

**64.**

⟨ Define **CATestApp** functions 60 ⟩ +≡
```
void CATestApp::OnAppAbout()
{
  CAboutDlg aboutDlg;
  aboutDlg.DoModal();
}
```

**65. class CAboutDlg declaration.** [LDF Undated.]

⟨ **class CAboutDlg** declaration 65 ⟩ ≡
```
class CAboutDlg : public CDialog {
public: ⟨ Declare public CAboutDlg functions 67 ⟩
  enum {
    IDD = IDD_ABOUTBOX
  };
protected: ⟨ Declare protected CAboutDlg functions 69 ⟩
protected: DECLARE_MESSAGE_MAP()
};
```
This code is used in section 74.

**66. Functions.**

**67. Constructor.**

⟨ Declare **public CAboutDlg** functions 67 ⟩ ≡
```
CAboutDlg();
```
This code is used in section 65.

**68.**

⟨ Define **public CAboutDlg** functions 68 ⟩ ≡
```
CAboutDlg::CAboutDlg()
: CDialog(CAboutDlg::IDD) { }
```
This code is used in section 73.

**69. Do Data Exchange.**

⟨ Declare **protected CAboutDlg** functions 69 ⟩ ≡
```
virtual void DoDataExchange(CDataExchange *pDX);
```
This code is used in section 65.

**70.**

⟨ Define **protected CAboutDlg** functions 70 ⟩ ≡
  **void CAboutDlg** :: *DoDataExchange* (**CDataExchange** $*pDX$)
  {
    **CDialog** :: *DoDataExchange* ($pDX$);
  }

This code is used in section 73.

**71. CAboutDlg message map.** [LDF Undated.]

⟨ **CAboutDlg** message map 71 ⟩ ≡
  BEGIN_MESSAGE_MAP(**CAboutDlg**, **CDialog**)
  END_MESSAGE_MAP( )

This code is used in section 73.

**72. Putting CATestApp and CABoutDlg together.**

**73.** This is what's compiled.

  ⟨ Include files 20 ⟩
  ⟨ Preprocessor macro definitions 18 ⟩
  ⟨ Global variable declarations 55 ⟩
  ⟨ **CATestApp** message map 57 ⟩
  ⟨ Define **CATestApp** functions 60 ⟩
  ⟨ Define **public CAboutDlg** functions 68 ⟩
  ⟨ Define **protected CAboutDlg** functions 70 ⟩
  ⟨ **CAboutDlg** message map 71 ⟩

**74.** This is what's written to `atest.h`.

⟨ `atest.h` 74 ⟩ ≡
  ⟨ Preprocessor macro calls 17 ⟩
**#ifndef** `__AFXWIN_H__`
**#error** `'stdafx.h'␣muss␣vor␣dieser␣Datei␣in␣PCH␣eingeschlossen␣werden.`
**#endif**
**#include** `"resource.h"` /* Hauptsymbole */
  ⟨ **class CAboutDlg** declaration 65 ⟩
  ⟨ **class CATestApp** declaration 54 ⟩
  ⟨ **extern** global variable declarations 56 ⟩

**75. CATestDoc (`atestdoc.web`).** [LDF 2006.10.10.]

**Log**

[LDF 2006.10.10.] Created this file. It contains code formerly in `ATestDoc.h` and `ATestDoc.cpp`.

---

⟨ `atestdoc.web` 75 ⟩ ≡ /* Empty section for use in cross-references. */
This code is cited in sections 6 and 9.
This code is used in section 881.

**76. Include files.**

⟨ Include files 20 ⟩ +≡
**#include** `"stdafx.h"`
**#include** `"ATest.h"`
**#include** `"ATestDoc.h"`
**#ifdef** `_DEBUG`
**#define new** `DEBUG_NEW`
**#endif**

**77. Preprocessor macro calls.**

⟨ Preprocessor macro calls 17 ⟩ +≡
**#pragma** *once*

**78. Preprocessor macro definitions.**

⟨ Preprocessor macro definitions 18 ⟩ +≡

**79. Declare class CATestDoc.** [LDF Undated.]

⟨ Declare **class CATestDoc** 79 ⟩ ≡
  **class CATestDoc** : **public CDocument** {
  **protected**: ⟨ Declare **protected CATestDoc** functions 81 ⟩
  **public**: ⟨ Declare **CATestDoc** functions 83 ⟩
  **protected**: `DECLARE_MESSAGE_MAP`( )
  };
This code is used in section 95.

**80. Functions.** [LDF 2006.10.10.]

**81. Constructor.**

⟨ Declare **protected CATestDoc** functions 81 ⟩ ≡

```
CATestDoc(); DECLARE_DYNCREATE(CATestDoc)
```

This code is used in section 79.

**82.**

⟨ Define **CATestDoc** functions 82 ⟩ ≡

```
CATestDoc::CATestDoc()
{
  return;
}
```

See also sections 84, 86, 88, 90, and 92.

This code is used in section 94.

**83. Destructor.**

⟨ Declare **CATestDoc** functions 83 ⟩ ≡

```
virtual ~CATestDoc();
```

See also sections 85, 87, 89, and 91.

This code is used in section 79.

**84.**

⟨ Define **CATestDoc** functions 82 ⟩ +≡

```
CATestDoc::~CATestDoc()
{
  return;
}
```

**85. On New Document.** [LDF Undated.]

⟨ Declare **CATestDoc** functions 83 ⟩ +≡

```
virtual BOOL OnNewDocument();
```

**86.**

⟨ Define **CATestDoc** functions 82 ⟩ +≡

```
BOOL CATestDoc::OnNewDocument()
{
  if (¬CDocument::OnNewDocument()) return FALSE;
      /* TODO: Hier Code zur Reinitialisierung einfügen */
      /* (SDI-Dokumente verwenden dieses Dokument) */
  return TRUE;
}
```

**87. Serialize.** [LDF Undated.]

⟨ Declare **CATestDoc** functions 83 ⟩ +≡

```
virtual void Serialize(CArchive &ar);
```

**88.**

⟨ Define **CATestDoc** functions 82 ⟩ +≡

```
void CATestDoc::Serialize(CArchive &ar)
{
  if (ar.IsStoring()) {      /* TODO: Hier Code zum Speichern einfügen */
  }
  else {      /* TODO: Hier Code zum Laden einfügen */
  }
}
```

**89. Assert Valid.** [LDF Undated.]

⟨ Declare **CATestDoc** functions 83 ⟩ +≡

```
#ifdef _DEBUG
  virtual void AssertValid() const;
#endif
```

**90.**

⟨ Define **CATestDoc** functions 82 ⟩ +≡

```
#ifdef _DEBUG
  void CATestDoc::AssertValid() const
  {
    CDocument::AssertValid();
  }
#endif
```

**91. Dump.** [LDF Undated.]

⟨ Declare **CATestDoc** functions 83 ⟩ +≡

```
#ifdef _DEBUG
  virtual void Dump(CDumpContext &dc) const;
#endif
```

**92.**

⟨ Define **CATestDoc** functions 82 ⟩ +≡

```
#ifdef _DEBUG
  void CATestDoc::Dump(CDumpContext &dc) const
  {
    CDocument::Dump(dc);
  }
#endif
```

**93. Putting CATestDoc together.**

**94.** This is what's compiled.

```
⟨ Include files 20 ⟩
IMPLEMENT_DYNCREATE(CATestDoc, CDocument)
BEGIN_MESSAGE_MAP(CATestDoc, CDocument)
END_MESSAGE_MAP()
⟨ Define CATestDoc functions 82 ⟩
```

**95.** This is what's written to `atestdoc.h`.

⟨ `atestdoc.h` 95 ⟩ ≡
  ⟨ Preprocessor macro calls 17 ⟩
  ⟨ Declare **class CATestDoc** 79 ⟩

**96. CATestView (`atstview.web`).** [LDF 2006.10.10.]

**Log**

[LDF 2006.10.10.] Created this file. It contains code formerly in `ATestView.h` and `ATestView.cpp`.

---

⟨ `atstview.web` 96 ⟩ ≡ /* Empty section for use in cross-references. */

This code is cited in sections 6 and 9.

This code is used in section 881.

**97. Include files.**

⟨ Include files 20 ⟩ +≡
```
#include "stdafx.h"
#include "ATest.h"
#include "ATestDoc.h"
#include ".\atstview.h"
#include ".\dialog1a.h"
#include ".\dialog2a.h"
```

**98. Preprocessor macro calls.**

⟨ Preprocessor macro calls 17 ⟩ +≡
```
#pragma once
```

**99. Preprocessor macro definitions.**

⟨ Preprocessor macro definitions 18 ⟩ +≡
```
#ifdef _DEBUG
#define new DEBUG_NEW
#endif
```

**100. Declare class CATestView.** [LDF Undated.]

⟨ Declare **class CATestView** 100 ⟩ ≡
```
class CATestView : public CView {
protected: ⟨ Declare protected CATestView functions 102 ⟩
public: ⟨ Declare CATestView functions 104 ⟩
protected: DECLARE_MESSAGE_MAP( )
};
```

This code is used in section 129.

**101. Functions.** [LDF 2006.10.10.]

**102. Constructor.** [LDF Undated.]

⟨ Declare **protected CATestView** functions 102 ⟩ ≡
```
CATestView( ); DECLARE_DYNCREATE(CATestView)
```

See also sections 117, 119, and 121.

This code is used in section 100.

**103.**

⟨ Define **CATestView** functions 103 ⟩ ≡

```
CATestView::CATestView()
{
  return;
}
```

See also sections 105, 107, 109, 111, 113, 115, 118, 120, 122, 124, and 126.

This code is used in section 128.

**104. Destructor.**

⟨ Declare **CATestView** functions 104 ⟩ ≡

```
virtual ~CATestView();
```

See also sections 106, 108, 110, 112, 114, 123, and 125.

This code is used in section 100.

**105.**

⟨ Define **CATestView** functions 103 ⟩ +≡

```
CATestView::~CATestView()
{
  return;
}
```

**106. Get Document.** [LDF Undated.]

⟨ Declare **CATestView** functions 104 ⟩ +≡

```
CATestDoc *GetDocument() const;
```

**107.**

⟨ Define **CATestView** functions 103 ⟩ +≡

```
#ifndef _DEBUG
  inline CATestDoc *CATestView::GetDocument() const
  {
    return reinterpret_cast<CATestDoc *>(m_pDocument);
  }
#endif
#ifdef _DEBUG
  CATestDoc *CATestView::GetDocument() const
  {
    ASSERT(m_pDocument->IsKindOf(RUNTIME_CLASS(CATestDoc)));
    return (CATestDoc *) m_pDocument;
  }
#endif
```

**108. On Draw.** [LDF Undated.]

⟨ Declare **CATestView** functions 104 ⟩ +≡

```
virtual void OnDraw(CDC *pDC);
```

**109.**

⟨ Define **CATestView** functions 103 ⟩ +≡

```
void CATestView::OnDraw(CDC *pDC)
{
  CATestDoc *pDoc = GetDocument();
  ASSERT_VALID(pDoc);
  if (¬pDoc) return;
}
```

**110. Pre-Create Window.** [LDF Undated.]

⟨ Declare **CATestView** functions 104 ⟩ +≡

```
virtual BOOL PreCreateWindow(CREATESTRUCT &cs);
```

**111.**

⟨ Define **CATestView** functions 103 ⟩ +≡

```
BOOL CATestView::PreCreateWindow(CREATESTRUCT &cs)
{
  return CView::PreCreateWindow(cs);
}
```

**112. Assert Valid.** [LDF Undated.]

⟨ Declare **CATestView** functions 104 ⟩ +≡

```
#ifdef _DEBUG
  virtual void AssertValid() const;
#endif
```

**113.**

⟨ Define **CATestView** functions 103 ⟩ +≡

```
#ifdef _DEBUG
  void CATestView::AssertValid() const
  {
    CView::AssertValid();
  }
#endif
```

**114. Dump.**

⟨ Declare **CATestView** functions 104 ⟩ +≡

```
#ifdef _DEBUG
  virtual void Dump(CDumpContext &dc) const;
#endif
```

**115.**

⟨ Define **CATestView** functions 103 ⟩ +≡
```
#ifdef _DEBUG
  void CATestView::Dump(CDumpContext &dc) const
  {
    CView::Dump(dc);
  }
#endif
```

**116. On Prepare Printing.** [LDF Undated.]

**117.**

⟨ Declare **protected CATestView** functions 102 ⟩ +≡
```
  virtual BOOL OnPreparePrinting(CPrintInfo *pInfo);
```

**118.**

⟨ Define **CATestView** functions 103 ⟩ +≡
```
  BOOL CATestView::OnPreparePrinting(CPrintInfo *pInfo)
  {
    return DoPreparePrinting(pInfo);
  }
```

**119. On Begin Printing.** [LDF Undated.]

⟨ Declare **protected CATestView** functions 102 ⟩ +≡
```
  virtual void OnBeginPrinting(CDC *pDC, CPrintInfo *pInfo);
```

**120.**

⟨ Define **CATestView** functions 103 ⟩ +≡
```
  void CATestView::OnBeginPrinting(CDC *pDC, CPrintInfo *pInfo)
  {
#if 0     /* 1 */
#endif
    return;
  }
```

**121. On End Printing.** [LDF Undated.]

⟨ Declare **protected CATestView** functions 102 ⟩ +≡
```
  virtual void OnEndPrinting(CDC *pDC, CPrintInfo *pInfo);
```

**122.**

⟨ Define **CATestView** functions 103 ⟩ +≡
```
  void CATestView::OnEndPrinting(CDC *pDC, CPrintInfo *pInfo)
  {
    return;
  }
```

**123. On Left Button Down.** [LDF Undated.]

⟨ Declare **CATestView** functions 104 ⟩ +≡
```
  afx_msg void OnLButtonDown(UINT nFlags, CPoint point);
```

**124.**

⟨ Define **CATestView** functions 103 ⟩ +≡
  **void CATestView** :: *OnLButtonDown*(**UINT** *nFlags*, **CPoint** *point*)
  {
**#if** 1
    **Dialog_1** *dlg_1*;
    *dlg_1*.*DoModal*( );
**#endif** /* **CView** :: *OnLButtonDown*(*nFlags*, *point*) */
  }

**125. On Right Button Down.** [LDF Undated.]

⟨ Declare **CATestView** functions 104 ⟩ +≡
  **afx_msg void** *OnRButtonDown*(**UINT** *nFlags*, **CPoint** *point*);

**126.**

⟨ Define **CATestView** functions 103 ⟩ +≡
  **void CATestView** :: *OnRButtonDown*(**UINT** *nFlags*, **CPoint** *point*)
  {
**#if** 1
    **Dialog_2** *dlg_2*;
    *dlg_2*.*DoModal*( );
**#endif** /* **CView** :: *OnRButtonDown*(*nFlags*, *point*) */
  }

**127. Putting CATestView together.**

**128.** This is what's compiled.

  ⟨ Include files 20 ⟩
  ⟨ Preprocessor macro definitions 18 ⟩
  `IMPLEMENT_DYNCREATE`(**CATestView**, **CView**)
  `BEGIN_MESSAGE_MAP`(**CATestView**, **CView**)
  `ON_COMMAND`(`ID_FILE_PRINT`, **CView** :: *OnFilePrint*)
  `ON_COMMAND`(`ID_FILE_PRINT_DIRECT`, **CView** :: *OnFilePrint*)
  `ON_COMMAND`(`ID_FILE_PRINT_PREVIEW`, **CView** :: *OnFilePrintPreview*)
  `ON_WM_LBUTTONDOWN`( )
  `ON_WM_MBUTTONDOWN`( )
  `ON_WM_RBUTTONDOWN`( )
  `END_MESSAGE_MAP`( )
  ⟨ Define **CATestView** functions 103 ⟩

**129.** This is what's written to `atstview.h`.

⟨ `atstview.h` 129 ⟩ ≡
⟨ Preprocessor macro calls 17 ⟩
⟨ Declare **class CATestView** 100 ⟩

**130. Dialog_1 (`dialog1a.web`).** [LDF 2006.09.28.]

**Log**

[LDF 2006.08.01.] Changed the name of **CDialog_1** to **Dialog_1**. Made all data members **protected** and all functions **public**, except *DoDataExchange*, which is **protected**.

[LDF 2006.09.28.] Replaced the files `Dialog_1.h`, `Dialog_1.cpp`, and `download_records.cpp` with this file, `dialog1a.web`.

---

⟨ `dialog1a.web` 130 ⟩ ≡ /* Empty section for use in cross-references. */

This code is cited in sections 6 and 9.

This code is used in section 881.

**131. Include files.**

⟨ Include files 20 ⟩ +≡
**#include** "stdafx.h"
**#include** <atltime.h>
**#include** "ATest.h"
**#include** "dialog1a.h"

**132. Preprocessor macro calls.**

⟨ Preprocessor macro calls 17 ⟩ +≡
**#pragma** *once*

**133. class Dialog_1 declaration.**

**Log**

[LDF 2006.09.11.] Added the **protected** variables **int** *day_of_month*, **int** *day_of_week*, **int** *month_of_year*, and **CTime** *t0*. They were formerly static variables, local to `Dialog_1.cpp`.

[LDF 2006.09.11.] Added the **const char** **records_file_name* argument to *download_records*.

---

⟨ **class Dialog_1** declaration 133 ⟩ ≡
**class Dialog_1** : **public CDialog** {
**public**: ⟨ Declare **public Dialog_1** functions 135 ⟩
**enum** {
IDD = IDD_DIALOG1
};
**protected**: ⟨ Declare **protected Dialog_1** functions 139 ⟩
**unsigned short** *timespan*;
**CString** *edit_1_str*;
**unsigned short** *metadata_source*;
**int** *day_of_month*;
**int** *day_of_week*;
**int** *month_of_year*;

```
    CTime t0;
    DECLARE_MESSAGE_MAP()
  };
```

This code is used in section 207.

**134. Functions.** [LDF Undated.]

**135. Constructor.** [LDF Undated.]

⟨ Declare **public Dialog_1** functions 135 ⟩ ≡

```
  DECLARE_DYNAMIC(Dialog_1)
  Dialog_1(CWnd *pParent = NULL);
```

See also sections 137, 141, 148, 150, 152, 154, 156, 158, 160, 162, 164, 166, 168, 170, 172, 174, 176, 178, 180, 188, 190, and 193.

This code is used in section 133.

**136.**

⟨ Define **Dialog_1** functions 136 ⟩ ≡

```
  IMPLEMENT_DYNAMIC(Dialog_1, CDialog)
  Dialog_1::Dialog_1(CWnd *pParent)
  : CDialog(Dialog_1::IDD, pParent), timespan(0), edit_1_str(_T("")), metadata_source(0) {
    return;
  }
```

See also sections 138, 140, 142, 143, 144, 145, 146, 149, 151, 153, 155, 157, 159, 161, 163, 165, 167, 169, 171, 173, 175, 177, 179, 181, 182, 183, 184, 185, 186, 187, 189, 191, 194, 195, 196, 197, 198, 199, 200, 201, 202, 203, and 204.

This code is used in section 206.

**137. Destructor.** [LDF Undated.]

⟨ Declare **public Dialog_1** functions 135 ⟩ +≡

```
  virtual ~Dialog_1(void);
```

**138.**

⟨ Define **Dialog_1** functions 136 ⟩ +≡

```
  Dialog_1::~Dialog_1()
  {
    return;
  }
```

**139. Do Data Exchange.** [LDF Undated.]

⟨ Declare **protected Dialog_1** functions 139 ⟩ ≡

```
  virtual void DoDataExchange(CDataExchange *pDX);
```

This code is used in section 133.

**140.**

⟨ Define **Dialog_1** functions 136 ⟩ +≡

```
void Dialog_1::DoDataExchange(CDataExchange *pDX)
{
  CDialog::DoDataExchange(pDX);
  DDX_Text(pDX, IDC_MESSAGES, edit_1_str);
}
```

**141. Initialize Dialog 1.** [LDF Undated.]

⟨ Declare **public Dialog_1** functions 135 ⟩ +≡

```
virtual BOOL OnInitDialog();
```

**142.**

⟨ Define **Dialog_1** functions 136 ⟩ +≡

```
BOOL Dialog_1::OnInitDialog(){ CDialog::OnInitDialog();
    CButton *pBT = (CButton *) GetDlgItem(IDC_TODAY);
    pBT->SetCheck(BST_CHECKED);
    pBT = NULL;
    pBT = (CButton *) GetDlgItem(IDC_DELETE_OLD_RECORDS);
    pBT->SetCheck(BST_CHECKED);
    pBT = NULL;
#if 0    /* 1 */
#define TIMMS_LDF
#else
#define DBT_LDF
#endif
```

**143.** Testing downloading metadata records from TIMMS. [LDF 2006.08.28.]

⟨ Define **Dialog_1** functions 136 ⟩ +≡

```
#ifdef TIMMS_LDF
  pBT = (CButton *) GetDlgItem(IDC_TIMMS);
  pBT->SetCheck(BST_CHECKED);
  pBT = NULL;
  metadata_source = Metadata_Source::TIMMS;
#else
```

**144.** Testing downloading metadata records from DBT (Digitale Bibliothek Thueringen). [LDF 2006.07.06.] ■

⟨ Define **Dialog_1** functions 136 ⟩ +≡

```
#ifdef DBT_LDF
  pBT = (CButton *) GetDlgItem(IDC_DBT);
  pBT->SetCheck(BST_CHECKED);
  pBT = NULL;
  metadata_source = Metadata_Source::DBT;
#endif
#endif
```

**145.**

⟨ Define **Dialog_1** functions 136 ⟩ +≡

```
timespan = Selector::TODAY;
```

**146.** Return TRUE unless you set the focus to a control. AUSNAHME: OCX-Eigenschaftenseite muss FALSE zurückgeben.

```
⟨ Define Dialog_1 functions 136 ⟩ +≡
  return TRUE; }      /* End of Dialog_1 :: OnInitDialog definition. */
```

**147. Event handlers.**

**Log**

[LDF 2006.09.28.] Added this section.

---

**148. OnBnClickedOk.** [LDF Undated.]

```
⟨ Declare public Dialog_1 functions 135 ⟩ +≡
  afx_msg void OnBnClickedOk( );
```

**149.**

```
⟨ Define Dialog_1 functions 136 ⟩ +≡
  void Dialog_1 :: OnBnClickedOk( )
  {
#if 0      /* 1 */
    exit(0);      /* Ends application. */
#endif
    OnOK( );
  }
```

**150. OnBnClickedToday.** [LDF Undated.]

```
⟨ Declare public Dialog_1 functions 135 ⟩ +≡
  afx_msg void OnBnClickedToday( );
```

**151.**

```
⟨ Define Dialog_1 functions 136 ⟩ +≡
  void Dialog_1 :: OnBnClickedToday( )
  {
    timespan = Selector :: TODAY;
    t0 = CTime :: GetCurrentTime( );
    edit_1_str = "Press␣<Download>␣to␣download␣metadata␣records␣from␣today,␣";
    edit_1_str += t0.Format("%A,␣%B␣%d,␣%Y");
    edit_1_str += ".\r\n";
    UpdateData(FALSE);
  }
```

**152. OnBnClickedYesterday.** [LDF Undated.]

```
⟨ Declare public Dialog_1 functions 135 ⟩ +≡
  afx_msg void OnBnClickedYesterday( );
```

**153.**

⟨ Define **Dialog_1** functions 136 ⟩ +≡

```
void Dialog_1::OnBnClickedYesterday()
{
  timespan = Selector::YESTERDAY;
  t0 = CTime::GetCurrentTime();
  t0 -= CTimeSpan(1, 0, 0, 0);
  edit_1_str = "Press␣<Download>␣to␣download␣metadata␣records␣since␣yesterday,␣";
  edit_1_str += t0.Format("%A,␣%B␣%d,␣%Y");
  edit_1_str += ".\r\n";
  UpdateData(FALSE);
}
```

**154. OnBnClickedThisWeek.** [LDF Undated.]

⟨ Declare **public Dialog_1** functions 135 ⟩ +≡

```
afx_msg void OnBnClickedThisWeek();
```

**155.**

⟨ Define **Dialog_1** functions 136 ⟩ +≡

```
void Dialog_1::OnBnClickedThisWeek()
{
  timespan = Selector::THIS_WEEK;
  t0 = CTime::GetCurrentTime();
  day_of_week = t0.GetDayOfWeek() - 1;
  t0 -= CTimeSpan(day_of_week, 0, 0, 0);
  edit_1_str = "Press␣<Download>␣to␣download␣metadata␣records␣for␣this␣week,␣beginning␣";
  edit_1_str += t0.Format("%A,␣%B␣%d,␣%Y");
  edit_1_str += ".\r\n";
  UpdateData(FALSE);
}
```

**156. OnBnClickedLastWeek.** [LDF Undated.]

⟨ Declare **public Dialog_1** functions 135 ⟩ +≡

```
afx_msg void OnBnClickedLastWeek();
```

**157.**

⟨ Define **Dialog_1** functions 136 ⟩ +≡

```
void Dialog_1::OnBnClickedLastWeek()
{
  timespan = Selector::LAST_WEEK;
  t0 = CTime::GetCurrentTime();
  day_of_week = t0.GetDayOfWeek() - 1;
  t0 -= CTimeSpan(7 + day_of_week, 0, 0, 0);
  edit_1_str = "Press␣<Download>␣to␣download␣metadata␣records␣for␣last␣week,␣beginning␣";
  edit_1_str += t0.Format("%A,␣%B␣%d,␣%Y");
  edit_1_str += ".\r\n";
  UpdateData(FALSE);
}
```

**158. OnBnClickedThisMonth.** [LDF Undated.]

⟨ Declare **public Dialog_1** functions 135 ⟩ +≡

```
afx_msg void OnBnClickedThisMonth();
```

**159.**

⟨ Define **Dialog_1** functions 136 ⟩ +≡

```
void Dialog_1::OnBnClickedThisMonth()
{
  timespan = Selector::THIS_MONTH;
  t0 = CTime::GetCurrentTime();
  day_of_month = t0.GetDay();
  t0 -= CTimeSpan(day_of_month - 1, 0, 0, 0);
  edit_1_str = "Press <Download> to download metadata records ";
  edit_1_str += "for this month, beginning ";
  edit_1_str += t0.Format("%A, %B %d, %Y");
  edit_1_str += ".\r\n";
  UpdateData(FALSE);
}
```

**160. OnBnClickedLastMonth.** [LDF Undated.]

⟨ Declare **public Dialog_1** functions 135 ⟩ +≡

```
afx_msg void OnBnClickedLastMonth();
```

**161.**

⟨ Define **Dialog_1** functions 136 ⟩ +≡

```
void Dialog_1::OnBnClickedLastMonth()
{
  timespan = Selector::LAST_MONTH;
  t0 = CTime::GetCurrentTime();
  day_of_month = t0.GetDay();
  t0 -= CTimeSpan(day_of_month, 0, 0, 0);
  day_of_month = t0.GetDay();
  t0 -= CTimeSpan(day_of_month - 1, 0, 0, 0);
  edit_1_str = "Press <Download> to download metadata records ";
  edit_1_str += "since last month, beginning ";
  edit_1_str += t0.Format("%A, %B %d, %Y");
  edit_1_str += "\r\n.";
  UpdateData(FALSE);
}
```

**162. OnBnClickedLast6Months.** [LDF Undated.]

⟨ Declare **public Dialog_1** functions 135 ⟩ +≡

```
afx_msg void OnBnClickedLast6Months();
```

**163.**

⟨ Define **Dialog_1** functions 136 ⟩ +≡

```
void Dialog_1::OnBnClickedLast6Months()
{
   timespan = Selector::LAST_6_MONTHS;
   t0 = CTime::GetCurrentTime();
   month_of_year = t0.GetMonth();
   if (month_of_year > 5) {
      CTime t1(t0.GetYear(), month_of_year - 5, 1, 0, 0, 0);
      t0 = t1;
   }
   else {
      CTime t1(t0.GetYear() - 1, month_of_year + 7, 1, 0, 0, 0);
      t0 = t1;
   }
   edit_1_str = "Press␣<Download>␣to␣download␣metadata␣records␣";
   edit_1_str += "from␣6␣months␣ago,␣beginning␣";
   edit_1_str += t0.Format("%A,␣%B␣%d,␣%Y");
   edit_1_str += ".\r\n";
   UpdateData(FALSE);
}
```

**164. OnBnClickedThisYear.** [LDF Undated.]

⟨ Declare **public Dialog_1** functions 135 ⟩ +≡

```
afx_msg void OnBnClickedThisYear();
```

**165.**

⟨ Define **Dialog_1** functions 136 ⟩ +≡

```
void Dialog_1::OnBnClickedThisYear()
{
   timespan = Selector::THIS_YEAR;
   t0 = CTime::GetCurrentTime();
   CTime t1(t0.GetYear(), 1, 1, 0, 0, 0);
   t0 = t1;
   edit_1_str = "Press␣<Download>␣to␣download␣metadata␣records␣";
   edit_1_str += "from␣this␣year,␣beginning␣";
   edit_1_str += t0.Format("%A,␣%B␣%d,␣%Y");
   edit_1_str += ".\r\n";
   UpdateData(FALSE);
}
```

**166. OnBnClickedLastYear.** [LDF Undated.]

⟨ Declare **public Dialog_1** functions 135 ⟩ +≡

```
afx_msg void OnBnClickedLastYear();
```

**167.**

⟨ Define **Dialog_1** functions 136 ⟩ +≡

```
void Dialog_1::OnBnClickedLastYear()
{
  timespan = Selector::LAST_YEAR;
  t0 = CTime::GetCurrentTime();
  CTime t1(t0.GetYear() − 1, 1, 1, 0, 0, 0);
  t0 = t1;
  edit_1_str = "Press␣<Download>␣to␣download␣metadata␣records␣";
  edit_1_str += "from␣last␣year,␣beginning␣";
  edit_1_str += t0.Format("%A,␣%B␣%d,␣%Y");
  edit_1_str += ".\r\n";
  UpdateData(FALSE);
}
```

**168. OnBnClickedLast2Years.** [LDF Undated.]

⟨ Declare **public Dialog_1** functions 135 ⟩ +≡

```
afx_msg void OnBnClickedLast2Years();
```

**169.**

⟨ Define **Dialog_1** functions 136 ⟩ +≡

```
void Dialog_1::OnBnClickedLast2Years()
{
  timespan = Selector::LAST_2_YEARS;
  t0 = CTime::GetCurrentTime();
  CTime t1(t0.GetYear() − 2, 1, 1, 0, 0, 0);
  t0 = t1;
  edit_1_str = "Press␣<Download>␣to␣download␣metadata␣records␣";
  edit_1_str += "from␣last␣two␣years,␣beginning␣";
  edit_1_str += t0.Format("%A,␣%B␣%d,␣%Y");
  edit_1_str += ".\r\n";
  UpdateData(FALSE);
}
```

**170. OnBnClickedLast5Years.** [LDF Undated.]

⟨ Declare **public Dialog_1** functions 135 ⟩ +≡

```
afx_msg void OnBnClickedLast5Years();
```

**171.**

⟨ Define **Dialog_1** functions 136 ⟩ +≡

```
void Dialog_1::OnBnClickedLast5Years()
{
    timespan = Selector::LAST_5_YEARS;
    t0 = CTime::GetCurrentTime();
    CTime t1(t0.GetYear() - 5, 1, 1, 0, 0, 0);
    t0 = t1;
    edit_1_str = "Press <Download> to download metadata records from ";
    edit_1_str += "last 5 years, beginning ";
    edit_1_str += t0.Format("%A, %B %d, %Y");
    edit_1_str += ".\r\n";
    UpdateData(FALSE);
}
```

**172. OnBnClickedLast10Years.** [LDF Undated.]

⟨ Declare **public Dialog_1** functions 135 ⟩ +≡

```
afx_msg void OnBnClickedLast10Years();
```

**173.**

⟨ Define **Dialog_1** functions 136 ⟩ +≡

```
void Dialog_1::OnBnClickedLast10Years()
{
    timespan = Selector::LAST_10_YEARS;
    t0 = CTime::GetCurrentTime();
    CTime t1(t0.GetYear() - 10, 1, 1, 0, 0, 0);
    t0 = t1;
    edit_1_str = "Press <Download> to download metadata records ";
    edit_1_str += "from last 10 years, beginning ";
    edit_1_str += t0.Format("%A, %B %d, %Y");
    edit_1_str += ".\r\n";
    UpdateData(FALSE);
}
```

**174. OnBnClickedLast20Years.** [LDF Undated.]

⟨ Declare **public Dialog_1** functions 135 ⟩ +≡

```
afx_msg void OnBnClickedLast20Years();
```

**175.**

⟨ Define **Dialog_1** functions 136 ⟩ +≡

```
void Dialog_1::OnBnClickedLast20Years()
{
  timespan = Selector::LAST_20_YEARS;
  t0 = CTime::GetCurrentTime();
  CTime t1(t0.GetYear() - 20, 1, 1, 0, 0, 0);
  t0 = t1;
  edit_1_str = "Press␣<Download>␣to␣download␣metadata␣records␣";
  edit_1_str += "from␣last␣20␣years,␣beginning␣";
  edit_1_str += t0.Format("%A,␣%B␣%d,␣%Y");
  edit_1_str += ".\r\n";
  UpdateData(FALSE);
}
```

**176. OnBnClickedAllRecords.** [LDF Undated.]

⟨ Declare **public Dialog_1** functions 135 ⟩ +≡

```
afx_msg void OnBnClickedAllRecords();
```

**177.**

⟨ Define **Dialog_1** functions 136 ⟩ +≡

```
void Dialog_1::OnBnClickedAllRecords()
{
  timespan = Selector::ALL_RECORDS;
  edit_1_str = "Press␣<Download>␣to␣download␣all␣metadata␣records.\r\n";
  UpdateData(FALSE);
}
```

**178. OnBnClickedCancel.** [LDF Undated.]

⟨ Declare **public Dialog_1** functions 135 ⟩ +≡

```
afx_msg void OnBnClickedCancel();
```

**179.**

⟨ Define **Dialog_1** functions 136 ⟩ +≡

```
void Dialog_1::OnBnClickedCancel()
{
  OnCancel();
  exit(0);
}
```

**180. OnBnClickedDownload.** [LDF Undated.]

---
**Log**

[LDF 2006.09.11.] Created this file (`oncldnld.cpp`). It contains the function **Dialog_1**::*OnBnClickedDownload*, which was formerly in `Dialog_1.cpp`.

[LDF 2006.10.10.] Moved the definition of this function from `oncldnld.cpp` to this file (`dialog1a.web`).

---

⟨ Declare **public Dialog_1** functions 135 ⟩ +≡

```
afx_msg void OnBnClickedDownload();
```

**181.**

⟨ Define **Dialog_1** functions 136 ⟩ +≡

```
void Dialog_1::OnBnClickedDownload(){ stringstream temp_strm;
    CEdit *result_box = (CEdit *) GetDlgItem(IDC_MESSAGES);
#if 0    /* 1 */
    temp_strm << "Entering 'Dialog_1::OnBnClickedDownload()'.";
    AfxMessageBox(temp_strm.str().c_str());
    temp_strm.str("");
#endif
    if (timespan == Selector::NULL_TIMESPAN) {
      timespan = Selector::TODAY;
      t0 = CTime::GetCurrentTime();
      edit_1_str += "No timespan chosen. ";
      edit_1_str += "Press <Download> to download metadata records from today,\r\n";
      edit_1_str += t0.Format("%A, %B %d, %Y");
      edit_1_str += ", or chose another timespan.\r\n";
      UpdateData(FALSE);
      result_box->UpdateWindow();
      return;
    }    /* if (timespan = Selector::NULL_TIMESPAN) */
    else if (timespan == Selector::TODAY) {
      t0 = CTime::GetCurrentTime();
      edit_1_str += "Downloading metadata records from today,\r\n";
      edit_1_str += t0.Format("%A, %B %d, %Y");
      edit_1_str += "\r\n";
      UpdateData(FALSE);
      result_box->UpdateWindow();
    }    /* if (timespan = Selector::TODAY) */
```

**182.** **CButton**::*GetState* returns a **UINT**, which can be checked against a bit mask. I could have used **static_cast**⟨**BOOL**⟩ in the assignment to *delete_old_records*, but it's six-of-one, half-a-dozen of the other. [LDF 2006.09.11.]

⟨ Define **Dialog_1** functions 136 ⟩ +≡

```
CButton *pBT = (CButton *) GetDlgItem(IDC_DELETE_OLD_RECORDS);
BOOL delete_old_records = (pBT->GetState()) ? TRUE : FALSE;
pBT = NULL;
char *resumption_token;
resumption_token = new char[Metadata_Source::MAX_RESUMPTION_TOKEN];
#define LDF_TESTING 1    /* 0 */
```

**183.** If you want to use this, you'll have to get a new resumption token from the OAI interface, because resumption tokens expire within a few minutes. [LDF 2006.07.05.]

```
⟨ Define Dialog_1 functions 136 ⟩ +≡
#if LDF_TESTING
#if 0    /* 1 */
  strcpy(resumption_token, "13476244132ListRecords2006-07-05.ans:2010:1950");
#endif
  strcpy(resumption_token, "");
#else
  strcpy(resumption_token, "");
#endif
  edit_1_str += "Downloading␣...\r\n";
  UpdateData(FALSE);
  result_box→UpdateWindow( );
  int ret_val = 0;
  edit_1_str += "";
  UpdateData(FALSE);
  result_box→UpdateWindow( );
  Metadata_Source *curr_metadata_source = 0;
  if (metadata_source ≡ Metadata_Source::TIMMS) {
    curr_metadata_source = new Metadata_Source(Metadata_Source::TIMMS);
  }
  else if (metadata_source ≡ Metadata_Source::DBT) {
    curr_metadata_source = new Metadata_Source(Metadata_Source::DBT);
  }
```

**184.** Error handling: *metadata_source* has invalid value. [LDF Undated.]

```
⟨ Define Dialog_1 functions 136 ⟩ +≡
  else {
    temp_strm ≪ "ERROR!␣␣In␣‘Dialog_1::::OnBnClickedDownload()’:" ≪
        endl ≪ "‘metadata_source’␣has␣invalid␣value:␣␣" ≪ metadata_source ≪
        "Exiting␣function␣unsuccessfully␣(no␣return␣value).";
    AfxMessageBox(temp_strm.str( ).c_str( ));
    temp_strm.str("");
    return;
  }    /* else metadata_source has invalid value. */
```

**185.** If *resumption_token* isn't empty, *update_database* is called again. [LDF 2006.07.04.]

⟨ Define **Dialog_1** functions 136 ⟩ +≡

```
  ofstream copy_strm;
  int i = 0;
  stringstream records_file_name_strm; do { ++i;
#if 0    /* 1 */
  temp_strm ≪ "Iteration␣" ≪ i;
  AfxMessageBox(temp_strm.str().c_str());
  temp_strm.str("");
#endif
  records_file_name_strm ≪ "records_" ≪ i ≪ ".xml";
#if 1    /* 0 */
  ret_val = download_records(metadata_source, timespan, &t0, resumption_token,
      records_file_name_strm.str().c_str());
#else
  AfxMessageBox("Testing␣---␣Not␣downloading␣records.");
#endif
  edit_1_str += "‘download_records’␣returned␣";
  temp_strm ≪ ret_val;
  edit_1_str += temp_strm.str().c_str();
  temp_strm.str("");
  edit_1_str += ".\r\n";
  UpdateData(FALSE);
  result_box→UpdateWindow();
  if (ret_val ≠ 0) {
    temp_strm ≪ "ERROR!␣␣In␣‘Dialog_1::OnBnClickedDownload’:" ≪ endl ≪
        "‘Dialog_1::download_records’␣failed,␣returning␣" ≪ ret_val ≪ endl ≪
        "Will␣try␣to␣continue.";
    AfxMessageBox(temp_strm.str().c_str());
    temp_strm.str("");
  }    /* if (ret_val ≠ 0) */
  else if (metadata_source ≡ Metadata_Source::TIMMS ∨ metadata_source ≡ Metadata_Source::DBT)
    {
    strcpy(resumption_token, "");
    ret_val = curr_metadata_source→update_database(delete_old_records, resumption_token,
        records_file_name_strm.str().c_str());
#if 1    /* 0 */
    temp_strm ≪ "curr_metadata_source->update_database()␣returned␣" ≪ ret_val ≪ endl ≪
        "‘resumption_token’␣==␣" ≪ resumption_token;
    AfxMessageBox(temp_strm.str().c_str());
    temp_strm.str("");
#endif
  }    /* else if ( ∨ metadata_source ≡ Metadata_Source::TIMMS metadata_source ≡
        Metadata_Source::DBT ) */
  else    /* ret_val ≡ 0, but metadata_source has an invalid value. */
  {
    return;
  }    /* else (ret_val ≡ 0, but metadata_source has an invalid value.) */
```

**186.** Set *delete_old_records* to `FALSE`. It should only ever be `TRUE` the first time through the loop. [LDF 2006.07.04.]

⟨ Define **Dialog_1** functions 136 ⟩ +≡
```
delete_old_records = FALSE;
edit_1_str = "";
UpdateData(FALSE);
result_box→UpdateWindow();
```

**187.** End of **do** loop. [LDF 2006.09.11.]

⟨ Define **Dialog_1** functions 136 ⟩ +≡
```
  records_file_name_strm.str(""); }   /* do */
#if LDF_TESTING
  while (strlen(resumption_token) > 0) ;
#else
  while (FALSE) ;
#endif
#undef LDF_TESTING
  edit_1_str = "";
  UpdateData(FALSE);
  result_box→UpdateWindow();
  delete curr_metadata_source;
  curr_metadata_source = 0;
  delete[] resumption_token;
  resumption_token = 0;
#if 1   /* 0 */
  temp_strm ≪ "Exiting␣‘Dialog_1::OnBnClickedDownload()’.";
  AfxMessageBox(temp_strm.str().c_str());
  temp_strm.str("");
#endif
  }   /* End of void Dialog_1::OnBnClickedDownload() definition. */
```

**188. OnBnClickedTimms.** [LDF Undated.]

⟨ Declare **public Dialog_1** functions 135 ⟩ +≡
```
afx_msg void OnBnClickedTimms();
```

**189.**

⟨ Define **Dialog_1** functions 136 ⟩ +≡
```
void Dialog_1::OnBnClickedTimms()
{
  metadata_source = Metadata_Source::TIMMS;
}
```

**190. OnBnClickedDbt.** [LDF Undated.]

⟨ Declare **public Dialog_1** functions 135 ⟩ +≡
```
afx_msg void OnBnClickedDbt();
```

**191.**

⟨ Define **Dialog_1** functions 136 ⟩ +≡

```
void Dialog_1::OnBnClickedDbt()
{
  metadata_source = Metadata_Source::DBT;
}
```

**192. Dialog_1 message map.** [LDF Undated.]

⟨ **Dialog_1** message map 192 ⟩ ≡

```
BEGIN_MESSAGE_MAP(Dialog_1, CDialog)
ON_BN_CLICKED(IDOK, OnBnClickedOk)
ON_BN_CLICKED(IDC_TODAY, OnBnClickedToday)
ON_BN_CLICKED(IDC_YESTERDAY, OnBnClickedYesterday)
ON_BN_CLICKED(IDC_THIS_WEEK, OnBnClickedThisWeek)
ON_BN_CLICKED(IDC_LAST_WEEK, OnBnClickedLastWeek)
ON_BN_CLICKED(IDC_THIS_MONTH, OnBnClickedThisMonth)
ON_BN_CLICKED(IDC_LAST_MONTH, OnBnClickedLastMonth)
ON_BN_CLICKED(IDC_LAST_6_MONTHS, OnBnClickedLast6Months)
ON_BN_CLICKED(IDC_THIS_YEAR, OnBnClickedThisYear)
ON_BN_CLICKED(IDC_LAST_YEAR, OnBnClickedLastYear)
ON_BN_CLICKED(IDC_LAST_2_YEARS, OnBnClickedLast2Years)
ON_BN_CLICKED(IDC_LAST_5_YEARS, OnBnClickedLast5Years)
ON_BN_CLICKED(IDC_LAST_10_YEARS, OnBnClickedLast10Years)
ON_BN_CLICKED(IDC_LAST_20_YEARS, OnBnClickedLast20Years)
ON_BN_CLICKED(IDC_ALL_RECORDS, OnBnClickedAllRecords)
ON_BN_CLICKED(IDCANCEL, OnBnClickedCancel)
ON_BN_CLICKED(IDC_DOWNLOAD, OnBnClickedDownload)
ON_BN_CLICKED(IDC_TIMMS, OnBnClickedTimms)
ON_BN_CLICKED(IDC_DBT, OnBnClickedDbt)
END_MESSAGE_MAP()
```

This code is used in section 206.

**193. Download records.** [LDF Undated.]

**Log**

[LDF Undated.] Added this function.

[LDF 2006.09.11.] Added the **const char** *∗records_file_name* argument.

---

⟨ Declare **public Dialog_1** functions 135 ⟩ +≡

```
int download_records(const unsigned short metadata_source,
const unsigned short ttimespan,
const CTime *ctime,
char *resumption_token,
const char *records_file_name);
```

**194.**

⟨ Define **Dialog_1** functions 136 ⟩ +≡

```
int Dialog_1::download_records(const unsigned short metadata_source, const unsigned short
        ttimespan, const CTime *ctime, char *resumption_token, const char *records_file_name){
        DWORD dwAccessType = PRE_CONFIG_INTERNET_ACCESS;
    stringstream message_strm;
    stringstream temp_strm;
    stringstream command_strm;
#if 0    /* 1 */
    temp_strm << "Entering␣`Dialog_1::download_records'.";
    AfxMessageBox(temp_strm.str().c_str());
    temp_strm.str("");
#endif
    CInternetSession session("ATest", dwAccessType);
    CHttpConnection *pServer = NULL;
    CHttpFile *pFile = NULL;
    BOOL b = false;
    DWORD dwServiceType;
    CString strServerName;
    CString strObject;
    INTERNET_PORT nPort;
    CString temp_str;
    UINT ret_val;
    LPCTSTR pszURL;
```

**195.** TIMMS.

```
⟨ Define Dialog_1 functions 136 ⟩ +≡
  if (metadata_source ≡ Metadata_Source::TIMMS) {
#if 0   /* 1 */
    temp_strm ≪ "`metadata_source'␣==␣`Metadata_Source::TIMMS'" ≪ endl ≪
        "`resumption_token'␣==␣" ≪ resumption_token;
    AfxMessageBox(temp_strm.str().c_str());
    temp_strm.str("");
#endif
    edit_1_str += "Downloading␣records␣from␣TIMMS.\r\n";
    temp_strm ≪ "http://oai.uni-tuebingen.de/OAIServer/oai2.aspx?verb=ListRecords";
    if (timespan ≡ Selector::ALL_RECORDS) ;   /* Do nothing. */
    else if (strlen(resumption_token) ≤ 0) {
      temp_strm ≪ "&from=" ≪ ctime→Format("%Y-%m-%d") ≪ "&metadataPrefix=oai_dc";
    }
    if (strlen(resumption_token) > 0) {
      temp_strm ≪ "&resumptionToken=" ≪ resumption_token;
    }
#if 0   /* 1 */
    AfxMessageBox(temp_strm.str().c_str());
#endif
    temp_str = temp_strm.str().c_str();
    pszURL = temp_str;
    edit_1_str += temp_str;
    edit_1_str += "\r\n";
    UpdateData(FALSE);
  }   /* if (metadata_source ≡ Metadata_Source::TIMMS) */
```

**196.**

```
⟨ Define Dialog_1 functions 136 ⟩ +≡
  else
    if (metadata_source ≡ Metadata_Source::DBT) {
#if 0    /* 1 */
      temp_strm ≪ "‘metadata_source’␣==␣‘Metadata_Source::DBT’" ≪ endl ≪
          "‘resumption_token’␣==␣" ≪ resumption_token;
      AfxMessageBox(temp_strm.str().c_str());
      temp_strm.str("");
#endif
      edit_1_str += "Downloading␣records␣from␣DBT.\r\n";
      command_strm ≪ "http://www.db-thueringen.de/servlets/" ≪
          "OAIDataProvider?verb=ListRecords";
      if (timespan ≡ Selector::ALL_RECORDS ∧ strlen(resumption_token) ≡ 0) {
        command_strm ≪ "&metadataPrefix=oai_dc";
      }
      else if (strlen(resumption_token) ≤ 0) {
        command_strm ≪ "&from=" ≪ ctime→Format("%Y-%m-%d") ≪ "&metadataPrefix=oai_dc";
      }
      if (strlen(resumption_token) > 0) {
        command_strm ≪ "&resumptionToken=" ≪ resumption_token;
      }
#if 0
      temp_strm ≪ "Command:␣␣" ≪ command_strm.str();
      AfxMessageBox(temp_strm.str().c_str());
      temp_strm.str("");
#endif
      temp_str = "";
      temp_str += command_strm.str().c_str();
      pszURL = temp_str;
      edit_1_str += temp_str;
      edit_1_str += "\r\n";
      UpdateData(FALSE);
      command_strm.str("");
    }    /* else if (metadata_source ≡ Metadata_Source::DBT) */
```

**197.**

```
⟨ Define Dialog_1 functions 136 ⟩ +≡
  else {
    message_strm ≪ "ERROR!␣␣In␣‘Dialog_1::download_records’:" ≪ endl ≪ "Invalid␣‘metadata_s\
        ource’.␣␣" ≪ "Exiting␣function␣unsuccessfully␣with␣return␣value␣1.";
    AfxMessageBox(message_strm.str().c_str());
    message_strm.str("");
    edit_1_str = "‘Dialog_1::download_records’␣failed:␣␣Invalid␣metadata␣source.";
    UpdateData(FALSE);
    return 1;
  }    /* else (Invalid value for metadata_source. */
  temp_strm.str("");
```

**198.** Parse the URL.

⟨ Define **Dialog_1** functions 136 ⟩ +≡

```
b = AfxParseURL(pszURL, dwServiceType, strServerName, strObject, nPort);
```

**199.** Error handling: *AfxParseURL* failed. [LDF 2006.06.01.]

⟨ Define **Dialog_1** functions 136 ⟩ +≡

```
if (¬b) {
  message_strm ≪ "ERROR!␣␣In␣‘download_records’:" ≪ endl ≪ "dwServiceType␣==␣" ≪
      dwServiceType ≪ endl ≪ "strServerName␣==␣" ≪ strServerName ≪ endl ≪
      "strObject␣==␣" ≪ strObject ≪ endl ≪ "nPort␣==␣" ≪ nPort;
  AfxMessageBox(message_strm.str().c_str());
  message_strm.str("");
}    /* ¬b (AfxParseURL failed. LDF 2006.06.01. */
```

**200.** *AfxParseURL* succeeded. [LDF 2006.06.01.]

⟨ Define **Dialog_1** functions 136 ⟩ +≡

```
#if 0
  else {
    message_strm ≪ "In␣‘download_records’:" ≪ "‘AfxParseURL’␣succeeded." ≪ endl ≪
        "dwServiceType␣==␣" ≪ dwServiceType ≪ endl ≪ "strServerName␣==␣" ≪ strServerName ≪
        endl ≪ "strObject␣==␣" ≪ strObject ≪ endl ≪ "nPort␣==␣" ≪ nPort;
    AfxMessageBox(message_strm.str().c_str());
    message_strm.str("");
  }   /* b ≡ true (AfxParseURL succeeded. LDF 2006.06.01. */
#endif
  pServer = session.GetHttpConnection(strServerName, nPort);
  if (pServer ≡ 0) {
    AfxMessageBox("session.GetHttpConnection␣failed.");
    message_strm.str("");
  }
#if 0   /* 1 */
  else   /* pServer ≠ 0 */
  {
    AfxMessageBox("session.GetHttpConnection␣succeeded.");
    message_strm.str("");
  }   /* else (pServer ≠ 0) */
#endif
  DWORD dwHttpRequestFlags = INTERNET_FLAG_EXISTING_CONNECT |
      INTERNET_FLAG_NO_AUTO_REDIRECT;
  const TCHAR szHeaders[] = _T("Accept:␣text/*\r\nUser-Agent:␣DBTest\r\n");
  pFile = pServer→OpenRequest(CHttpConnection::HTTP_VERB_GET, strObject, NULL, 1, NULL, NULL,
      dwHttpRequestFlags);
  pFile→AddRequestHeaders(szHeaders);
  pFile→SendRequest();
  DWORD dwRet;
  pFile→QueryInfoStatusCode(dwRet);
#if 0   /* 1 */
  message_strm ≪ "Result␣of␣‘pFile->QueryInfoStatusCode’␣==␣" ≪ dwRet;
  AfxMessageBox(message_strm.str().c_str());
  message_strm.str("");
#endif
  if (dwRet ≡ HTTP_STATUS_DENIED) {
    message_strm ≪ "ERROR!␣␣In␣‘download_records’:" ≪ endl ≪
        "HTTP␣request␣failed.␣␣Status␣code␣==␣HTTP_STATUS_DENIED." ≪ endl ≪
        "Exiting␣function␣with␣return␣value␣1.";
    AfxMessageBox(message_strm.str().c_str());
    message_strm.str("");
    session.Close();
    return 1;
  }   /* if (dwRet ≡ HTTP_STATUS_DENIED) */
```

**201.** Read text from *pFile* into the character buffer *buff* and write it to **ofstream** *out_strm*, which is attached to the file whose name *records_file_name*. [LDF 2006.06.01.] [LDF 2006.09.11.]

⟨ Define **Dialog_1** functions 136 ⟩ +≡

```
  unsigned int BUFF_SIZE;
  BUFF_SIZE = 1048576;   /* 1 MB */   /* BUFF_SIZE ≡ 8000 */
  char *buff = new char[BUFF_SIZE];
  ofstream out_strm;
  out_strm.open(records_file_name);
  ULONGLONG file_length;
  file_length = pFile→GetLength();
#if 0
  temp_strm ≪ "file_length␣==␣" ≪ file_length;
  AfxMessageBox(temp_strm.str().c_str());
  temp_strm.str("");
#endif
  int i = 1;
  ret_val = 0;
```

**202.**

⟨ Define **Dialog_1** functions 136 ⟩ +≡

```
  do
  {
#if 0
    temp_strm ≪ "Iteration␣" ≪ i++ ≪ endl ≪ "ret_val␣==␣" ≪ ret_val;
    AfxMessageBox(temp_strm.str().c_str());
    temp_strm.str("");
#endif
    ret_val = pFile→Read(buff, BUFF_SIZE);
    if (ret_val > 0) {
      out_strm.write(buff, ret_val);
    }
#if 0
    message_strm ≪ "Characters␣read␣(‘ret_val’)␣==␣" ≪ ret_val ≪ endl;
#if 0   /* 1 */
    message_strm ≪ buff;
#endif
    AfxMessageBox(message_strm.str().c_str());
    message_strm.str("");
#endif
  }   /* do */
```

**203.**

---

**Log**

[LDF 2006.07.06.] !! BUG FIX: Changed conditional from *ret_val* ≡ BUFF_SIZE to *ret_val* $> 0$. The way it was before worked for TIMMS, but failed for DBT.

---

⟨ Define **Dialog_1** functions 136 ⟩ +≡

```
  while (ret_val > 0) ;
```

**204.** Exit *download_records* successfully with return value 0.

```
⟨ Define Dialog_1 functions 136 ⟩ +≡
  out_strm.close( );
  delete[ ] buff;
  session.Close( );
#if 1   /* 0 */
  temp_strm << "Exiting␣‘Dialog_1::download_records’.";
  AfxMessageBox(temp_strm.str( ).c_str( ));
  temp_strm.str("");
#endif
  return 0; }   /* End of Dialog_1 :: download_records definition. */
```

**205. Putting Dialog_1 together.**

**206.** This is what's compiled.

```
  ⟨ Include files 20 ⟩
  ⟨ Dialog_1 message map 192 ⟩
  ⟨ Define Dialog_1 functions 136 ⟩
```

**207.** This is what's written to `dialog1a.h`.

⟨ `dialog1a.h` 207 ⟩ ≡
⟨ Preprocessor macro calls 17 ⟩
**#include** "afxwin.h"
⟨ **class Dialog_1** declaration 133 ⟩

**208. Dialog_2 (`dialog2a.web`).** [LDF 2006.09.28.]

**Log**

[LDF 2006.09.28.] Replaced the files `Dialog_2.h`, `Dialog_2.cpp`, with this file, `dialog2a.web`.

---

⟨ `dialog2a.web` 208 ⟩ ≡ /* Empty section for use in cross-references. */

This code is cited in sections 6 and 9.

This code is used in section 881.

**209. Include files.**

⟨ Include files 20 ⟩ +≡
**#include** "stdafx.h"
**#include** "ATest.h"
**#include** "dialog2a.h"

**210. Preprocessor macro calls.**

⟨ Preprocessor macro calls 17 ⟩ +≡
**#pragma** *once*

**211. Preprocessor macro definitions.**

⟨ Preprocessor macro definitions 18 ⟩ +≡

**212. class Dialog_2 declaration.** [LDF Undated.]

**Log**

[LDF 2006.08.01.] Made all **CString** *results_str* **private**.

---

⟨ **class Dialog_2** declaration 212 ⟩ ≡
**class Dialog_2** : **public CDialog** {
DECLARE_DYNAMIC(**Dialog_2**)
**private**: **CString** *results_str*;
**CString** *search_str*;
**unsigned int** *select_value*;
**BOOL** *ignore_case*;
**unsigned int** *search_options*;
**unsigned short** *timespan*;
**unsigned short** *sort_order*;
**unsigned short** *sort_field*;
**unsigned short** *use_date_type*;
**BOOL** *suppress_duplicate_records*;
**public**: ⟨ Declare **public Dialog_2** functions 216 ⟩
**enum** {
IDD = IDD_DIALOG2

```
    };
  protected: ⟨ Declare protected Dialog_2 functions 227 ⟩
    DECLARE_MESSAGE_MAP( )
  };
```

This code is used in section 293.

**213. Global variable declarations.** [LDF Undated.]

⟨ Global variable declarations 55 ⟩ +≡

**214. extern global variable declarations.** [LDF Undated.]

⟨ **extern** global variable declarations 56 ⟩ +≡

**215. Functions.** [LDF Undated.]

**216. Constructor.** [LDF Undated.]

⟨ Declare **public Dialog_2** functions 216 ⟩ ≡

```
  Dialog_2(CWnd *pParent = NULL);
```

See also sections 218, 220, 230, 232, 245, 247, 249, 251, 253, 255, 257, 259, 261, 263, 265, 267, 269, 274, 276, 278, 280, 282, 284, 286, and 288.

This code is used in section 212.

**217.**

⟨ Define **Dialog_2** functions 217 ⟩ ≡

```
  IMPLEMENT_DYNAMIC(Dialog_2, CDialog)Dialog_2::Dialog_2(CWnd *pParent)      /* = NULL */
  : CDialog(Dialog_2::IDD, pParent)
  , results_str(_T(""))
  , search_str(_T(""))
  , select_value(0)
  , ignore_case(TRUE)
  , search_options(Selector::BEG_OR_WHOLE_WORD | Selector::IGNORE_CASE)
  , timespan(0)
  , sort_order(0)
  , sort_field(0)
  , use_date_type(Selector::USE_DC_DATE)
  , suppress_duplicate_records(FALSE)
  { }
```

See also sections 219, 221, 222, 223, 224, 225, 226, 228, 231, 233, 234, 235, 236, 237, 238, 239, 240, 241, 242, 243, 244, 246, 248, 250, 252, 254, 256, 258, 260, 262, 264, 266, 268, 270, 271, 272, 273, 275, 277, 279, 281, 283, 285, 287, and 289.

This code is used in section 292.

**218. Destructor.** [LDF Undated.]

⟨ Declare **public Dialog_2** functions 216 ⟩ +≡

```
  virtual ~Dialog_2( );
```

**219.**

⟨ Define **Dialog_2** functions 217 ⟩ +≡
**Dialog_2** :: ~**Dialog_2**( )
{ }

**220. Initialize Dialog 2.**

⟨ Declare **public Dialog_2** functions 216 ⟩ +≡
**virtual BOOL** *OnInitDialog*( );

**221.**

⟨ Define **Dialog_2** functions 217 ⟩ +≡
**BOOL Dialog_2** :: *OnInitDialog*( ){ **CDialog** :: *OnInitDialog*( );

**222.** Enter table names and groups of table names in selection edit box. [LDF Undated.]

⟨ Define **Dialog_2** functions 217 ⟩ +≡

```
CListBox *pLB = (CListBox *) GetDlgItem(IDC_SELECT);
pLB→InsertString(−1, "Titles, Subjects, Descriptions");
pLB→InsertString(−1, "Creators, Contributors");
pLB→InsertString(−1, "All Fields");
pLB→InsertString(−1, "Titles");
pLB→InsertString(−1, "Subjects");
pLB→InsertString(−1, "Descriptions");
pLB→InsertString(−1, "Creators");
pLB→InsertString(−1, "Contributors");
pLB→InsertString(−1, "Types");
pLB→InsertString(−1, "Publishers");
pLB→InsertString(−1, "Languages");
pLB→InsertString(−1, "Rights");
pLB→InsertString(−1, "Identifiers");
pLB→SetSel(0, TRUE);
pLB = NULL;
CButton *pBT = (CButton *) GetDlgItem(IDC_BEG_OR_WHOLE_WORD);
pBT→SetCheck(BST_CHECKED);
pBT = NULL;
```

**223.** "List All Records" group. [LDF Undated.]

⟨ Define **Dialog_2** functions 217 ⟩ +≡

```
pLB = (CListBox *) GetDlgItem(IDC_SORT_BY);
pLB→InsertString(−1, "Record Number");
pLB→InsertString(−1, "Creators");
pLB→InsertString(−1, "Titles");
pLB→InsertString(−1, "Date (dc:date)");
pLB→InsertString(−1, "Date (header datestamp)");
pLB→SetCurSel(0);
pBT = (CButton *) GetDlgItem(IDC_ALL_DATES);
pBT→SetCheck(BST_CHECKED);
pBT = NULL;
timespan = Selector :: ALL_RECORDS;
sort_order = Selector :: SORT_ASCENDING;
```

**224.**

⟨ Define **Dialog_2** functions 217 ⟩ +≡

```
pBT = (CButton *) GetDlgItem(IDC_USE_DC_DATE);
pBT→SetCheck(BST_CHECKED);
pBT = NULL;
pBT = (CButton *) GetDlgItem(IDC_NO_DUPLICATES);
pBT→SetCheck(BST_CHECKED);
pBT = NULL;
```

**225.** If I change the tab sequence in the dialog box, I'll have to call *SetFocus* and have this function return 0. Otherwise, if this function doesn't return 0, the first item in the dialog box has the input focus. [LDF 2006.06.20.]

⟨ Define **Dialog_2** functions 217 ⟩ +≡

```
#if 0
  CEdit *pEB = (CEdit *) GetDlgItem(IDC_SEARCH_STRING);
  pEB→SetFocus( );
#endif
```

**226.** Return TRUE unless you set the focus to a control. AUSNAHME: OCX-Eigenschaftenseite muss FALSE zurückgeben.

⟨ Define **Dialog_2** functions 217 ⟩ +≡

```
#if 0    /* 1 */
  return TRUE;
#endif
  return CDialog::OnInitDialog( ); }    /* End of Dialog_2::OnInitDialog definition. */
```

**227. Do Data Exchange.** [LDF Undated.]

⟨ Declare **protected Dialog_2** functions 227 ⟩ ≡

```
virtual void DoDataExchange(CDataExchange *pDX);
```

This code is used in section 212.

**228.**

⟨ Define **Dialog_2** functions 217 ⟩ +≡

```
void Dialog_2::DoDataExchange(CDataExchange *pDX)
{
  CDialog::DoDataExchange(pDX);
  DDX_Text(pDX, IDC_RESULTS, results_str);
  DDX_Text(pDX, IDC_SEARCH_STRING, search_str);
  DDX_Check(pDX, IDC_CASE_IGNORE, ignore_case);
}
```

**229. Event handlers.**

**Log**

[LDF 2006.09.28.] Added this section.

**230. OnBnClickedOk.** [LDF Undated.]

⟨ Declare **public Dialog_2** functions 216 ⟩ +≡

```
afx_msg void OnBnClickedOk( );
```

**231.**

⟨ Define **Dialog_2** functions 217 ⟩ +≡

```
void Dialog_2::OnBnClickedOk()
{
#if 0    /* 1 */
    OnOK();
#endif
    exit(0);
}
```

**232. OnBnClickedSearch.** [LDF Undated.]

**Log**

[LDF 2006.10.09.] !! BUG FIX: Now declaring **stringstream** *temp_strm*.

---

⟨ Declare **public Dialog_2** functions 216 ⟩ +≡

```
afx_msg void OnBnClickedSearch();
```

**233.**

⟨ Define **Dialog_2** functions 217 ⟩ +≡

```
void Dialog_2::OnBnClickedSearch(){ stringstream temp_strm;
    CEdit *pEB = (CEdit *) GetDlgItem(IDC_SEARCH_STRING);
    CListBox *pLB = (CListBox *) GetDlgItem(IDC_SELECT);
    while (TRUE) {
      pEB->GetWindowText(search_str);
      if (search_str ≡ "") {
        AfxMessageBox("Please enter a search string.");
        return;
      }
      else break;
    }    /* while */
#if 0
    temp_strm << "search_options == " << search_options << endl << "search_str == " << search_str;
    AfxMessageBox(temp_strm.str().c_str());
    temp_strm.str("");
#endif
#if 0
    temp_strm << "search_str == " << search_str;
    AfxMessageBox(temp_strm.str().c_str());
    temp_strm.str("");
#endif
    int buff[16];
    int items_ctr;
    items_ctr = pLB->GetSelItems(16, buff);
```

**234.**

⟨ Define **Dialog_2** functions 217 ⟩ +≡

```
if (items_ctr ≡ LB_ERR) {
    temp_strm ≪ "ERROR!␣␣In␣‘Dialog_2::OnBnClickedSearch’:" ≪ endl ≪
        "Invalid␣search␣field,␣‘items_ctr’␣==␣" ≪ items_ctr ≪ endl ≪
        "Exiting␣function␣unsuccessfully␣(no␣return␣value).";
    AfxMessageBox(temp_strm.str( ).c_str( ));
    temp_strm.str("");
    return;
}
```

**235.**

⟨ Define **Dialog_2** functions 217 ⟩ +≡

```
else
    if (items_ctr ≡ 0) {
        temp_strm ≪ "WARNING!␣␣In␣‘Dialog_2::OnBnClickedSearch’:" ≪ endl ≪
            "No␣search␣field␣selected,␣‘items_ctr’␣==␣" ≪ items_ctr ≪ endl ≪
            "Please␣pick␣a␣search␣field.";
        AfxMessageBox(temp_strm.str( ).c_str( ));
        temp_strm.str("");
    }
```

**236.**

⟨ Define **Dialog_2** functions 217 ⟩ +≡

```
else    /* ¬(items_ctr ≡ LB_ERR ∨ items_ctr ≡ 0) */
{ select_value = 0;
CString selection_str; for (int i = 0; i < items_ctr; ++i) { pLB→GetText(buff[i], selection_str);
#if 0
temp_strm ≪ "buff[" ≪ i ≪ "]␣==␣" ≪ buff[i] ≪ endl ≪ "selection_str␣==␣" ≪ selection_str ≪
    endl;
#endif
if (buff[i] ≡ 0)    /* Titles, Subjects, Descriptions */
{
    select_value |= Selector::DESCRIPTIONS | Selector::SUBJECTS | Selector::TITLES;
#if 0
    temp_strm ≪ "Selected␣\"Titles,␣Subjects,␣Descriptions\"." ≪ endl;
#endif
}    /* if */
```

**237.**

⟨ Define **Dialog_2** functions 217 ⟩ +≡

```
else
    if (buff[i] ≡ 1)    /* */
    {
        select_value |= Selector::CREATORS | Selector::CONTRIBUTORS;
#if 0
        temp_strm ≪ "Selected␣\"Creators,␣Contributors\"." ≪ endl;
#endif
    }    /* else if */
```

**238.**

```
⟨ Define Dialog_2 functions 217 ⟩ +≡
  else
    if (buff[i] ≡ 2)    /* All Fields */
    {
      select_value = Selector::CONTRIBUTORS | Selector::CREATORS | Selector::DESCRIPTIONS |
          Selector::IDENTIFIERS | Selector::LANGUAGES | Selector::PUBLISHERS | Selector::RIGHTS |
          Selector::SUBJECTS | Selector::TITLES | Selector::TYPES;
#if 0
      temp_strm ≪ "Selected␣\"All␣Fields\".␣" ≪ "Breaking." ≪ endl;
#endif
      break;
    }    /* else if */
```

**239.**

```
⟨ Define Dialog_2 functions 217 ⟩ +≡
  else
    if (selection_str ≡ "Contributors") {
      select_value |= Selector::CONTRIBUTORS;
    }
    else if (selection_str ≡ "Creators") {
      select_value |= Selector::CREATORS;
    }
    else if (selection_str ≡ "Descriptions") {
      select_value |= Selector::DESCRIPTIONS;
    }
    else if (selection_str ≡ "Identifiers") {
      select_value |= Selector::IDENTIFIERS;
    }
    else if (selection_str ≡ "Languages") {
      select_value |= Selector::LANGUAGES;
    }
    else if (selection_str ≡ "Publishers") {
      select_value |= Selector::PUBLISHERS;
    }
    else if (selection_str ≡ "Rights") {
      select_value |= Selector::RIGHTS;
    }
    else if (selection_str ≡ "Subjects") {
      select_value |= Selector::SUBJECTS;
    }
    else if (selection_str ≡ "Titles") {
      select_value |= Selector::TITLES;
    }
    else if (selection_str ≡ "Types") {
      select_value |= Selector::TYPES;
    }
  }    /* for */
```

**240.**

⟨ Define **Dialog_2** functions 217 ⟩ +≡

```
#if 0
  temp_strm << "select_value == " << select_value;
  AfxMessageBox(temp_strm.str().c_str());
  temp_strm.str("");
#endif
  temp_strm.str("");
  CEdit *result_box = (CEdit *) GetDlgItem(IDC_RESULTS);
  results_str = "Searching database ...";
  UpdateData(FALSE);
  result_box->UpdateWindow();
  CString result_str;
  int ret_val;
```

**241.**

⟨ Define **Dialog_2** functions 217 ⟩ +≡

```
  Selector selector;
  selector.use_date_type = use_date_type;
```

**242.** Error handling: **Selector** constructor failed. [LDF Undated.]

⟨ Define **Dialog_2** functions 217 ⟩ +≡

```
  if (errno ≠ 0) {
    temp_strm << "ERROR!  In 'Dialog_2::OnBnClickedSearch': " <<
        "Failed to create a 'Selector' object." << endl << "Can't search database.  " <<
        "Exiting function unsuccessfully.";
    AfxMessageBox(temp_strm.str().c_str());
    temp_strm.str("");
    return;
  }    /* if (errno ≠ 0) */
```

**243.**

⟨ Define **Dialog_2** functions 217 ⟩ +≡

```
  ret_val = selector.select_from_database(select_value, search_str, search_options);
#if 0
  temp_strm << "'Selector::select_from_database' returned " << ret_val;
  AfxMessageBox(temp_strm.str().c_str());
  temp_strm.str("");
#endif
  temp_strm << "Finished search. " << ret_val << " record";
  if (ret_val > 1) temp_strm << "s";
  temp_strm << " found.";
  results_str = temp_strm.str().c_str();
  UpdateData(FALSE);
  result_box->UpdateWindow();
  UpdateData(FALSE);
  temp_strm.str(""); }    /* else (nIndex ≠ LB_ERR) */
```

**244.**

⟨ Define **Dialog_2** functions 217 ⟩ +≡
```
}   /* End of Dialog_2::OnBnClickedSearch definition. */
```

**245. OnBnClickedBegOrWholeWord.** [LDF Undated.]

⟨ Declare **public Dialog_2** functions 216 ⟩ +≡
```
afx_msg void OnBnClickedBegOrWholeWord();
```

**246.**

⟨ Define **Dialog_2** functions 217 ⟩ +≡
```
void Dialog_2::OnBnClickedBegOrWholeWord()
{
  search_options |= Selector::BEG_OR_WHOLE_WORD;
  search_options &= ~Selector::WHOLE_WORD_ONLY;
  search_options &= ~Selector::WHOLE_OR_PARTIAL_WORD;
  search_options &= ~Selector::EXACT_MATCH;
}
```

**247. OnBnClickedWholeWordOnly.** [LDF Undated.]

⟨ Declare **public Dialog_2** functions 216 ⟩ +≡
```
afx_msg void OnBnClickedWholeWordOnly();
```

**248.**

⟨ Define **Dialog_2** functions 217 ⟩ +≡
```
void Dialog_2::OnBnClickedWholeWordOnly()
{
  search_options |= Selector::WHOLE_WORD_ONLY;
  search_options &= ~Selector::BEG_OR_WHOLE_WORD;
  search_options &= ~Selector::WHOLE_OR_PARTIAL_WORD;
  search_options &= ~Selector::EXACT_MATCH;
}
```

**249. OnBnClickedWholeOrPartialWord.** [LDF Undated.]

⟨ Declare **public Dialog_2** functions 216 ⟩ +≡
```
afx_msg void OnBnClickedWholeOrPartialWord();
```

**250.**

⟨ Define **Dialog_2** functions 217 ⟩ +≡
```
void Dialog_2::OnBnClickedWholeOrPartialWord()
{
  search_options |= Selector::WHOLE_OR_PARTIAL_WORD;
  search_options &= ~Selector::BEG_OR_WHOLE_WORD;
  search_options &= ~Selector::WHOLE_WORD_ONLY;
  search_options &= ~Selector::EXACT_MATCH;
}
```

**251. OnBnClickedCaseIgnore.** [LDF Undated.]

⟨ Declare **public Dialog_2** functions 216 ⟩ +≡
```
afx_msg void OnBnClickedCaseIgnore();
```

**252.**

⟨ Define **Dialog_2** functions 217 ⟩ +≡

```
void Dialog_2::OnBnClickedCaseIgnore()
{
   ignore_case = ¬ignore_case;
   if (ignore_case) search_options |= Selector::IGNORE_CASE;
   else search_options &= ~Selector::IGNORE_CASE;
}
```

**253. OnBnClickedExactMatch.** [LDF Undated.]

⟨ Declare **public Dialog_2** functions 216 ⟩ +≡

```
afx_msg void OnBnClickedExactMatch();
```

**254.**

⟨ Define **Dialog_2** functions 217 ⟩ +≡

```
void Dialog_2::OnBnClickedExactMatch()
{
   search_options |= Selector::EXACT_MATCH;
   search_options &= ~Selector::WHOLE_OR_PARTIAL_WORD;
   search_options &= ~Selector::BEG_OR_WHOLE_WORD;
   search_options &= ~Selector::WHOLE_WORD_ONLY;
}
```

**255. OnBnClickedAllDates.** [LDF Undated.]

⟨ Declare **public Dialog_2** functions 216 ⟩ +≡

```
afx_msg void OnBnClickedAllDates();
```

**256.**

⟨ Define **Dialog_2** functions 217 ⟩ +≡

```
void Dialog_2::OnBnClickedAllDates()
{
   timespan = Selector::ALL_RECORDS;
}
```

**257. OnBnClickedSinceLastYear.** [LDF Undated.]

⟨ Declare **public Dialog_2** functions 216 ⟩ +≡

```
afx_msg void OnBnClickedSinceLastYear();
```

**258.**

⟨ Define **Dialog_2** functions 217 ⟩ +≡

```
void Dialog_2::OnBnClickedSinceLastYear()
{
   timespan = Selector::LAST_YEAR;
}
```

**259. OnBnClickedThisYear.** [LDF Undated.]

⟨ Declare **public Dialog_2** functions 216 ⟩ +≡

```
afx_msg void OnBnClickedThisYear();
```

**260.**

⟨ Define **Dialog_2** functions 217 ⟩ +≡
```
void Dialog_2::OnBnClickedThisYear()
{
  timespan = Selector::THIS_YEAR;
}
```

**261. OnBnClickedLast6Months.** [LDF Undated.]

⟨ Declare **public Dialog_2** functions 216 ⟩ +≡
```
afx_msg void OnBnClickedLast6Months();
```

**262.**

⟨ Define **Dialog_2** functions 217 ⟩ +≡
```
void Dialog_2::OnBnClickedLast6Months()
{
  timespan = Selector::LAST_6_MONTHS;
}
```

**263. OnBnClickedLastMonth.** [LDF Undated.]

⟨ Declare **public Dialog_2** functions 216 ⟩ +≡
```
afx_msg void OnBnClickedLastMonth();
```

**264.**

⟨ Define **Dialog_2** functions 217 ⟩ +≡
```
void Dialog_2::OnBnClickedLastMonth()
{
  timespan = Selector::LAST_MONTH;
}
```

**265. OnBnClickedThisMonth.** [LDF Undated.]

⟨ Declare **public Dialog_2** functions 216 ⟩ +≡
```
afx_msg void OnBnClickedThisMonth();
```

**266.**

⟨ Define **Dialog_2** functions 217 ⟩ +≡
```
void Dialog_2::OnBnClickedThisMonth()
{
  timespan = Selector::THIS_MONTH;
}
```

**267. OnBnClickedThisWeek.** [LDF Undated.]

⟨ Declare **public Dialog_2** functions 216 ⟩ +≡
```
afx_msg void OnBnClickedThisWeek();
```

**268.**

⟨ Define **Dialog_2** functions 217 ⟩ +≡
```
void Dialog_2::OnBnClickedThisWeek()
{
  timespan = Selector::THIS_WEEK;
}
```

**269. OnBnClickedListRecords.** [LDF Undated.]

**Log**

[LDF 2006.10.09.] !! BUG FIX: Now declaring **stringstream** *temp_strm*.

⟨ Declare **public Dialog_2** functions 216 ⟩ +≡
```
afx_msg void OnBnClickedListRecords( );
```

**270.**

⟨ Define **Dialog_2** functions 217 ⟩ +≡
```
void Dialog_2::OnBnClickedListRecords( ){ stringstream temp_strm;
    CListBox *sort_box = (CListBox *) GetDlgItem(IDC_SORT_BY);
    CEdit *result_box = (CEdit *) GetDlgItem(IDC_RESULTS);
    sort_field = sort_box→GetCurSel( );
    if (sort_field ≡ LB_ERR) {
      temp_strm ≪ "ERROR!␣␣In␣‘Dialog_2::OnBnClickedListRecords’:" ≪
          "‘sort_box->GetCurSel()’␣failed." ≪ endl ≪
          "Setting␣‘sort_field’␣=␣‘Selector::SORT_FIELD_RECORD_ID’␣" ≪ "and␣continuing.";
      AfxMessageBox(temp_strm.str( ).c_str( ));
      temp_strm.str("");
      sort_field = Selector::SORT_FIELD_RECORD_ID;
    }
    else if (sort_field ≡ 0) sort_field = Selector::SORT_FIELD_RECORD_ID;
    else if (sort_field ≡ 1) sort_field = Selector::SORT_FIELD_CREATOR;
    else if (sort_field ≡ 2) sort_field = Selector::SORT_FIELD_TITLE;
    else if (sort_field ≡ 3) sort_field = Selector::SORT_FIELD_DC_DATE;
    else if (sort_field ≡ 4) sort_field = Selector::SORT_FIELD_HEADER_DATESTAMP;
    else {
      temp_strm ≪ "ERROR!␣␣In␣‘Dialog_2::OnBnClickedListRecords’:" ≪
          "‘sort_field’␣has␣invalid␣value:␣" ≪ sort_field ≪ endl ≪
          "Setting␣‘sort_field’␣=␣‘Selector::SORT_FIELD_RECORD_ID’␣" ≪ "and␣continuing.";
      AfxMessageBox(temp_strm.str( ).c_str( ));
      temp_strm.str("");
      sort_field = Selector::SORT_FIELD_RECORD_ID;
    }
    int ret_val;
```

**271.**

⟨ Define **Dialog_2** functions 217 ⟩ +≡
```
Selector selector;
selector.use_date_type = use_date_type;
```

**272.** Error handling: **Selector** constructor failed. [LDF Undated.]

⟨ Define **Dialog_2** functions 217 ⟩ +≡

```
if (errno ≠ 0) {
   temp_strm ≪ "ERROR!␣␣In␣‘Dialog_2::OnBnClickedListRecords’:␣" ≪
       "Failed␣to␣create␣a␣‘Selector’␣object." ≪ endl ≪
       "Can’t␣search␣database.␣␣Exiting␣function␣unsuccessfully.";
   AfxMessageBox(temp_strm.str().c_str());
   temp_strm.str("");
   return;
}   /* if (errno ≠ 0) */
```

**273.**

⟨ Define **Dialog_2** functions 217 ⟩ +≡

```
results_str = "Searching␣database␣...␣";
UpdateData(FALSE);
result_box→UpdateWindow();
CButton *pBT = (CButton *) GetDlgItem(IDC_NO_DUPLICATES);
suppress_duplicate_records = (pBT→GetState()) ? TRUE : FALSE;
ret_val = selector.list_records(sort_field, sort_order, timespan, suppress_duplicate_records);
temp_strm ≪ "Finished␣search.␣" ≪ ret_val ≪ "␣record";
if (ret_val > 1) temp_strm ≪ "s";
temp_strm ≪ "␣found.";
results_str = temp_strm.str().c_str();
UpdateData(FALSE);
result_box→UpdateWindow();
temp_strm.str("");
return; }   /* End of OnBnClickedListRecords definition. */
```

**274. OnBnClickedDescending.** [LDF Undated.]

**Log**

[LDF 2006.10.09.] !! BUG FIX: Now declaring **stringstream** *temp_strm*.

---

⟨ Declare **public Dialog_2** functions 216 ⟩ +≡

```
afx_msg void OnBnClickedDescending();
```

**275.**

⟨ Define **Dialog_2** functions 217 ⟩ +≡

```
void Dialog_2::OnBnClickedDescending()
{
  stringstream temp_strm;
  if (sort_order ≡ Selector::SORT_ASCENDING) sort_order = Selector::SORT_DESCENDING;
  else if (sort_order ≡ Selector::SORT_DESCENDING) sort_order = Selector::SORT_ASCENDING;
  else {
    temp_strm ≪ "WARNING!␣␣In␣‘Dialog_2::OnBnClickedDescending’:" ≪
        endl ≪ "‘sort_order’␣has␣invalid␣value:␣" ≪ sort_order ≪ endl ≪
        "Setting␣‘sort_order␣=␣Selector::SORT_DESCENDING’␣and␣continuing.";
    AfxMessageBox(temp_strm.str().c_str());
    temp_strm.str("");
  }
}
```

**276. OnBnClickedUseDcDate.** [LDF Undated.]

⟨ Declare **public Dialog_2** functions 216 ⟩ +≡

```
afx_msg void OnBnClickedUseDcDate();
```

**277.**

⟨ Define **Dialog_2** functions 217 ⟩ +≡

```
void Dialog_2::OnBnClickedUseDcDate()
{
  use_date_type = Selector::USE_DC_DATE;
}
```

**278. OnBnClickedUseHeaderDatestamp.** [LDF Undated.]

⟨ Declare **public Dialog_2** functions 216 ⟩ +≡

```
afx_msg void OnBnClickedUseHeaderDatestamp();
```

**279.**

⟨ Define **Dialog_2** functions 217 ⟩ +≡

```
void Dialog_2::OnBnClickedUseHeaderDatestamp()
{
  use_date_type = Selector::USE_HEADER_DATESTAMP;
}
```

**280. OnBnClickedListTitles.** [LDF 2006.06.29.]

**Log**

[LDF 2006.06.29.] Added this function.

[LDF 2006.10.09.] !! BUG FIX: Now declaring **stringstream** *temp_strm*.

---

⟨ Declare **public Dialog_2** functions 216 ⟩ +≡

```
afx_msg void OnBnClickedListTitles();
```

**281.**

⟨ Define **Dialog_2** functions 217 ⟩ +≡

```
void Dialog_2::OnBnClickedListTitles()
{
  stringstream temp_strm;
  Selector selector;
  selector.use_date_type = use_date_type;
  CEdit *result_box = (CEdit *) GetDlgItem(IDC_RESULTS);
  results_str = "Searching database ...";
  UpdateData(FALSE);
  result_box->UpdateWindow();
  int ret_val = selector.list_table(Selector::TITLES, timespan, sort_order);
  if (ret_val < 0) {
    temp_strm << "ERROR!  `Selector::list_table' failed, returning " << ret_val << ".";
    results_str = temp_strm.str().c_str();
    temp_strm.str("");
  }
  else if (ret_val == 0) {
    results_str = "Finished.  No records found.";
  }
  else   /* (ret_val > 0) */
  {
    temp_strm << "Finished.  Found " << ret_val << " records.\r\n" <<
        "See `titles.html' for results.";
    results_str = temp_strm.str().c_str();
    temp_strm.str("");
  }
  UpdateData(FALSE);
  result_box->UpdateWindow();
  return;
}   /* End of Dialog_2::OnBnClickedListTitles definition. */
```

**282. OnBnClickedCreators.** [LDF Undated.]

**Log**

[LDF 2006.10.09.] !! BUG FIX: Now declaring **stringstream** *temp_strm*.

---

⟨ Declare **public Dialog_2** functions 216 ⟩ +≡

```
afx_msg void OnBnClickedCreators();
```

**283.**

⟨ Define **Dialog_2** functions 217 ⟩ +≡

```
void Dialog_2::OnBnClickedCreators()
{
  stringstream temp_strm;
  Selector selector;
  selector.use_date_type = use_date_type;
  CEdit *result_box = (CEdit *) GetDlgItem(IDC_RESULTS);
  results_str = "Searching database ...";
  UpdateData(FALSE);
  result_box->UpdateWindow();
  int ret_val = selector.list_table(Selector::CREATORS, timespan, sort_order);
  if (ret_val < 0) {
    temp_strm << "ERROR!  `Selector::list_table' failed, returning " << ret_val << ".";
    results_str = temp_strm.str().c_str();
    temp_strm.str("");
  }
  else if (ret_val == 0) {
    results_str = "Finished.  No records found.";
  }
  else   /* (ret_val > 0) */
  {
    temp_strm << "Finished.  Found " << ret_val << " records.\r\n" <<
        "See `creators.html' for results.";
    results_str = temp_strm.str().c_str();
    temp_strm.str("");
  }
  UpdateData(FALSE);
  result_box->UpdateWindow();
  return;
}   /* End of Dialog_2::OnBnClickedListCreators definition. */
```

**284. OnBnClickedCancel.** [LDF Undated.]

⟨ Declare **public Dialog_2** functions 216 ⟩ +≡

```
afx_msg void OnBnClickedCancel();
```

**285.**

⟨ Define **Dialog_2** functions 217 ⟩ +≡

```
void Dialog_2::OnBnClickedCancel()
{
  OnCancel();
#if 0   /* 1 */
  exit(0);
#endif
}
```

**286. OnBnClickedContributors.** [LDF Undated.]

**Log**

[LDF 2006.10.09.] !! BUG FIX: Now declaring **stringstream** *temp_strm*.

⟨ Declare **public Dialog_2** functions 216 ⟩ +≡
```
afx_msg void OnBnClickedContributors();
```

**287.**

⟨ Define **Dialog_2** functions 217 ⟩ +≡
```
void Dialog_2::OnBnClickedContributors()
{
  stringstream temp_strm;
  Selector selector;
  selector.use_date_type = use_date_type;
  CEdit *result_box = (CEdit *) GetDlgItem(IDC_RESULTS);
  results_str = "Searching database ...";
  UpdateData(FALSE);
  result_box->UpdateWindow();
  int ret_val = selector.list_table(Selector::CONTRIBUTORS, timespan, sort_order);
  if (ret_val < 0) {
    temp_strm << "ERROR!  `Selector::list_table' failed, returning " << ret_val << ".";
    results_str = temp_strm.str().c_str();
    temp_strm.str("");
  }
  else if (ret_val == 0) {
    results_str = "Finished.  No records found.";
  }
  else    /* (ret_val > 0) */
  {
    temp_strm << "Finished.  Found " << ret_val << " records.\r\n" <<
        "See `contributors.html' for results.";
    results_str = temp_strm.str().c_str();
    temp_strm.str("");
  }
  UpdateData(FALSE);
  result_box->UpdateWindow();
  return;
}    /* End of Dialog_2::OnBnClickedListContributors definition. */
```

**288. OnBnClickedSubjects.** [LDF Undated.]

**Log**

[LDF 2006.10.09.] !! BUG FIX: Now declaring **stringstream** *temp_strm*.

⟨ Declare **public Dialog_2** functions 216 ⟩ +≡
```
afx_msg void OnBnClickedSubjects();
```

**289.**

⟨ Define **Dialog_2** functions 217 ⟩ +≡

```
void Dialog_2::OnBnClickedSubjects()
{
  stringstream temp_strm;
  Selector selector;
  selector.use_date_type = use_date_type;
  CEdit *result_box = (CEdit *) GetDlgItem(IDC_RESULTS);
  results_str = "Searching database ...";
  UpdateData(FALSE);
  result_box->UpdateWindow();
  int ret_val = selector.list_table(Selector::SUBJECTS, timespan, sort_order);
  if (ret_val < 0) {
    temp_strm << "ERROR!  `Selector::list_table' failed, returning " << ret_val << ".";
    results_str = temp_strm.str().c_str();
    temp_strm.str("");
  }
  else if (ret_val == 0) {
    results_str = "Finished.  No records found.";
  }
  else /* (ret_val > 0) */
  {
    temp_strm << "Finished.  Found " << ret_val << " records.\r\n" <<
        "See `subjects.html' for results.";
    results_str = temp_strm.str().c_str();
    temp_strm.str("");
  }
  UpdateData(FALSE);
  result_box->UpdateWindow();
  return;
} /* End of Dialog_2::OnBnClickedSubjects definition. */
```

**290. Dialog_2 message map.** [LDF Undated.]

⟨ **Dialog_2** message map 290 ⟩ ≡

```
BEGIN_MESSAGE_MAP(Dialog_2, CDialog)
ON_BN_CLICKED(IDOK, OnBnClickedOk)
ON_BN_CLICKED(IDC_SEARCH, OnBnClickedSearch)
ON_BN_CLICKED(IDC_BEG_OR_WHOLE_WORD, OnBnClickedBegOrWholeWord)
ON_BN_CLICKED(IDC_WHOLE_WORD_ONLY, OnBnClickedWholeWordOnly)
ON_BN_CLICKED(IDC_WHOLE_OR_PARTIAL_WORD, OnBnClickedWholeOrPartialWord)
ON_BN_CLICKED(IDC_CASE_IGNORE, OnBnClickedCaseIgnore)
ON_BN_CLICKED(IDC_EXACT_MATCH, OnBnClickedExactMatch)
ON_BN_CLICKED(IDC_ALL_DATES, OnBnClickedAllDates)
ON_BN_CLICKED(IDC_SINCE_LAST_YEAR, OnBnClickedSinceLastYear)
ON_BN_CLICKED(IDC_THIS_YEAR, OnBnClickedThisYear)
ON_BN_CLICKED(IDC_LAST_6_MONTHS, OnBnClickedLast6Months)
ON_BN_CLICKED(IDC_LAST_MONTH, OnBnClickedLastMonth)
ON_BN_CLICKED(IDC_THIS_MONTH, OnBnClickedThisMonth)
ON_BN_CLICKED(IDC_THIS_WEEK, OnBnClickedThisWeek)
ON_BN_CLICKED(IDC_LIST_RECORDS, OnBnClickedListRecords)
```

```
ON_BN_CLICKED(IDC_DESCENDING, OnBnClickedDescending)
ON_BN_CLICKED(IDC_USE_DC_DATE, OnBnClickedUseDcDate)
ON_BN_CLICKED(IDC_USE_HEADER_DATESTAMP, OnBnClickedUseHeaderDatestamp)
ON_BN_CLICKED(IDC_LIST_TITLES, OnBnClickedListTitles)
ON_BN_CLICKED(IDC_CREATORS, OnBnClickedCreators)
ON_BN_CLICKED(IDCANCEL, OnBnClickedCancel)
ON_BN_CLICKED(IDC_CONTRIBUTORS, OnBnClickedContributors)
ON_BN_CLICKED(IDC_SUBJECTS, OnBnClickedSubjects)
END_MESSAGE_MAP()
```

This code is used in section 292.

**291. Putting Dialog_2 together.**

**292.** This is what's compiled.

⟨ Include files 20 ⟩
⟨ Preprocessor macro definitions 18 ⟩
⟨ Global variable declarations 55 ⟩
⟨ **Dialog_2** message map 290 ⟩
⟨ Define **Dialog_2** functions 217 ⟩

**293.** This is what's written to `dialog2a.h`.

⟨ `dialog2a.h` 293 ⟩ ≡
⟨ Preprocessor macro calls 17 ⟩
⟨ **class Dialog_2** declaration 212 ⟩
⟨ **extern** global variable declarations 56 ⟩

**294. Global functions (`glblfncs.web`).** [LDF 2006.09.28.]

⟨ `glblfncs.web` 294 ⟩ ≡ /* Empty section for use in cross-references. */

This code is cited in sections 6 and 9.

This code is used in section 881.

**295. Include files.**

⟨ Include files 20 ⟩ +≡
**#include** "stdafx.h"

**296. Global functions.**

**297. Get HTTP file.** [LDF 2006.05.30.]

**Log**

[LDF 2006.05.30.] Added this function.

---

⟨ Declare global functions 297 ⟩ ≡
**int** *get_http_file*(**LPCTSTR** *pszURL*);

See also sections 305 and 307.

This code is used in section 311.

**298.**

⟨ Define global functions 298 ⟩ ≡

```
int get_http_file(LPCTSTR pszURL){ DWORD dwAccessType = PRE_CONFIG_INTERNET_ACCESS;
    stringstream message_strm;
    stringstream temp_strm;
    CInternetSession session("DBTest", dwAccessType);
    CHttpConnection *pServer = NULL;
    CHttpFile *pFile = NULL;
    BOOL b = false;
    DWORD dwServiceType;
    CString strServerName;
    CString strObject;
    INTERNET_PORT nPort;
```

See also sections 299, 300, 301, 302, 303, 304, 306, and 308.

This code is used in section 310.

**299.** Parse the URL passed in the **LPCTSTR** *pszURL* parameter. [LDF 2006.06.01.]

⟨ Define global functions 298 ⟩ +≡

```
b = AfxParseURL(pszURL, dwServiceType, strServerName, strObject, nPort);
```

**300.** Error handling: *AfxParseURL* failed. [LDF 2006.06.01.]

⟨ Define global functions 298 ⟩ +≡

```
if (¬b) {
  message_strm << "ERROR!␣␣In␣'get_http_file':" << endl << "dwServiceType␣==␣" <<
      dwServiceType << endl << "strServerName␣==␣" << strServerName << endl <<
      "strObject␣==␣" << strObject << endl << "nPort␣==␣" << nPort;
  AfxMessageBox(message_strm.str().c_str());
  message_strm.str("");
}   /* ¬b (AfxParseURL failed. LDF 2006.06.01. */
```

**301.** *AfxParseURL* succeeded. [LDF 2006.06.01.]

⟨ Define global functions 298 ⟩ +≡

```
#if 0
  else {
    message_strm << "In␣'get_http_file':" << "'AfxParseURL'␣succeeded." << endl <<
        "dwServiceType␣==␣" << dwServiceType << endl << "strServerName␣==␣" << strServerName <<
        endl << "strObject␣==␣" << strObject << endl << "nPort␣==␣" << nPort;
    AfxMessageBox(message_strm.str().c_str());
    message_strm.str("");
  }   /* b ≡ true (AfxParseURL succeeded. LDF 2006.06.01. */
#endif
```

**302.**

```
⟨ Define global functions 298 ⟩ +≡
  pServer = session.GetHttpConnection(strServerName, nPort);
  if (pServer ≡ 0) {
    AfxMessageBox("session.GetHttpConnection␣failed.");
    message_strm.str("");
  }
#if 0
  else    /* pServer ≠ 0 */
  {
    AfxMessageBox("session.GetHttpConnection␣succeeded.");
    message_strm.str("");
  }    /* else (pServer ≠ 0) */
#endif
  DWORD dwHttpRequestFlags = INTERNET_FLAG_EXISTING_CONNECT |
      INTERNET_FLAG_NO_AUTO_REDIRECT;
  const TCHARszHeaders[] = _T("Accept:␣text/*\r\nUser-Agent:␣DBTest\r\n");
  pFile = pServer→OpenRequest(CHttpConnection::HTTP_VERB_GET, strObject, NULL, 1, NULL, NULL,
      dwHttpRequestFlags);
  pFile→AddRequestHeaders(szHeaders);
  pFile→SendRequest();
  DWORD dwRet;
  pFile→QueryInfoStatusCode(dwRet);
#if 0
  message_strm ≪ "Result␣of␣‘pFile->QueryInfoStatusCode’␣==␣" ≪ dwRet;
  AfxMessageBox(message_strm.str().c_str());
  message_strm.str("");
#endif
  if (dwRet ≡ HTTP_STATUS_DENIED) {
    message_strm ≪ "ERROR!␣␣In␣‘get_http_file’:" ≪ endl ≪
        "HTTP␣request␣failed.␣␣Status␣code␣==␣HTTP_STATUS_DENIED." ≪ endl ≪
        "Exiting␣function␣with␣return␣value␣1.";
    AfxMessageBox(message_strm.str().c_str());
    message_strm.str("");
    session.Close();
    return 1;
  }    /* if (dwRet ≡ HTTP_STATUS_DENIED) */
```

**303.** Read text from *pFile* into the character buffer *buff* and write it to **ofstream** *out_strm*, which is attached to the file `records.xml`. [LDF 2006.06.01.]

⟨ Define global functions 298 ⟩ +≡

```
unsigned int BUFF_SIZE;
BUFF_SIZE = 1048576;    /* 1 MB */
char *buff = new char[BUFF_SIZE];
UINT ret_val;
ofstream out_strm;
out_strm.open("records.xml");
ULONGLONG file_length;
file_length = pFile→GetLength();
do {
  ret_val = pFile→Read(buff, BUFF_SIZE);
  if (ret_val > 0) {
    out_strm.write(buff, ret_val);
  }
#if 0
  message_strm ≪ "Characters␣read␣==␣" ≪ ret_val ≪ endl ≪ buff;
  AfxMessageBox(message_strm.str().c_str());
  message_strm.str("");
#endif
} while (ret_val ≡ BUFF_SIZE);
```

**304.** Exit *get_http_file* successfully with return value 0. [LDF Undated.]

⟨ Define global functions 298 ⟩ +≡

```
out_strm.close();
delete[] buff;
session.Close();
AfxMessageBox("Returning␣successfully␣from␣‘get_htpp_file’.");
return 0; }    /* End of get_http_file definition. */
```

**305. Initialize copyright strings.** [LDF 2006.12.11.]

**Log**

[LDF 2006.12.11.] Added this function.

---

⟨ Declare global functions 297 ⟩ +≡

```
int init_copyright_strings(void);
```

**306.**

```
⟨ Define global functions 298 ⟩ +≡
  int init_copyright_strings(void)
  {
#if 0     /* 1 */
#define DEBUG_OUTPUT 1
#else
#undef DEBUG_OUTPUT
#endif
    stringstream temp_strm;
#ifdef DEBUG_OUTPUT
    temp_strm.str("Entering␣‘init_copyright_strings’.");
    AfxMessageBox(temp_strm.str().c_str());
    temp_strm.str("");
#endif
    temp_strm ≪ "<h2␣align=\"center\">\n<a␣name=\"Copyright\">␣Copyright␣and␣License." ≪
        "\n</a>\n</h2>\n" ≪ "<p>\nThis␣file␣was␣generated␣by␣the␣IWF\
        ␣Metadata␣Harvester,␣" ≪ "a␣package␣for␣metadata␣harvesting.␣␣" ≪
        "The␣following␣copyright␣notice␣applies␣to␣this␣file.␣␣" ≪
        "The␣records␣contained␣in␣this␣file␣and␣the␣resources␣to␣which␣they␣refer␣" ≪
        "are␣subject␣to␣their␣own␣copyrights␣and␣licenses.\n</p>\n" ≪
        "<p>\nCopyright␣(C)␣2006,␣2007␣IWF␣Wissen␣und␣Medien␣gGmbH\n</p>\n" ≪
        "<p>\nThe␣author␣is␣Laurence␣D.␣Finston.\n</p>\n" ≪ "<p>\n" ≪ "The␣IWF␣Metadata␣Ha\
        rvester␣is␣free␣software;␣you␣can␣redistribute␣" ≪ "it␣and/or␣modify␣" ≪
        "it␣under␣the␣terms␣of␣the␣GNU␣General␣Public␣License␣as␣published␣by␣" ≪
        "the␣Free␣Software␣Foundation;␣either␣version␣2␣of␣the␣License,␣or␣" ≪
        "(at␣your␣option)␣any␣later␣version.␣" ≪ "</p>\n" ≪ "<p>\n" ≪
        "The␣IWF␣Metadata␣Harvester␣is␣distributed␣in␣the␣hope␣that␣it␣will␣be␣useful,␣" ≪
        "but␣WITHOUT␣ANY␣WARRANTY;␣without␣even␣the␣implied␣warranty␣of␣" ≪
        "MERCHANTABILITY␣or␣FITNESS␣FOR␣A␣PARTICULAR␣PURPOSE.␣␣See␣the␣" ≪
        "GNU␣General␣Public␣License␣for␣more␣details.␣" ≪ "</p>\n" ≪ "<p>\n" ≪
        "You␣should␣have␣received␣a␣copy␣of␣the␣GNU␣General␣Public␣License␣" ≪
        "along␣with␣the␣IWF␣Metadata␣Harvester;␣if␣not,␣write␣to␣the␣Free␣Software␣" ≪
        "Foundation,␣Inc.,␣51␣Franklin␣St,␣Fifth␣Floor,␣Boston,␣MA␣␣02110-1301␣␣USA␣" ≪
        "</p>\n" ≪ "<p>\n" ≪ "The␣IWF␣Metadata␣Harvester␣is␣available␣for␣downloading␣fro\
        m␣the␣" ≪ "following␣FTP␣server:␣" ≪ "ftp://ftp.gwdg.de/pub/gnu2/iwfmdh/␣" ≪
        "</p>\n" ≪ "<p>\n" ≪ "Please␣send␣bug␣reports␣to␣lfinsto1@gwdg.de␣" ≪
        "</p>\n" ≪ "<p>\n" ≪ "The␣author␣can␣be␣contacted␣at:␣" ≪ "</p>\n" ≪
        "<p>\n" ≪ "Laurence␣D.␣Finston<br>\n" ≪ "Kreuzbergring␣41<br>\n" ≪
        "D-37075␣Goettingen<br>\n" ≪ "Germany\n" ≪ "</p>\n" ≪ "<p>\n" ≪
        "lfinsto1@gwdg.de<br>\n" ≪ "s246794@stud.uni-goettingen.de\n" ≪ "</p>\n";
    copyright_html_str = temp_strm.str();
#ifdef DEBUG_OUTPUT
    temp_strm.str("Exiting␣‘init_copyright_strings’.");
    AfxMessageBox(temp_strm.str().c_str());
    temp_strm.str("");
#endif
    return 0;
#undef DEBUG_OUTPUT
  }    /* End of init_copyright_strings definition. */
```

**307. Initialize** *special_char_encodings_map.* [LDF 2006.10.27.]

**Log**

[LDF 2006.10.27.] Added this function.

[LDF 2006.10.30.] Added code to this function for all of the special characters that were accounted for before.

**To Do**

[LDF 2006.10.30.] Add code to account for more special characters!

⟨ Declare global functions 297 ⟩ +≡
**int** *init_special_char_encodings_map*(**void**);

**308.**

```
⟨ Define global functions 298 ⟩ +≡
  int init_special_char_encodings_map(void)
  {
#if 0     /* 1 */
#define DEBUG_OUTPUT 1
#else
#undef DEBUG_OUTPUT
#endif
    stringstream temp_strm;
#ifdef DEBUG_OUTPUT
    temp_strm.str("Entering␣‘init_special_char_encodings_map’.");
    AfxMessageBox(temp_strm.str().c_str());
    temp_strm.str("");
#endif
    special_char_encodings_map_mutex.Lock();
    special_char_encodings_map[string("Â\204")] = '\204';    /* #C2 84 */
    special_char_encodings_map[string("Â\223")] = '\223';    /* #C2 93 */
    special_char_encodings_map[string("Â¡")] = '¡';    /* #C2 A1 */
    special_char_encodings_map[string("Â§")] = '§';    /* #C2 A7 */
    special_char_encodings_map[string("Â¨")] = '¨';    /* #C2 A8 */
    special_char_encodings_map[string("Â®")] = '®';    /* #C2 AE */
    special_char_encodings_map[string("Â°")] = '°';    /* #C2 B0 */
    special_char_encodings_map[string("Â±")] = '±';    /* #C2 B1 */
    special_char_encodings_map[string("Â·")] = '·';    /* #C2 B7 */
    special_char_encodings_map[string("Â¸")] = '¸';    /* #C2 B8 */
    special_char_encodings_map[string("Â³")] = '³';    /* #C2 B3 */
    special_char_encodings_map[string("Â´")] = '´';    /* #C2 B4 */
    special_char_encodings_map[string("Âµ")] = 'µ';    /* #C2 B5 */
    special_char_encodings_map[string("Â¸")] = '¸';    /* #C2 B8 */
    special_char_encodings_map[string("Â½")] = '½';    /* #C2 BD */
    special_char_encodings_map[string("Â¿")] = '¿';    /* #C2 BF */
    special_char_encodings_map[string("Ã\200")] = 'À';    /* #C3 80 */
    special_char_encodings_map[string("Ã\201")] = 'Á';    /* #C3 81 */
    special_char_encodings_map[string("Ã\204")] = 'Ä';    /* #C3 84 */
    special_char_encodings_map[string("Ã\205")] = 'Å';    /* #C3 85 */
    special_char_encodings_map[string("Ã\206")] = 'Æ';    /* #C3 86 */
    special_char_encodings_map[string("Ã\207")] = 'Ç';    /* #C3 87 */
    special_char_encodings_map[string("Ã\210")] = 'È';    /* #C3 88 */
    special_char_encodings_map[string("Ã\211")] = 'É';    /* #C3 89 */
    special_char_encodings_map[string("Ã\212")] = 'Ê';    /* #C3 8A */
    special_char_encodings_map[string("Ã\213")] = 'Ë';    /* #C3 8B */
    special_char_encodings_map[string("Ã\216")] = 'Î';    /* #C3 8E */
    special_char_encodings_map[string("Ã\226")] = 'Ö';    /* #C3 96 */
    special_char_encodings_map[string("Ã\227")] = '×';    /* #C3 97 */
    special_char_encodings_map[string("Ã\233")] = 'Û';    /* #C3 9B */
    special_char_encodings_map[string("Ã\234")] = 'Ü';    /* #C3 9C */
    special_char_encodings_map[string("Ã\237")] = 'ß';    /* #C3 9F */
    special_char_encodings_map[string("Ã\240")] = 'à';    /* #C3 A0 */
    special_char_encodings_map[string("Ã¡")] = 'á';    /* #C3 A1 */
    special_char_encodings_map[string("Ã¤")] = 'ä';    /* #C3 A4 */
    special_char_encodings_map[string("Ã§")] = 'ç';    /* #C3 A7 */
```

```
    special_char_encodings_map[string("Ã¨")] = 'è';    /* #C3 A8 */
    special_char_encodings_map[string("Ã©")] = 'é';    /* #C3 A9 */
    special_char_encodings_map[string("Ã-")] = 'í';    /* #C3 AD */
    special_char_encodings_map[string("Ã®")] = 'î';    /* #C3 AE */
    special_char_encodings_map[string("Ã¯")] = 'ï';    /* #C3 AF */
    special_char_encodings_map[string("Ã±")] = 'ñ';    /* #C3 B1 */
    special_char_encodings_map[string("Ã³")] = 'ó';    /* #C3 B3 */
    special_char_encodings_map[string("Ã¶")] = 'ö';    /* #C3 B6 */
    special_char_encodings_map[string("Ã¼")] = 'ü';    /* #C3 BC */
    special_char_encodings_map[string("Ã½")] = 'ý';    /* #C3 BD */
    special_char_encodings_map[string("Ã¿")] = 'ÿ';    /* #C3 BF */
    special_char_encodings_map_mutex.Unlock();
#ifdef DEBUG_OUTPUT
    for (map<string, char>::const_iterator iter = special_char_encodings_map.begin();
            iter ≠ special_char_encodings_map.end(); ++iter) {
      temp_strm ≪ iter→first ≪ ":␣" ≪ iter→second ≪ endl;
    }
    AfxMessageBox(temp_strm.str().c_str());
    temp_strm.str("");
#endif
    return 0;
#undef DEBUG_OUTPUT
  }    /* End of init_special_char_encodings_map definition. */
```

**309. Putting global functions together.**

**310.** This is what's compiled.

```
  ⟨ Include files 20 ⟩
#define _UNICODE
  ⟨ Define global functions 298 ⟩
```

**311.** This is what's written to `glblfncs.h`.

⟨ `glblfncs.h` 311 ⟩ ≡
⟨ Declare global functions 297 ⟩

**312. Metadata_Source (`mtdtsrc.web`).** [LDF Undated.]
WARNING! This file contains special characters. Care should be exercised, if
it's edited in editors that change the encoding of characters, such as Emacs. They are in the function definition of *parse_record*. [LDF 2006.07.04.]

⟨ `mtdtsrc.web` 312 ⟩ ≡ /* Empty section for use in cross-references. */

This code is cited in sections 6 and 9.

This code is used in section 881.

**313. Include files.**

⟨ Include files 20 ⟩ +≡
**#include** `"stdafx.h"`

**314. Preprocessor macro calls.**

⟨ Preprocessor macro calls 17 ⟩ +≡
**#pragma** *once*

**315. class Metadata_Source declaration.** [LDF Undated.]

**Log**

[LDF 2006.09.11.] Added **const char** *∗records_file_name* argument to *update_database*, *sub_update_database_timms*, and *sub_update_database_dbt*.

---

⟨ Declare **class Metadata_Source** 315 ⟩ ≡
**class Metadata_Source** {
**public**: ⟨ Declare **static** constants in **class Metadata_Source** 316 ⟩
⟨ Declare **Metadata_Source** functions 320 ⟩
**protected**: **unsigned short** *id*;
};

This code is used in section 378.

**316. static constants in class Metadata_Source.** [LDF Undated.]

**Log**

[LDF 2006.07.06.] Added **static const unsigned short** `DBT`. It refers to the Digitale Bibliothek Thüringen.

---

⟨ Declare **static** constants in **class Metadata_Source** 316 ⟩ ≡
**static const unsigned short** `NULL_METADATA_SOURCE`;
**static const unsigned short** `TIMMS`;
**static const unsigned short** `DBT`;
**static const unsigned short** `MAX_TAG_LENGTH`;
**static const unsigned short** `MAX_RESUMPTION_TOKEN`;

This code is used in section 315.

**317.**

⟨ Initialize **static** constants in **class Metadata_Source** 317 ⟩ ≡
**const unsigned short Metadata_Source** :: `MAX_TAG_LENGTH` = 128;
**const unsigned short Metadata_Source** :: `MAX_RESUMPTION_TOKEN` = 128;
**const unsigned short Metadata_Source** :: `NULL_METADATA_SOURCE` = 0;
**const unsigned short Metadata_Source** :: `TIMMS` = 1;
**const unsigned short Metadata_Source** :: `DBT` = 2;

This code is used in section 377.

**318. Functions.** [LDF 2006.10.04.]

**319. Constructors.**

**320. Default constructor.** [LDF Undated.]

⟨ Declare **Metadata_Source** functions 320 ⟩ ≡
**Metadata_Source**(**void**);

See also sections 322, 324, 327, 336, 341, and 348.

This code is used in section 315.

**321.**

⟨ Define **Metadata_Source** functions 321 ⟩ ≡
**Metadata_Source** :: **Metadata_Source**(**void**)
: *id*(0) {
**return**;
}

See also sections 323, 325, 328, 329, 330, 331, 332, 333, 334, 335, 337, 338, 339, 340, 342, 343, 344, 345, 346, and 347.

This code is used in section 377.

**322. const unsigned short argument.** [LDF Undated.]

⟨ Declare **Metadata_Source** functions 320 ⟩ +≡
**Metadata_Source**(**const unsigned short** *iid*);

**323.**

⟨ Define **Metadata_Source** functions 321 ⟩ +≡
**Metadata_Source** :: **Metadata_Source**(**const unsigned short** *iid*)
: *id*(*iid*) {
**return**;
}

**324. Destructor.**

⟨ Declare **Metadata_Source** functions 320 ⟩ +≡
**virtual** ~**Metadata_Source**(**void**);

**325.**

⟨ Define **Metadata_Source** functions 321 ⟩ +≡

```
Metadata_Source::~Metadata_Source(void)
{
  return;
}
```

**326. Updating the database.** [LDF 2006.09.28.]

**Log**

[LDF 2006.09.28.] Added this section.

---

**327. Update database.** [LDF Undated.]

**Log**

[LDF Undated.] Added this function.

[LDF 2006.09.11.] Added **const char** **records_file_name* argument. Also added it to *sub_update_database_timms*, and *sub_update_database_dbt*, so I've altered the calls to these functions in this function accordingly.

---

⟨ Declare **Metadata_Source** functions 320 ⟩ +≡

```
int update_database(BOOL delete_tables, char *resumption_token, const char *records_file_name);
```

**328.**

⟨ Define **Metadata_Source** functions 321 ⟩ +≡

```
int Metadata_Source::update_database(BOOL delete_tables, char *resumption_token, const char
        *records_file_name){ stringstream temp_strm;
    Records curr_record;
    curr_record.Open();
    CDatabase *cdb = curr_record.m_pDatabase;
    cdb->SetLoginTimeout(120);
    cdb->SetQueryTimeout(120);
    int ret_val;
#if 1    /* 0 */
    temp_strm << "Entering␣‘Metadata_Source::update_database’.";
    AfxMessageBox(temp_strm.str().c_str());
    temp_strm.str("");
#endif
```

**329.** Test whether *cdb* is open. [LDF 2006.05.26.]

⟨ Define **Metadata_Source** functions 321 ⟩ +≡

```
BOOL b;
try {
  b = cdb→IsOpen( );
}
catch(CDBException *e)
{
  temp_strm ≪ "Exception␣when␣calling␣‘cdb.IsOpen’:␣" ≪ e→m_strError ≪
      endl ≪ "Exiting␣‘Metadata_Source::update_database’␣" ≪
      "unsuccessfully␣with␣return␣value␣1.";
  AfxMessageBox(temp_strm.str( ).c_str( ));
  temp_strm.str("");
  e→Delete( );
  curr_record.Close( );
  return 1;
}
#if 0
  if (b) AfxMessageBox("‘cdb’␣is␣open.");
  else AfxMessageBox("‘cdb’␣isn’t␣open.");
#endif
```

**330.** Test whether *cdb* can be updated. [LDF 2006.05.26.]

⟨ Define **Metadata_Source** functions 321 ⟩ +≡

```
try {
  b = cdb→CanUpdate( );
}
catch(CDBException *e)
{
  AfxMessageBox("Exception␣when␣calling␣‘cdb.CanUpdate’.");
  AfxMessageBox(e→m_strError, MB_ICONEXCLAMATION);
  AfxMessageBox("Exiting␣‘test_cdatabase’␣unsuccessfully␣with␣return␣value␣1.");
  e→Delete( );
  curr_record.Close( );
  return 1;
}    /* catch */
#if 0
  if (b) AfxMessageBox("‘cdb’␣can␣be␣updated.");
  else AfxMessageBox("‘cdb’␣can␣be␣updated.");
#endif
```

**331.** Test whether transactions are supported for *cdb*. [LDF 2006.05.26.]

⟨ Define **Metadata_Source** functions 321 ⟩ +≡

```
try {
  b = cdb→CanTransact( );
}
catch(CDBException *e)
{
  AfxMessageBox("Exception␣when␣calling␣‘cdb.CanTransact’.");
  AfxMessageBox(e→m_strError, MB_ICONEXCLAMATION);
  AfxMessageBox("Exiting␣‘test_cdatabase’␣unsuccessfully␣with␣return␣value␣1.");
  e→Delete( );
  curr_record.Close( );
  return 1;
}   /* catch */
#if 0
  if (b) AfxMessageBox("Transactions␣can␣be␣performed␣using␣‘cdb’.");
  else AfxMessageBox("Transactions␣cannot␣be␣performed␣using␣‘cdb’.");
#endif
  string cmd_str;
```

**332.** Delete tables. [LDF 2006.05.30.]

**Log**

[LDF 2006.06.26. ] Searching for names doesn't work right. "Whole word" doesn't find last names the way they're stored by TIMMS, e.g., "Feil, Robert".

**To Do**

[LDF 2006.07.04.] This problem seems to have been fixed, somehow. Check this.

---

[LDF Undated.] I had a problem with a timeout when deleting tables. After I did it in the SQL Query Analyzer, tried a couple of times with this program, and executed other SQL commands from this program, the problem disappeared. I think it may be because SQL was then able to use an "execution plan", or whatever this is called. The procedure *delete_tables* ran rather slowly when I called it in SQL Query Analyzer. This problem might not occur on a faster machine, and when it's not running on a virtual machine.

---

```
⟨Define Metadata_Source functions 321⟩ +≡
  if (delete_tables) {
    try {
      cdb→ExecuteSQL("exec␣delete_tables");
    }
    catch(CDBException *e)
    {    /* The error code is in e→m_nRetCode */
      temp_strm ≪ "Exception␣when␣calling␣‘cdb->ExecuteSQL’." ≪ endl ≪ "Error␣code␣=␣" ≪
          e→m_nRetCode ≪ endl ≪ "Exception␣==␣" ≪ e→m_strError ≪ endl ≪
          "Exiting␣‘test_cdatabase’␣unsuccessfully␣with␣return␣value␣1.";
      AfxMessageBox(temp_strm.str().c_str());
      temp_strm.str("");
      e→Delete();
      curr_record.Close();
      return 1;
    }    /* catch */
#if 0
    AfxMessageBox("Deleting␣tables␣succeeded.");
#endif
  }    /* if (delete_tables) */    /* **** (4) id ≡ TIMMS */
  if (id ≡ TIMMS) {
    ret_val = sub_update_database_timms(cdb, curr_record, resumption_token, records_file_name);
#if 0    /* 1 */
    temp_strm ≪ "‘sub_update_database_timms’␣returned␣" ≪ ret_val;
    AfxMessageBox(temp_strm.str().c_str());
    temp_strm.str("");
#endif    /* ***** (5) */
    temp_strm ≪ "exec␣fill_catalogues";
    try {
      cdb→ExecuteSQL(temp_strm.str().c_str());
    }
    catch(CDBException *e)
    {    /* The error code is in e→m_nRetCode */
```

```
      temp_strm << "Exception␣when␣calling␣`cdb->ExecuteSQL'." << endl << "Error␣code␣=␣" <<
          e→m_nRetCode << endl << "Exception␣==␣" << e→m_strError << endl <<
          "Exiting␣`test_cdatabase'␣unsuccessfully␣with␣return␣value␣1.";
      AfxMessageBox(temp_strm.str().c_str());
      temp_strm.str("");
      e→Delete();
      curr_record.Close();
      return 1;
    }   /* catch */
    temp_strm.str("");
  }   /* if (id ≡ TIMMS) */
```

**333.** *id* ≡ DBT. [LDF Undated.]

```
⟨Define Metadata_Source functions 321⟩ +≡
  else
    if (id ≡ DBT) {
      ret_val = sub_update_database_dbt(cdb, curr_record, resumption_token, records_file_name);
#if 1   /* 0 */
      temp_strm << "`sub_update_database_dbt␣returned␣" << ret_val;
      AfxMessageBox(temp_strm.str().c_str());
      temp_strm.str("");
#endif
    }   /* else if (id ≡ DBT) */
```

**334.** *id* has invalid value. [LDF Undated.]

```
⟨Define Metadata_Source functions 321⟩ +≡
  else {
    temp_strm << "ERROR!␣␣In␣`Metadata_Source::update_database':␣␣" <<
        "`id'␣has␣invalid␣value:␣␣" << id << endl <<
        "Exiting␣function␣unsuccessfully␣with␣return␣value␣1.";
    AfxMessageBox(temp_strm.str().c_str());
    temp_strm.str("");
    return 1;
  }   /* else (id has invalid value). */
```

**335.** Exit function successfully with return value 0. [LDF Undated.]

```
⟨Define Metadata_Source functions 321⟩ +≡
#if 1   /* 0 */
  temp_strm << "Exiting␣`Metadata_Source::update_database'.";
  AfxMessageBox(temp_strm.str().c_str());
  temp_strm.str("");
#endif
  return 0; }   /* End of int Metadata_Source::update_database definition. */
```

**336. Sub-update database DBT.** [LDF Undated.]

**Log**

[LDF Undated.] Added this function.

[LDF 2006.09.11.] Added **const char** *∗records_file_name* argument.

---

⟨ Declare **Metadata_Source** functions 320 ⟩ +≡

```
int sub_update_database_dbt(CDatabase *cdb,
Records &curr_record,
char *resumption_token,
const char *records_file_name);
```

**337.**

⟨ Define **Metadata_Source** functions 321 ⟩ +≡

```
int Metadata_Source::sub_update_database_dbt(
CDatabase *cdb,
Records &curr_record,
char *resumption_token,
const char *records_file_name){ stringstream temp_strm;
#if 0     /* 1 */
    temp_strm << "Entering␣`Metadata_Source::sub_update_database_dbt'.";
    AfxMessageBox(temp_strm.str().c_str());
    temp_strm.str("");
#endif
```

**338.** Test reading XML data from a file. [LDF 2006.05.29. ]

⟨ Define **Metadata_Source** functions 321 ⟩ +≡

```
ifstream in_file_strm;
in_file_strm.open(records_file_name);
if (in_file_strm.is_open()) {
#if 0
    AfxMessageBox("`in_file_strm'␣is␣open.");
#endif
}
else {
    temp_strm << "ERROR!␣␣In␣`Metadata_Source::sub_update_cdatabase_dbt':␣␣" <<
        "Failed␣to␣open␣`in_file_strm'." << endl <<
        "Exiting␣function␣unsuccessfully␣with␣return␣value␣1.";
    AfxMessageBox(temp_strm.str().c_str());
    temp_strm.str("");
    curr_record.Close();
    return 1;
}   /* else (Failed to open in_file_strm). LDF 2006.05.30. */
string record_str;
string prefix_str;
string prefix_str_1;
string suffix_str;
string cmd_str;
int ret_val = 0;
temp_strm << "declare␣@Outer_namespace_data␣varchar(8000)\n" << "set␣@Outer_namespac\
    e_data␣=␣" << "'<ns␣xmlns=\"http://www.openarchives.org/OAI/2.0/\"\n" <<
    "xmlns:oai_dc=\"http://www.openarchives.org/OAI/2.0/oai_dc/\"\n" <<
    "xmlns:dc=\"http://purl.org/dc/elements/1.1/\"\n" <<
    "xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\"\n" <<
    "xsi:schemaLocation=\"http://www.openarchives.org/OAI/2.0/\n" <<
    "http://www.openarchives.org/OAI/2.0/OAI-PMH.xsd\"/>'\n" << "exec␣write_to_tables␣" <<
    DBT << ",␣";
prefix_str = temp_strm.str();
temp_strm.str("");
prefix_str_1 = ",␣N'";
suffix_str = "',␣@Outer_namespace_data";
char *temp_resumption_token;
temp_resumption_token = new char[MAX_RESUMPTION_TOKEN];
```

**339.** Call *parse_record* in a loop until we reach the end of *in_file_strm*. [LDF 2006.05.30.]

⟨ Define **Metadata_Source** functions 321 ⟩ +≡

```
  unsigned long record_ctr = 1;
  strcpy(resumption_token, ""); while (¬in_file_strm.eof()) { strcpy(temp_resumption_token, "");
  ret_val = parse_record(&in_file_strm, &record_str, temp_resumption_token);
  if (strlen(temp_resumption_token) > 0) {
    strcpy(resumption_token, temp_resumption_token);
#if 0   /* 1 */
    temp_strm ≪ "temp_resumption_token␣==␣" ≪ temp_resumption_token ≪ endl ≪
        "resumption_token␣==␣" ≪ temp_resumption_token;
    AfxMessageBox(temp_strm.str().c_str());
    temp_strm.str("");
#endif
  }   /* if */
  if (ret_val ≡ 1) {
    temp_strm ≪ "ERROR!␣␣In␣‘Metadata_Source::sub_update_database_dbt’:␣␣" ≪
        "‘parse_record’␣failed." ≪ endl ≪ "Exiting␣function␣unsuccessfully␣with␣re\
        turn␣value␣1.";
    AfxMessageBox(temp_strm.str().c_str());
    temp_strm.str("");
    curr_record.Close();
    in_file_strm.close();
    return 1;
  }   /* if (ret_val ≠ 0) */
  else if (ret_val ≡ 2) {
#if 0   /* 1 */
    temp_strm ≪ "In␣‘Metadata_Source::sub_update_database_dbt’:␣␣" ≪ "Reached␣end␣of␣‘in_\
        file_strm’␣in␣‘parse_record’␣without␣" ≪ "collecting␣a␣‘<record>’." ≪ endl ≪
        "Exiting␣function␣successfully␣with␣return␣value␣0.";
    AfxMessageBox(temp_strm.str().c_str());
    temp_strm.str("");
#endif
#if 0
    if (strlen(resumption_token) > 0) {
      temp_strm ≪ "After␣loop:␣␣resumption_token␣==␣" ≪ resumption_token;
      AfxMessageBox(temp_strm.str().c_str());
      temp_strm.str("");
    }
#endif
    delete[] temp_resumption_token;
    temp_resumption_token = 0;
      /* Close open resources and exit function successfully with return value 0. LDF 2006.05.26. */
    curr_record.Close();
    in_file_strm.close();
#if 0   /* 1 */
    temp_strm ≪ "Exiting␣‘Metadata_Source::sub_update_database_dbt’.";
    AfxMessageBox(temp_strm.str().c_str());
    temp_strm.str("");
#endif
    return 0;
  }   /* if (ret_val ≠ 0) */
#if 0   /* 1 */
```

```
  temp_strm ≪ "record_str:" ≪ endl ≪ record_str;
  AfxMessageBox(temp_strm.str().c_str());
  temp_strm.str("");
#endif
  string::size_type start_pos = record_str.find("<metadata>");
  start_pos += 10;
  string::size_type curr_pos = start_pos;
  unsigned int brace_ctr;
  char curr_char;
  while ((curr_char = record_str[start_pos++]) ≠ '<') ;
  curr_pos = start_pos--;
  brace_ctr = 1;
  for ( ; brace_ctr > 0; ++curr_pos) {
    curr_char = record_str[curr_pos];
    if (curr_char ≡ '<') ++brace_ctr;
    else if (curr_char ≡ '>') --brace_ctr;
  }    /* for */
#if 0    /* 1 */
  temp_strm ≪ "start_pos␣==␣" ≪ start_pos ≪ endl ≪ "curr_pos␣==␣" ≪ curr_pos ≪ endl ≪
      "String␣to␣be␣replaced:" ≪ endl ≪ "\"" ≪ record_str.substr(start_pos,
      curr_pos − start_pos) ≪ "\"";
  AfxMessageBox(temp_strm.str().c_str());
  temp_strm.str("");
#endif
  temp_strm ≪ "<oai_dc:dc␣xmlns:oai_dc=" ≪ "\"http://www.openarchives.org/OAI/2.0/o\
    ai_dc/\"\n" ≪ "xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\"\n" ≪
    "xmlns:dc=\"http://purl.org/dc/elements/1.1/\"\n" ≪
    "xsi:schemaLocation=\"http://www.openarchives.org/OAI/2.0/oai_dc/\n" ≪
    "http://www.openarchives.org/OAI/2.0/oai_dc.xsd\">";
  record_str.replace(start_pos, curr_pos − start_pos, temp_strm.str());
  temp_strm.str("");
#if 0    /* 1 */
  temp_strm ≪ "record_str␣after␣replacement:\n" ≪ record_str;
  AfxMessageBox(temp_strm.str().c_str());
  temp_strm.str("");
#endif
  temp_strm ≪ prefix_str ≪ record_ctr++ ≪ prefix_str_1 ≪ record_str ≪ suffix_str;
  cmd_str = temp_strm.str();
  temp_strm.str("");
#if 0    /* 1 */
  temp_strm ≪ "cmd_str:" ≪ endl ≪ cmd_str;
  AfxMessageBox(temp_strm.str().c_str());
  temp_strm.str("");
#endif
  try {
    cdb→ExecuteSQL(cmd_str.c_str());
  }
  catch(CDBException *e){    /* The error code is in e→m_nRetCode */
      temp_strm ≪ "ERROR!␣␣In␣‘Metadata_Source::sub_update_database_dbt’:" ≪ endl ≪
          "Exception␣thrown␣when␣calling␣‘cdb->ExecuteSQL’:" ≪ endl ≪ "Error␣code␣==␣" ≪
```

```
        e→m_nRetCode ≪ endl ≪ "Error␣string␣==␣" ≪ e→m_strError ≪ endl ≪ "‘cmd_str’␣==␣" ≪
        cmd_str;
#if 0   /* 1 */
    ≪ endl ≪ "Writing␣‘cmd_str’␣to␣log␣file␣and␣will␣try␣to␣continue.";
#endif
    AfxMessageBox(temp_strm.str().c_str());
    temp_strm.str("");
    e→Delete();
    curr_record.Close();
    in_file_strm.close();
#if 0   /* 1 */
    log_file ≪ endl ≪ cmd_str ≪ endl;
#endif
    record_str.clear();
    cmd_str.clear();
    continue; }   /* catch */
#if 0   /* 1 */
    AfxMessageBox("Calling␣‘write_to_tables’␣succeeded.");
#endif
    record_str.clear();
    cmd_str.clear(); }   /* while (¬in_file_strm.eof()) */
#if 0
    if (strlen(resumption_token) > 0) {
      temp_strm ≪ "After␣loop:␣␣resumption_token␣==␣" ≪ resumption_token;
      AfxMessageBox(temp_strm.str().c_str());
      temp_strm.str("");
    }
#endif
    delete[] temp_resumption_token;
    temp_resumption_token = 0;
```

**340.** End of “Test reading XML data from a file”.
Close open resources and exit function successfully with return value 0. [LDF 2006.05.26.]

⟨ Define **Metadata_Source** functions 321 ⟩ +≡

```
  in_file_strm.close();
  curr_record.Close();
#if 0   /* 1 */
  temp_strm ≪ "Exiting␣‘Metadata_Source::sub_update_database_dbt’.";
  AfxMessageBox(temp_strm.str().c_str());
  temp_strm.str("");
#endif
  return 0; }   /* End of Metadata_Source::sub_update_database_dbt definition. */
```

**341. Sub-update database TIMMS.** [LDF Undated.]

**Log**

[LDF 2006.07.06.] Renamed this file to `sub_update_database_timms.cpp` from `sub_update_database.cpp`.

[LDF 2006.09.11.] Added **const char** **records_file_name* argument.

[LDF 2006.10.09.] Moved the definition of **Metadata_Source** :: *sub_update_database_timms* from `sub_update_database_timms.cpp` to this file (`mtdtsrc.web`).

---

⟨ Declare **Metadata_Source** functions 320 ⟩ +≡
**int** *sub_update_database_timms*(**CDatabase** **cdb*, **Records** &*curr_record*, **char** **resumption_token*, **const char** **records_file_name*);

**342.**

⟨ Define **Metadata_Source** functions 321 ⟩ +≡
**int Metadata_Source** :: *sub_update_database_timms*(**CDatabase** **cdb*, **Records** &*curr_record*, **char** **resumption_token*, **const char** **records_file_name*){

**343.** Read XML data from `records.xml`. [LDF 2006.05.29.]

⟨ Define **Metadata_Source** functions 321 ⟩ +≡

```
  ifstream in_file_strm(records_file_name);
  stringstream temp_strm;
  if (in_file_strm.is_open()) {
#if 0     /* 1 */
    AfxMessageBox("`in_file_strm'␣is␣open.");
#endif
  }
  else {
    temp_strm ≪ "Error!␣␣In␣`sub_update_cdatabase_timms':␣␣" ≪ "Failed␣to␣open␣`in_\
        file_strm'." ≪ endl ≪ "Exiting␣function␣unsuccessfully␣with␣return␣value␣1.";
    AfxMessageBox(temp_strm.str().c_str());
    temp_strm.str("");
    curr_record.Close();
    return 1;
  }     /* else (Failed to open in_file_strm). LDF 2006.05.30. */
  string record_str;
  string prefix_str;
  string prefix_str_1;
  string suffix_str;
  string cmd_str;
  int ret_val = 0;
  temp_strm ≪ "declare␣@Outer_namespace_data␣varchar(8000)\n" ≪
      "set␣@Outer_namespace_data␣=␣" ≪ "'<ns␣xmlns=\"http://www.openarchives.or\
      g/OAI/2.0\"\n" ≪ "xmlns:dc=\"http://purl.org/dc/elements/1.1/\"\n" ≪
      "xmlns:oai_dc=\"http://www.openarchives.org/OAI/2.0/oai_dc/\"/>'\n" ≪
      "exec␣write_to_tables␣" ≪ TIMMS ≪ ",␣";
  prefix_str = temp_strm.str();
  temp_strm.str("");
  prefix_str_1 = ",␣N'";
  suffix_str = "',␣@Outer_namespace_data";
  char *temp_resumption_token;
  temp_resumption_token = new char[MAX_RESUMPTION_TOKEN];
```

**344.** Call *parse_record* in a loop until we reach the end of *in_file_strm*. [LDF 2006.05.30.]

⟨ Define **Metadata_Source** functions 321 ⟩ +≡

```
  unsigned long record_ctr = 1;
  strcpy(resumption_token, ""); while (¬in_file_strm.eof()) { strcpy(temp_resumption_token, "");
  ret_val = parse_record(&in_file_strm, &record_str, temp_resumption_token);
```

**345.** The records file contained a resumption token. [LDF 2006.08.29.]

⟨ Define **Metadata_Source** functions 321 ⟩ +≡

```
  if (strlen(temp_resumption_token) > 0) {
    strcpy(resumption_token, temp_resumption_token);
#if 0      /* 1 */
    temp_strm << "resumption_token␣==␣" << resumption_token;
    AfxMessageBox(temp_strm.str().c_str());
    temp_strm.str("");
#endif
  }      /* if (strlen(temp_resumption_token) > 0) */
  if (ret_val ≡ 1) {
    temp_strm << "ERROR!␣␣In␣`Metadata_Source::sub_update_database_timms':␣␣" <<
        "`parse_record'␣failed." << endl << "Exiting␣function␣unsuccessfully␣with␣re\
        turn␣value␣1.";
    AfxMessageBox(temp_strm.str().c_str());
    temp_strm.str("");
    curr_record.Close();
    in_file_strm.close();
    return 1;
  }      /* if (ret_val ≠ 0) */
  else if (ret_val ≡ 2) {
#if 0
  temp_strm << "In␣`Metadata_Source::sub_update_database_timms':␣␣" << "Reached␣end␣of␣`in_\
      file_strm'␣in␣`parse_record'␣without␣" << "collecting␣a␣`<record>'." << endl <<
      "Exiting␣function␣successfully␣with␣return␣value␣0.";
  AfxMessageBox(temp_strm.str().c_str());
  temp_strm.str("");
#endif
#if 0      /* 1 */
  if (strlen(resumption_token) > 0) {
    temp_strm << "In␣last␣iteration:␣␣resumption_token␣==␣" << resumption_token;
    AfxMessageBox(temp_strm.str().c_str());
    temp_strm.str("");
  }
#endif
  delete[] temp_resumption_token;
  temp_resumption_token = 0;
```

**346.** Close open resources and exit function successfully with return value 0. [LDF 2006.05.26.]

⟨ Define **Metadata_Source** functions 321 ⟩ +≡

```
  curr_record.Close();
  in_file_strm.close();
  return 0; }    /* else if (ret_val ≡ 2) */
#if 0
  temp_strm << "record_str:" << endl << record_str;
  AfxMessageBox(temp_strm.str().c_str());
  temp_strm.str("");
#endif
  temp_strm << prefix_str << record_ctr++ << prefix_str_1 << record_str << suffix_str;
  cmd_str = temp_strm.str();
  temp_strm.str("");
#if 0
  temp_strm << "cmd_str:" << endl << cmd_str;
  AfxMessageBox(temp_strm.str().c_str());
  temp_strm.str("");
#endif
  try {
    cdb->ExecuteSQL(cmd_str.c_str());
  }
  catch(CDBException *e)
  {    /* The error code is in e->m_nRetCode */
    temp_strm << "ERROR!  In 'Metadata_Source::sub_update_database_timms':" << endl <<
        "Exception thrown when calling 'cdb->ExecuteSQL':" << endl << "Error code == " <<
        e->m_nRetCode << endl << "Error string == " << e->m_strError << endl << "'cmd_str' == " <<
        cmd_str << endl << "Writing 'cmd_str' to log file and will try to continue.";
    AfxMessageBox(temp_strm.str().c_str());
    temp_strm.str("");
    e->Delete();
    curr_record.Close();
    in_file_strm.close();
#if 0    /* 1 */
    log_file_mutex.Lock();
    log_file << endl << cmd_str << endl;
    log_file_mutex.Unlock();
#endif
    record_str.clear();
    cmd_str.clear();
    continue;
  }    /* catch */
#if 0
  AfxMessageBox("Calling 'write_to_tables' succeeded.");
#endif
  record_str.clear();
  cmd_str.clear(); }    /* while (¬in_file_strm.eof()) */
#if 0
  if (strlen(resumption_token) > 0) {
    temp_strm << "After loop:  resumption_token == " << resumption_token;
    AfxMessageBox(temp_strm.str().c_str());
    temp_strm.str("");
```

```
  }
#endif
  delete[] temp_resumption_token;
  temp_resumption_token = 0;
```

**347.** End of reading XML data from the records file. [LDF 2006.09.11.]
Close open resources and exit function successfully with return value 0. [LDF 2006.05.26.]

```
⟨ Define Metadata_Source functions 321 ⟩ +≡
  in_file_strm.close();
  curr_record.Close();
  return 0; }   /* End of Metadata_Source::sub_update_database_timms definition. */
```

**348. Parse record.** [LDF Undated.]

```
⟨ Declare Metadata_Source functions 320 ⟩ +≡
  int parse_record(ifstream *in_strm, string *out_str, char *resumption_token);
```

**349.**

```
⟨ Define Metadata_Source::parse_record 349 ⟩ ≡
  int Metadata_Source::parse_record(ifstream *in_strm,
  string *out_str,
  char *resumption_token){ stringstream temp_strm;
      char curr_char = '\0';
      char special_str[3] = "";
      char tag_str[MAX_TAG_LENGTH];
      int tag_ctr = 0;
      stringstream message_strm;
```

See also sections 350, 351, 352, 353, 354, 355, 356, 357, 358, 359, 360, 361, 362, 363, 364, 365, 366, 367, 368, 369, 370, 371, 372, 373, 374, and 375.

This code is cited in section 11.

This code is used in section 377.

**350.** Error handling: **ifstream** $*in\_strm \equiv 0$ [LDF 2006.05.29.]

```
⟨ Define Metadata_Source::parse_record 349 ⟩ +≡
  if (in_strm ≡ 0) {
    message_strm ≪ "ERROR!␣␣In␣‘parse_record’:␣␣‘ifstream*␣in_strm’␣==␣0." ≪ endl ≪
        "Exiting␣function␣unsucessfully␣with␣return␣value␣1.";
    AfxMessageBox(message_strm.str().c_str());
    message_strm.str("");
    return 1;
  }   /* if (in_strm ≡ 0) */
```

**351.** Error handling: $out\_str \equiv 0$. [LDF 2006.05.29.]

```
⟨ Define Metadata_Source::parse_record 349 ⟩ +≡
  if (out_str ≡ 0) {
    message_strm ≪ "ERROR!␣␣In␣‘parse_record’:␣␣‘string*␣out_str’␣==␣0." ≪ endl ≪
        "Exiting␣function␣unsucessfully␣with␣return␣value␣1.";
    AfxMessageBox(message_strm.str().c_str());
    message_strm.str("");
    return 1;
  }   /* if (out_str ≡ 0) */
  *out_str = "";   /* Clear *out_str. LDF 2006.05.29. */
```

**352.** Outer loop. [LDF 2006.05.29.]

⟨ Define **Metadata_Source** :: *parse_record* 349 ⟩ +≡

```
BOOL skipping = true; for ( ; ; ) { curr_char = in_strm→get();
```

**353.** Found EOF either before a < *record* > began, or if one has begun, before it ended. The former case is alright, the second is an error condition. [LDF 2006.05.30.]

⟨ Define **Metadata_Source** :: *parse_record* 349 ⟩ +≡

```
  if (curr_char ≡ EOF) {
    if (skipping) {
#if 0
      message_strm ≪ "In␣‘parse_record’:" ≪ endl ≪
          "Read␣‘EOF’␣from␣‘in_strm’␣before␣a␣‘<record>’␣began." ≪ endl ≪
          "Exiting␣function␣successfully␣with␣return␣value␣2.";
      AfxMessageBox(message_strm.str().c_str());
      message_strm.str("");
#endif
      return 2;
    }    /* if (skipping) */
    else {
      message_strm ≪ "ERROR!␣␣In␣‘parse_record’:" ≪ endl ≪
          "Read␣‘EOF’␣from␣‘in_strm’␣before␣a␣‘<record>’␣ended." ≪ endl ≪
          "Exiting␣function␣unsuccessfully␣with␣return␣value␣1.";
      AfxMessageBox(message_strm.str().c_str());
      message_strm.str("");
      return 1;
    }    /* else (¬skipping) */
  }    /* if (curr_char ≡ EOF) */
```

**354.** Found '<' while *skipping* ≡ *true*. [LDF 2006.05.29.]

⟨ Define **Metadata_Source** :: *parse_record* 349 ⟩ +≡

```
  if (skipping ∧ curr_char ≡ '<') {
#if 0
  log_file ≪ "curr_char␣==␣" ≪ curr_char ≪ ".␣␣Searching␣for␣‘<record>’.\n";
#endif
```

**355.** Discard everything until first < *record* > tag. [LDF 2006.05.29.]

Check the first character after '<': If it's not 'r' or 'R', just continue. If *in_strm*→*get*() returns EOF, just return 2. [LDF 2006.05.30.]

⟨ Define **Metadata_Source** :: *parse_record* 349 ⟩ +≡
```
  tag_ctr = 0;
  strcpy(tag_str, "");
  curr_char = in_strm→get();
  if (curr_char ≡ EOF) {
    return 2;
  }
  tag_str[tag_ctr] = curr_char;
  ++tag_ctr;
  if (¬(curr_char ≡ 'r' ∨ curr_char ≡ 'R')) {
#if 0
    log_file ≪ "‘curr_char’␣isn’t␣’r’␣or␣’R’,␣so␣continuing␣to␣skip." ≪ endl;
#endif
    continue;
  }
```

**356.**

⟨ Define **Metadata_Source** :: *parse_record* 349 ⟩ +≡
```
  while (tag_ctr < MAX_TAG_LENGTH ∧ curr_char ≠ '>') { curr_char = in_strm→get();
```

**357.** Error handling: *curr_char* ≡ EOF. [LDF 2006.05.30.]

⟨ Define **Metadata_Source** :: *parse_record* 349 ⟩ +≡
```
  if (curr_char ≡ EOF) {
    message_strm ≪ "ERROR!␣␣In␣‘parse_record’:" ≪ endl ≪
        "Read␣‘EOF’␣while␣collecting␣a␣tag." ≪ endl ≪
        "Exiting␣function␣unsuccessfully␣with␣return␣value␣1.";
    AfxMessageBox(message_strm.str().c_str());
    message_strm.str("");
    return 1;
  }    /* if (curr_char ≡ EOF) */
```

**358.**

⟨ Define **Metadata_Source** :: *parse_record* 349 ⟩ +≡
```
  tag_str[tag_ctr] = curr_char;
  ++tag_ctr; }    /* End of inner while LDF 2006.05.29. */
```

**359.** *tag_ctr* ≥ MAX_TAG_LENGTH

⟨ Define **Metadata_Source** :: *parse_record* 349 ⟩ +≡
```
  if (tag_ctr ≥ MAX_TAG_LENGTH) {
    message_strm ≪ "In␣‘parse_record’:␣␣" ≪ "‘tag_ctr’␣>=␣‘MAX_TAG_LENGTH’:" ≪ endl ≪
        "tag_ctr␣==␣" ≪ tag_ctr ≪ endl ≪ "Continuing␣to␣skip.";
    AfxMessageBox(message_strm.str().c_str());
    message_strm.str("");
    continue;
  }    /* if (tag_ctr ≥ MAX_TAG_LENGTH) */
```

**360.**

⟨ Define **Metadata_Source** :: *parse_record* 349 ⟩ +≡

```
  else {
    --tag_ctr;
    tag_str[tag_ctr] = '\0';
  }     /* else (tag_ctr < MAX_TAG_LENGTH) */
#if 0
  message_strm << "tag_str␣==␣" << tag_str;
  AfxMessageBox(message_strm.str().c_str());
  message_strm.str("");
#endif
  if (¬_stricmp(tag_str, "record")) {
#if 0
    AfxMessageBox("It's␣\"<record>\".");
#endif
    *out_str += "<record>";
    skipping = false;
  }
```

**361.**

⟨ Define **Metadata_Source** :: *parse_record* 349 ⟩ +≡

```
  else if (¬_strnicmp(tag_str, "resumptiontoken", 15)) {
#if 0
  temp_strm << "Found␣resumption␣token.␣␣tag_str␣==␣" << tag_str;
  AfxMessageBox(temp_strm.str().c_str());
  temp_strm.str("");
#endif
```

**362.**

**To Do**

Add error handling. [LDF 2006.07.04.]

---

⟨ Define **Metadata_Source** :: *parse_record* 349 ⟩ +≡

```
  int i; for (i = 0; ; ++i) { curr_char = in_strm→get(); if (curr_char ≡ '<') { resumption_token[i] = '\0';
#if 0
  temp_strm << "(Before␣loop):␣␣resumption_token␣==␣" << resumption_token;
  AfxMessageBox(temp_strm.str().c_str());
  temp_strm.str("");
#endif
```

**363.** Discard characters up to and including the closing '>'.

**To Do**

Add error handling. [LDF 2006.07.04.]

---

⟨ Define **Metadata_Source** :: *parse_record* 349 ⟩ +≡

```
  while (curr_char ≠ '>') {
    curr_char = in_strm→get();
  }
  break; }     /* if (curr_char = '<') */
```

**364.** Collecting characters and storing them in *resumption_token*. [LDF Undated.]

```
⟨ Define Metadata_Source :: parse_record 349 ⟩ +≡
  else {
    resumption_token[i] = curr_char;
  }
  }    /* for */
#if 0
  temp_strm ≪ "Resumption_token␣==␣" ≪ resumption_token;
  AfxMessageBox(temp_strm.str().c_str());
  temp_strm.str("");
#endif
  }    /* else if (¬_strnicmp(tag_str, "resumptiontoken", 15)) */
  else {
#if 0
    AfxMessageBox("It's␣not␣\"<record>\".");
#endif
    continue;
  }
  continue; }    /* if (skipping ∧ curr_char ≡ '<') */
```

**365.** Check tag. Exit function if it's "⟨/record⟩". [LDF 2006.05.29.]

```
⟨ Define Metadata_Source :: parse_record 349 ⟩ +≡
  if (¬skipping ∧ curr_char ≡ '<') { *out_str += curr_char;
  curr_char = in_strm→get();
```

**366.** Error handling: *curr_char* ≡ EOF. [LDF 2006.05.30.]

```
⟨ Define Metadata_Source :: parse_record 349 ⟩ +≡
  if (curr_char ≡ EOF) {
    message_strm ≪ "ERROR!␣␣In␣`parse_record':" ≪ endl ≪
        "Read␣`EOF'␣while␣collecting␣a␣closing␣tag." ≪ endl ≪
        "Exiting␣function␣unsuccessfully␣with␣return␣value␣1.";
    AfxMessageBox(message_strm.str().c_str());
    message_strm.str("");
    return 1;
  }    /* if (curr_char ≡ EOF) */
  if (curr_char ≠ '/') {
    *out_str += curr_char;
  }
  else if (curr_char ≡ '/') { *out_str += curr_char;
  strcpy(tag_str, "");
  tag_ctr = 0;
```

**367.** Check the first character after '<': If it's not 'r' or 'R', just continue. [LDF 2006.05.30.]

```
⟨ Define Metadata_Source :: parse_record 349 ⟩ +≡
  tag_ctr = 0;
  strcpy(tag_str, "");
  curr_char = in_strm→get();
```

**368.** Error handling: *curr_char* ≡ `EOF`. [LDF 2006.05.30.]

⟨ Define **Metadata_Source** :: *parse_record* 349 ⟩ +≡

```
  if (curr_char ≡ EOF) {
    message_strm ≪ "ERROR!␣␣In␣‘parse_record’:" ≪ endl ≪
        "Read␣‘EOF’␣while␣collecting␣a␣closing␣tag." ≪ endl ≪
        "Exiting␣function␣unsuccessfully␣with␣return␣value␣1.";
    AfxMessageBox(message_strm.str().c_str());
    message_strm.str("");
    return 1;
  }    /* if (curr_char ≡ EOF) */
  tag_str[tag_ctr] = curr_char;
  ++tag_ctr;
  if (¬(curr_char ≡ 'r' ∨ curr_char ≡ 'R')) {
#if 0
    log_file ≪ "‘curr_char’␣isn’t␣’r’␣or␣’R’,␣" ≪ "so␣continuing␣to␣collect␣text." ≪ endl;
#endif
    *out_str += curr_char;
    continue;
  }
  while (curr_char ≠ '>') { curr_char = in_strm→get();
```

**369.** Error handling: $curr\_char \equiv$ EOF. [LDF 2006.05.30.]

⟨ Define **Metadata_Source** :: *parse_record* 349 ⟩ +≡

```
  if (curr_char ≡ EOF) {
    message_strm ≪ "ERROR!␣␣In␣‘parse_record’:" ≪ endl ≪
        "Read␣‘EOF’␣while␣collecting␣a␣tag." ≪ endl ≪
        "Exiting␣function␣unsuccessfully␣with␣return␣value␣1.";
    AfxMessageBox(message_strm.str( ).c_str( ));
    message_strm.str("");
    return 1;
  }    /* if (curr_char ≡ EOF) */
  tag_str[tag_ctr++] = curr_char; }    /* while */
  tag_str[−−tag_ctr] = '\0';
#if 0
  message_strm ≪ "tag_str␣==␣" ≪ tag_str;
  log_file ≪ message_strm.str( ) ≪ endl;
  AfxMessageBox(message_strm.str( ).c_str( ));
  message_strm.str("");
#endif
  ∗out_str += tag_str;
  ∗out_str += '>';
  if (¬_stricmp(tag_str, "record")) {
#if 0
    log_file ≪ "It's␣\"</record>\".␣␣Exiting␣function." ≪ endl;
#endif
    return 0;
  }
#if 0
  else {
    log_file ≪ "It's␣not␣\"</record>\".␣␣Continuing." ≪ endl;
  }
#endif
  }    /* else if (curr_char ≡ '/') */
  }    /* if (¬skipping ∧ curr_char ≡ '<') */
  else
    if (skipping)    /* and curr_char ≠ '<'. */
    {
#if 0
      log_file ≪ "curr_char␣==␣" ≪ curr_char ≪ ".␣␣Continuing␣to␣skip.\n";
#endif
      continue;
    }
```

**370.** Not skipping and $curr\_char \neq$ '<'. [LDF 2006.05.29.]

⟨ Define **Metadata_Source** :: *parse_record* 349 ⟩ +≡

```
  else {
```

**371.** Special character encoding. [LDF Undated.]

**Log**

[LDF 2006.10.04.] Changed the conditional from **if** $(curr\_char \equiv$ 'Ã'$)$ to **if** $(curr\_char \equiv$ 'Â' $\lor$ $curr\_char \equiv$ 'Ã'$)$.

⟨ Define **Metadata_Source** :: *parse_record* 349 ⟩ +≡

```
if (curr_char ≡ 'Â' ∨ curr_char ≡ 'Ã') {
special_str[0] = curr_char;
curr_char = in_strm→get();
```

**372.** Error handling: $curr\_char \equiv$ `EOF`. [LDF 2006.05.30.]

⟨ Define **Metadata_Source** :: *parse_record* 349 ⟩ +≡

```
if (curr_char ≡ EOF) {
  message_strm ≪ "ERROR!␣␣In␣‘parse_record’:" ≪ endl ≪
      "Read␣‘EOF’␣while␣collecting␣a␣special␣character␣encoding." ≪ endl ≪
      "Exiting␣function␣unsuccessfully␣with␣return␣value␣1.";
  AfxMessageBox(message_strm.str().c_str());
  message_strm.str("");
  return 1;
}    /* if (curr_char ≡ EOF) */
special_str[1] = curr_char;
special_str[2] = '\0';
```

**373. Handle special characters.**
Handle special characters in **Metadata_Source::parse_record**.

See also ⟨ Special character formatting 11 ⟩. [LDF 2006.10.02.]

**Log**

[LDF 2006.10.26.] Added code.

[LDF 2006.10.30.] Removed the code for handling specific special characters. Now looking the encodings up in *special_char_encodings_map*.

[LDF 2006.10.30.] Now adding "XXX" to ∗*out_str*, if the special character coding is unknown. Formerly, the special character coding was added anyway, which caused errors when attempting to write it to the database.

---

```
⟨ Define Metadata_Source::parse_record 349 ⟩ +≡
  map⟨string, char⟩::iterator iter = special_char_encodings_map.find(string(special_str));
  if (iter ≡ special_char_encodings_map.end()) {
    message_strm ≪ "curr_char␣is␣an␣unknown␣special␣character:␣␣" ≪ curr_char ≪ endl ≪
        "special_str␣==␣" ≪ special_str ≪ endl ≪ "*out_str␣==␣" ≪ ∗out_str ≪ endl ≪
        "Adding␣\"XXX\"␣to␣‘*out_str’.";
    AfxMessageBox(message_strm.str().c_str());
    message_strm.str("");
    ∗out_str += "XXX";
  }
  else {
#if 0    /∗ 1 ∗/
    temp_strm ≪ "\"" ≪ special_str ≪ "\"␣==␣" ≪ iter→second;
    AfxMessageBox(temp_strm.str().c_str());
    temp_strm.str("");
#endif
    ∗out_str += iter→second;
  }
  }    /∗ if (curr_char ≡ 'Â' ∨ curr_char ≡ 'Ã') (Special character sequence). ∗/
```

**374.**

```
⟨ Define Metadata_Source::parse_record 349 ⟩ +≡
  else
    if (curr_char ≡ '\'') {
      ∗out_str += "''";
    }
    else {
      ∗out_str += curr_char;
    }
  }    /∗ else (Not skipping and curr_char ≠ '<'). ∗/
```

**375.** End of outer loop. [LDF 2006.05.29.]

```
⟨ Define Metadata_Source::parse_record 349 ⟩ +≡
  }    /∗ Outer while. ∗/
  }    /∗ End of Metadata_Source::parse_record definition. ∗/
```

**376. Putting class Metadata_Source together.** [LDF Undated.]

**377.** This is what's compiled.

⟨ Include files 20 ⟩
⟨ Initialize **static** constants in **class Metadata_Source** 317 ⟩
⟨ Define **Metadata_Source** functions 321 ⟩
⟨ Define **Metadata_Source** :: *parse_record* 349 ⟩

**378.** This is what's written to `mtdtsrc.h`.

⟨ `mtdtsrc.h` 378 ⟩ ≡
  ⟨ Preprocessor macro calls 17 ⟩
  ⟨ Declare **class Metadata_Source** 315 ⟩

**379. Selector (`selector.web`).** [LDF 2006.10.04.]

**Log**

[LDF 2006.10.04.] Created this file. It contains code formerly in `Selector.h`, `Selector.cpp`, `select_from_database.cpp`, `list_records.cpp`, `fill_table_streams.cpp`, `write_html_from_streams.cpp`, and `list_table.cpp`.

---

⟨ `selector.web` 379 ⟩ ≡ /* Empty section for use in cross-references. */

This code is cited in sections 6 and 9.

This code is used in section 881.

**380. Include files.**

⟨ Include files 20 ⟩ +≡

```
#include "stdafx.h"
```

**381. Preprocessor macro calls.**

⟨ Preprocessor macro calls 17 ⟩ +≡

```
#pragma once
```

**382. Preprocessor macro definitions.**

⟨ Preprocessor macro definitions 18 ⟩ +≡

**383. Forward declarations.** [LDF Undated.]

⟨ Forward declarations 383 ⟩ ≡

```
  class Dialog_2;
```

This code is used in section 502.

**384. Declare class Selector.** [LDF Undated.]

**Log**

[LDF 2006.08.01.] Added **friend** declaration for **class Dialog_2**.

---

⟨ Declare **class Selector** 384 ⟩ ≡

```
  class Selector {
    friend class Dialog_2;
  protected: Records curr_record;
    CDatabase *cdb;
    unsigned short use_date_type;
    unsigned short query_type;
    Records records;
    Records_Temp records_temp;
    Temp_IDs temp_ids;
    Temp_IDs_1 temp_ids_1;
    Contributors_Temp contributors_temp;
    Creators_Temp creators_temp;
```

**Descriptions_Temp** *descriptions_temp*;
**Identifiers_Temp** *identifiers_temp*;
**Languages_Temp** *languages_temp*;
**Publishers_Temp** *publishers_temp*;
**Rights_Temp** *rights_temp*;
**Subjects_Temp** *subjects_temp*;
**Titles_Temp** *titles_temp*;
**Types_Temp** *types_temp*;
**Contributors** *contributors*;
**Creators** *creators*;
**Subjects** *subjects*;
**Titles** *titles*;
**stringstream** *contributor_strm*;
**stringstream** *creator_strm*;
**stringstream** *dc_date_strm*;
**stringstream** *description_strm*;
**stringstream** *header_datestamp_strm*;
**stringstream** *identifier_strm*;
**stringstream** *language_strm*;
**stringstream** *publisher_strm*;
**stringstream** *rights_strm*;
**stringstream** *subject_strm*;
**stringstream** *title_strm*;
**stringstream** *type_strm*;
**stringstream** *temp_strm*;
**ofstream** *html_strm*;
**map**⟨**unsigned short**, **string**⟩ *sort_field_map*;
**map**⟨**unsigned short**, **string**⟩ *timespan_map*;

**public**:
⟨ Declare **Selector** functions 387 ⟩
**static const unsigned short** `QUERY_NULL_TYPE`;
**static const unsigned short** `QUERY_SEARCH`;
**static const unsigned short** `QUERY_LISTING`;
**static const unsigned short** `NULL_TIMESPAN`;
**static const unsigned short** `TODAY`;
**static const unsigned short** `YESTERDAY`;
**static const unsigned short** `THIS_WEEK`;
**static const unsigned short** `LAST_WEEK`;
**static const unsigned short** `THIS_MONTH`;
**static const unsigned short** `LAST_MONTH`;
**static const unsigned short** `LAST_6_MONTHS`;
**static const unsigned short** `THIS_YEAR`;
**static const unsigned short** `LAST_YEAR`;
**static const unsigned short** `LAST_2_YEARS`;
**static const unsigned short** `LAST_5_YEARS`;
**static const unsigned short** `LAST_10_YEARS`;
**static const unsigned short** `LAST_20_YEARS`;
**static const unsigned short** `ALL_RECORDS`;
**static const unsigned short** `USE_DC_DATE`;
**static const unsigned short** `USE_HEADER_DATESTAMP`;
**static const unsigned short** `SORT_ASCENDING`;
**static const unsigned short** `SORT_DESCENDING`;

```
    static const unsigned short SORT_FIELD_RECORD_ID;
    static const unsigned short SORT_FIELD_CREATOR;
    static const unsigned short SORT_FIELD_TITLE;
    static const unsigned short SORT_FIELD_DC_DATE;
    static const unsigned short SORT_FIELD_HEADER_DATESTAMP;
    static const unsigned int CONTRIBUTORS;
    static const unsigned int CREATORS;
    static const unsigned int DC_DATES;
    static const unsigned int DESCRIPTIONS;
    static const unsigned int HEADER_DATESTAMPS;
    static const unsigned int IDENTIFIERS;
    static const unsigned int LANGUAGES;
    static const unsigned int PUBLISHERS;
    static const unsigned int RIGHTS;
    static const unsigned int SUBJECTS;
    static const unsigned int TITLES;
    static const unsigned int TYPES;
    static const unsigned int BEG_OR_WHOLE_WORD;
    static const unsigned int WHOLE_WORD_ONLY;
    static const unsigned int WHOLE_OR_PARTIAL_WORD;
    static const unsigned int EXACT_MATCH;
    static const unsigned int IGNORE_CASE;
  };
```

This code is used in section 502.

**385. Initialize static constants in class Selector.** [LDF Undated.]

⟨ Initialize **static** constants in **class Selector** 385 ⟩ ≡

```
  const unsigned short Selector::QUERY_NULL_TYPE = 0;
  const unsigned short Selector::QUERY_SEARCH = 1;
  const unsigned short Selector::QUERY_LISTING = 2;
  const unsigned int Selector::CONTRIBUTORS = 1;
  const unsigned int Selector::CREATORS = 2;
  const unsigned int Selector::DC_DATES = 4;
  const unsigned int Selector::DESCRIPTIONS = 8;
  const unsigned int Selector::HEADER_DATESTAMPS = 16;
  const unsigned int Selector::IDENTIFIERS = 32;
  const unsigned int Selector::LANGUAGES = 64;
  const unsigned int Selector::PUBLISHERS = 128;
  const unsigned int Selector::RIGHTS = 256;
  const unsigned int Selector::SUBJECTS = 512;
  const unsigned int Selector::TITLES = 1024;
  const unsigned int Selector::TYPES = 2048;
  const unsigned int Selector::BEG_OR_WHOLE_WORD = 1;
  const unsigned int Selector::WHOLE_WORD_ONLY = 2;
  const unsigned int Selector::WHOLE_OR_PARTIAL_WORD = 4;
  const unsigned int Selector::EXACT_MATCH = 8;
  const unsigned int Selector::IGNORE_CASE = 16;
  const unsigned short Selector::NULL_TIMESPAN = 0;
  const unsigned short Selector::TODAY = 1;
  const unsigned short Selector::YESTERDAY = 2;
  const unsigned short Selector::THIS_WEEK = 3;
  const unsigned short Selector::LAST_WEEK = 4;
```

**const unsigned short Selector** :: THIS_MONTH = 5;
**const unsigned short Selector** :: LAST_MONTH = 6;
**const unsigned short Selector** :: LAST_6_MONTHS = 7;
**const unsigned short Selector** :: THIS_YEAR = 8;
**const unsigned short Selector** :: LAST_YEAR = 9;
**const unsigned short Selector** :: LAST_2_YEARS = 10;
**const unsigned short Selector** :: LAST_5_YEARS = 11;
**const unsigned short Selector** :: LAST_10_YEARS = 12;
**const unsigned short Selector** :: LAST_20_YEARS = 13;
**const unsigned short Selector** :: ALL_RECORDS = 14;
**const unsigned short Selector** :: USE_DC_DATE = 1;
**const unsigned short Selector** :: USE_HEADER_DATESTAMP = 2;
**const unsigned short Selector** :: SORT_ASCENDING = 1;
**const unsigned short Selector** :: SORT_DESCENDING = 2;
**const unsigned short Selector** :: SORT_FIELD_RECORD_ID = 1;
**const unsigned short Selector** :: SORT_FIELD_CREATOR = 2;
**const unsigned short Selector** :: SORT_FIELD_TITLE = 3;
**const unsigned short Selector** :: SORT_FIELD_DC_DATE = 4;
**const unsigned short Selector** :: SORT_FIELD_HEADER_DATESTAMP = 5;

This code is used in section 501.

**386. Functions.** [LDF 2006.10.04.]

**387. Constructor.**

⟨ Declare **Selector** functions 387 ⟩ ≡
**Selector**(**void**);

See also sections 395, 397, 417, 440, 456, and 485.

This code is used in section 384.

**388.**

⟨ Define **Selector** functions 388 ⟩ ≡
**Selector** :: **Selector**(**void**): *use_date_type*(USE_DC_DATE) {
*curr_record*.*Open*( );
*cdb* = *curr_record*.*m_pDatabase*;
**BOOL** *b*;

See also sections 389, 390, 391, 392, 393, 394, 396, 398, 399, 400, 401, 402, 404, 405, 406, 407, 408, 409, 410, 411, 412, 413, 414, 415, 416, 418, 419, 420, 421, 422, 423, 424, 425, 426, 427, 428, 429, 430, 431, 432, 433, 434, 435, 436, 437, 438, 439, 441, 442, 443, 444, 445, 446, 447, 448, 449, 450, 451, 452, 453, 454, 455, 457, 458, 459, 460, 461, 462, 463, 464, 465, 466, 467, 468, 469, 470, 471, 472, 473, 474, 475, 476, 477, 478, 479, 480, 481, 482, 483, 484, 486, 487, 488, 489, 490, 491, 492, 493, 494, 495, 496, 497, 498, and 499.

This code is used in section 501.

**389.** Test whether *cdb* is open. [LDF 2006.05.26.]

⟨ Define **Selector** functions 388 ⟩ +≡

```
  try {
    b = cdb→IsOpen( );
  }
  catch(CDBException *e)
  {
    temp_strm ≪ "Exception␣when␣calling␣‘cdb.IsOpen’:␣" ≪ e→m_strError ≪
        "Exiting␣‘Selector::Selector’␣" ≪ "unsuccessfully␣with␣return␣value␣-1.";
    AfxMessageBox(temp_strm.str( ).c_str( ));
    temp_strm.str("");
    e→Delete( );
    errno = 1;
    return;
  }   /* catch */
#if 0
  if (b) AfxMessageBox("‘cdb’␣is␣open.");
  else AfxMessageBox("‘cdb’␣isn’t␣open.");
#endif
```

**390.** Test whether *cdb* can be updated. [LDF 2006.05.26.]

⟨ Define **Selector** functions 388 ⟩ +≡

```
  try {
    b = cdb→CanUpdate( );
  }
  catch(CDBException *e)
  {
    temp_strm ≪ "Exception␣when␣calling␣‘cdb.CanUpdate’:␣" ≪ e→m_strError ≪ endl ≪
        "Exiting␣‘Selector::Selector’␣unsuccessfully␣" ≪ "with␣return␣value␣-1.";
    AfxMessageBox(temp_strm.str( ).c_str( ));
    temp_strm.str("");
    e→Delete( );
    errno = 1;
    return;
  }   /* catch */
#if 0
  if (b) AfxMessageBox("‘cdb’␣can␣be␣updated.");
  else AfxMessageBox("‘cdb’␣can␣be␣updated.");
#endif
```

**391.** Test whether transactions are supported for *cdb*. [LDF 2006.05.26.]

⟨ Define **Selector** functions 388 ⟩ +≡
```
try {
  b = cdb→CanTransact();
}
catch(CDBException *e)
{
  temp_strm ≪ "Exception␣when␣calling␣‘cdb.CanTransact’:␣" ≪ e→m_strError ≪ endl ≪
      "Exiting␣‘select_from_database’␣unsuccessfully␣" ≪ "with␣return␣value␣-1.";
  AfxMessageBox(temp_strm.str().c_str());
  temp_strm.str("");
  e→Delete();
  errno = 1;
  return;
}    /* catch */
#if 0
  if (b) AfxMessageBox("Transactions␣can␣be␣performed␣using␣‘cdb’.");
  else AfxMessageBox("Transactions␣cannot␣be␣performed␣using␣‘cdb’.");
#endif
```

**392.** Fill *sort_field_map*. [LDF Undated.]

⟨ Define **Selector** functions 388 ⟩ +≡
```
sort_field_map[SORT_FIELD_RECORD_ID] = "Record␣Number";
sort_field_map[SORT_FIELD_CREATOR] = "Creator";
sort_field_map[SORT_FIELD_TITLE] = "Title";
sort_field_map[SORT_FIELD_DC_DATE] = "Date␣(dc:date)";
sort_field_map[SORT_FIELD_HEADER_DATESTAMP] = "Date␣(header␣datestamp)";
```

**393.** Fill *timespan_map*. [LDF Undated.]

⟨ Define **Selector** functions 388 ⟩ +≡
```
timespan_map[NULL_TIMESPAN] = "No␣Timespan";
timespan_map[TODAY] = "Today";
timespan_map[YESTERDAY] = "Yesterday";
timespan_map[THIS_WEEK] = "This␣Week";
timespan_map[LAST_WEEK] = "Last␣Week";
timespan_map[THIS_MONTH] = "This␣Month";
timespan_map[LAST_MONTH] = "Last␣Month␣and␣This␣Month";
timespan_map[LAST_6_MONTHS] = "Last␣Six␣Months";
timespan_map[THIS_YEAR] = "This␣Year␣Only";
timespan_map[LAST_YEAR] = "Last␣Year␣and␣This␣Year";
timespan_map[LAST_2_YEARS] = "Last␣Two␣Years";
timespan_map[LAST_5_YEARS] = "Last␣Five␣Years";
timespan_map[LAST_10_YEARS] = "Last␣Ten␣Years";
timespan_map[LAST_20_YEARS] = "Last␣20␣Years";
timespan_map[ALL_RECORDS] = "No␣Limit";
```

**394.**

⟨ Define **Selector** functions 388 ⟩ +≡
*errno* = 0;
**return**; } /* End of **Selector** :: **Selector**(**void**) definition. */

**395. Destructor.**

⟨ Declare **Selector** functions 387 ⟩ +≡
∼**Selector**(**void**);

**396.**

⟨ Define **Selector** functions 388 ⟩ +≡
**Selector** :: ∼**Selector**(**void**)
{
  **if** (*curr_record*.*IsOpen*( )) *curr_record*.*Close*( );
  *cdb* = NULL;
}

**397. Select from database.** [LDF Undated.]

⟨ Declare **Selector** functions 387 ⟩ +≡
**int** *select_from_database*(
**const unsigned int** *select_value*,
**CString** &*search_str*,
**const unsigned int** *search_options*);

**398.**

```
⟨ Define Selector functions 388 ⟩ +≡
  int Selector::select_from_database(
  const unsigned int select_value,
  CString &search_str,
  const unsigned int search_options){
      int ret_val = 0;
      stringstream sql_strm;
      unsigned int search_arg = search_options & ~IGNORE_CASE;
#if 0   /* 1 */
      temp_strm << "search_arg == " << search_arg;
      AfxMessageBox(temp_strm.str().c_str());
      temp_strm.str("");
#endif
      temp_strm << "delete temp_ids exec search_for_records " << "'" << search_str << "', " <<
          search_arg << ", ";
      if (select_value & CONTRIBUTORS) temp_strm << "1, ";
      else temp_strm << "0, ";
      if (select_value & CREATORS) temp_strm << "1, ";
      else temp_strm << "0, ";
      if (select_value & DESCRIPTIONS) temp_strm << "1, ";
      else temp_strm << "0, ";
      if (select_value & IDENTIFIERS) temp_strm << "1, ";
      else temp_strm << "0, ";
      if (select_value & LANGUAGES) temp_strm << "1, ";
      else temp_strm << "0, ";
      if (select_value & PUBLISHERS) temp_strm << "1, ";
      else temp_strm << "0, ";
      if (select_value & RIGHTS) temp_strm << "1, ";
      else temp_strm << "0, ";
      if (select_value & SUBJECTS) temp_strm << "1, ";
      else temp_strm << "0, ";
      if (select_value & TITLES) temp_strm << "1, ";
      else temp_strm << "0, ";
      if (select_value & TYPES) temp_strm << "1";
      else temp_strm << "0";
```

**399.**

```
⟨ Define Selector functions 388 ⟩ +≡
  try {
    cdb->ExecuteSQL(temp_strm.str().c_str());
  }
```

**400.** The error code is in *e→m_nRetCode*. [LDF Undated.]

⟨ Define **Selector** functions 388 ⟩ +≡

```
catch (CDBException *e)
{
  temp_strm ≪ "Exception␣when␣calling␣‘cdb->ExecuteSQL’." ≪ endl ≪ "Error␣code␣=␣" ≪
      e→m_nRetCode ≪ endl ≪ "Exception␣==␣" ≪ e→m_strError ≪ endl ≪
      "Exiting␣‘select_from_database’␣unsuccessfully␣with␣return␣value␣-1.";
  AfxMessageBox(temp_strm.str().c_str());
  temp_strm.str("");
  e→Delete();
  return −1;
}   /* catch */
temp_strm.str("");   /* Do not delete! */
```

**401.** Get items from **Temp_IDs** table. [LDF Undated.]

⟨ Define **Selector** functions 388 ⟩ +≡

```
long record_ctr = 0;
temp_ids.Open();
while (¬temp_ids.IsEOF()) {
  temp_ids.MoveNext();
}
record_ctr = temp_ids.GetRecordCount();
html_strm.open("results.html");
html_strm ≪ "<!--␣results.html␣-->" ≪ endl ≪ endl ≪ "<!DOCTYPE␣HTML␣PUBLIC␣" ≪
    "\"-//W3C//DTD␣HTML␣4.01␣Transitional//EN\"" ≪ endl ≪
    "\"http://www.w3.org/TR/html4/loose.dtd\">" ≪ endl ≪ "<!␣file:///u|/results.html>" ≪
    "<html>\n<head>\n<title>\nSearch␣Results\n</title>" ≪ endl ≪
    "</head>\n<body>\n<font␣color=\"black\">" ≪ "\n<h1␣align=\"center\">\n<a␣name=\"Top\\
    ">␣Search␣Results" ≪ "\n</a>\n</h1>\n</font>\n\n" ≪ copyright_html_str ≪
    "<hr␣size=\"2\"␣color=\"black\">\n\n" ≪ "<p>\n<b>Search␣string:</b>␣<q>" ≪
    search_str ≪ "</q>\n<br>\n<br>\n";
BOOL first_field = TRUE;
html_strm ≪ "<b>Fields␣searched:</b>␣";
if (select_value & CONTRIBUTORS) {
  if (first_field) first_field = FALSE;
  else html_strm ≪ ",␣";
  html_strm ≪ "Contributors";
}
if (select_value & CREATORS) {
  if (first_field) first_field = FALSE;
  else html_strm ≪ ",␣";
  html_strm ≪ "Creators";
}
if (select_value & DESCRIPTIONS) {
  if (first_field) first_field = FALSE;
  else html_strm ≪ ",␣";
  html_strm ≪ "Descriptions";
}
if (select_value & IDENTIFIERS) {
  if (first_field) first_field = FALSE;
  else html_strm ≪ ",␣";
  html_strm ≪ "Identifiers";
}
if (select_value & LANGUAGES) {
  if (first_field) first_field = FALSE;
  else html_strm ≪ ",␣";
  html_strm ≪ "Languages";
}
if (select_value & PUBLISHERS) {
  if (first_field) first_field = FALSE;
  else html_strm ≪ ",␣";
  html_strm ≪ "Publishers";
}
if (select_value & RIGHTS) {
  if (first_field) first_field = FALSE;
```

```
    else html_strm ≪ ",␣";
    html_strm ≪ "Rights";
  }
  if (select_value & SUBJECTS) {
    if (first_field) first_field = FALSE;
    else html_strm ≪ ",␣";
    html_strm ≪ "Subjects";
  }
  if (select_value & TITLES) {
    if (first_field) first_field = FALSE;
    else html_strm ≪ ",␣";
    html_strm ≪ "Titles";
  }
  if (select_value & TYPES) {
    if (first_field) first_field = FALSE;
    else html_strm ≪ ",␣";
    html_strm ≪ "Types";
  }
  html_strm ≪ "\n\n<br>\n<br>\n";
```

**402.** Search Options. [LDF Undated.]

```
⟨ Define Selector functions 388 ⟩ +≡
  html_strm ≪ "<b>Search␣Options:</b>␣";
  if (search_options & BEG_OR_WHOLE_WORD) html_strm ≪ "Beginning␣of␣word␣or␣whole␣word.␣";
  else if (search_options & WHOLE_WORD_ONLY) html_strm ≪ "Whole␣word␣only.␣";
  else if (search_options & WHOLE_OR_PARTIAL_WORD) html_strm ≪ "Whole␣or␣partial␣word.␣";
  else if (search_options & EXACT_MATCH) html_strm ≪ "Exact␣match.␣";
  else html_strm ≪ "Beginning␣of␣word␣or␣whole␣word.␣";
  if (search_options & IGNORE_CASE) html_strm ≪ "Ignoring␣case.";
  else html_strm ≪ "Not␣ignoring␣case␣(Not␣yet␣implemented!).";
  html_strm ≪ "\n\n<br>\n<br>\n\n";
  if (record_ctr ≤ 0) {
    html_strm ≪ "No␣records␣found.\n</p><br><br>\n" ≪
        "<hr␣size=\"2\"␣color=\"black\">\n\n" ≪ "</font>\n</body>\n</html>\n";
    temp_ids.Close();
    if (records_temp.IsOpen()) records_temp.Close();
    html_strm.close();
    return 0;
  }
  else {
    html_strm ≪ "<b>Records␣found:</b>␣" ≪ record_ctr ≪ "</p>\n\n";
    temp_ids.MoveFirst();
  }
  html_strm ≪ "<hr␣size=\"2\"␣color=\"black\">\n\n";
```

**403.** Open record sets for the temporary tables. [LDF Undated.]

**404.**

⟨ Define **Selector** functions 388 ⟩ +≡

```
contributors_temp.Open();
creators_temp.Open();
descriptions_temp.Open();
identifiers_temp.Open();
languages_temp.Open();
publishers_temp.Open();
rights_temp.Open();
subjects_temp.Open();
titles_temp.Open();
types_temp.Open();
```

**405.** Loop through *temp_ids* and fill the other temporary tables. [LDF Undated.]

---

**Log**

[LDF 2006.08.29.] !! BUG FIX: Now closing and reopening *records_temp* instead of calling **CRecordset** :: *Requery*( ). I had problems with the latter function in ZTest. The use of *Requery* was causing the problem that large data sets weren't sorted completely until the search had been performed two or three times.

---

⟨ Define **Selector** functions 388 ⟩ +≡

```
int temp_ctr = 1; while (¬temp_ids.IsEOF()) { sql_strm ≪ "delete␣records_temp\
    ␣insert␣records_temp␣" ≪ "select␣*␣from␣records␣where␣record_id␣=␣" ≪
    temp_ids.m_temp_id;
cdb→ExecuteSQL(sql_strm.str().c_str());
sql_strm.str("");
```

**406.** This doesn't seem to be needed anymore. Leaving the code here, just in case. [LDF 2006.08.29.]

⟨ Define **Selector** functions 388 ⟩ +≡

```
#if 0    /* 1 */
  if (records_temp.IsOpen()) records_temp.Close();
#endif
```

**407.**

⟨ Define **Selector** functions 388 ⟩ +≡

```
records_temp.Open();
html_strm ≪ "<h3>\n<a␣name=\"Result_" ≪ temp_ctr ≪ "\">␣Result␣" ≪ temp_ctr++ ≪
    ".</a>\n</h3>\n\n" ≪ "<p>\n<b>Record␣" ≪ records_temp.m_record_id ≪ ".</b>\n<br>\n";
ret_val = fill_table_streams(cdb);
```

**408.** Error handling: *fill_table_streams* failed. [LDF Undated.]

⟨ Define **Selector** functions 388 ⟩ +≡
```
if (ret_val ≡ −1) {
    temp_strm ≪ "ERROR!␣␣In␣‘Selector::select_from_database’:" ≪ endl ≪
        "‘Selector::fill_table_streams’␣failed,␣returning␣-1." ≪ endl ≪
        "Exiting␣function␣unsuccessfully␣with␣return␣value␣-1.";
    AfxMessageBox(temp_strm.str().c_str());
    temp_strm.str("");
    temp_ids.Close();
    html_strm.close();
    if (records_temp.IsOpen()) records_temp.Close();
    return −1;
}   /* if (ret_val ≡ −1) (fill_table_streams failed.) */
#if 0   /* 1 */
  temp_strm ≪ "records_temp.m_record_id␣==␣" ≪ records_temp.m_record_id ≪ endl;
  AfxMessageBox(temp_strm.str().c_str());
  temp_strm.str("");
#endif
```

**409.** Write from streams to *html_strm*. [LDF Undated.]

⟨ Define **Selector** functions 388 ⟩ +≡
```
ret_val = write_html_from_streams(QUERY_SEARCH, select_value, &search_str);
```

**410.** Error handling: *write_html_from_streams* failed.

⟨ Define **Selector** functions 388 ⟩ +≡
```
if (ret_val ≠ 0) {
    temp_strm ≪ "ERROR!␣␣In␣‘Selector::select_from_database’:" ≪ endl ≪
        "‘Selector::write_html_from_streams’␣failed,␣returning␣" ≪ ret_val ≪ "." ≪ endl ≪
        "Exiting␣function␣unsuccessfully␣with␣return␣value␣-1.";
    AfxMessageBox(temp_strm.str().c_str());
    temp_strm.str("");
    temp_ids.Close();
    html_strm.close();   /* Currently, there is no error return from write_html_from_streams. If I add
        one, and I close these record sets, I'll have to remove this code. LDF 2006.06.27. */
#if 0   /* 1 */
    if (records_temp.IsOpen())
#endif
      records_temp.Close();
    contributors_temp.Close();
    creators_temp.Close();
    descriptions_temp.Close();
    identifiers_temp.Close();
    languages_temp.Close();
    publishers_temp.Close();
    rights_temp.Close();
    subjects_temp.Close();
    titles_temp.Close();
    types_temp.Close();
    return −1;
}   /* if (ret_val ≠ 0) (write_html_from_streams failed.) */
```

**411.** Move to next record. [LDF Undated.]

⟨ Define **Selector** functions 388 ⟩ +≡
```
temp_ids.MoveNext();
records_temp.Close(); }   /* while (Iterating through records_temp) */
```

**412.** Write final code to *html_strm* and close it. [LDF Undated.]

⟨ Define **Selector** functions 388 ⟩ +≡
```
html_strm << "\n</font>\n</body>\n</html>\n";
html_strm.close();
```

**413.** Close record sets. [LDF Undated.]

⟨ Define **Selector** functions 388 ⟩ +≡
```
temp_ids.Close();
```

**414.** WARNING! *records_temp* shouldn't be open at this point. [LDF 2006.08.29.]

⟨ Define **Selector** functions 388 ⟩ +≡
```
if (records_temp.IsOpen()) {
  temp_strm << "WARNING!  `records_temp' is open at the end of " <<
      "`Selector::select_from_database'." << "It shouldn't be.  Check this!";
  AfxMessageBox(temp_strm.str().c_str());
  temp_strm.str("");
  records_temp.Close();
}   /* if (records_temp.IsOpen()) */
```

**415.**

⟨ Define **Selector** functions 388 ⟩ +≡
```
contributors_temp.Close();
creators_temp.Close();
descriptions_temp.Close();
identifiers_temp.Close();
languages_temp.Close();
publishers_temp.Close();
rights_temp.Close();
subjects_temp.Close();
titles_temp.Close();
types_temp.Close();
```

**416.** Exit *select_from_database*, returning the number of records.

⟨ Define **Selector** functions 388 ⟩ +≡
```
return record_ctr; }   /* End of Selector::select_from_database definition. */
```

**417. List records.** [LDF Undated.]

⟨ Declare **Selector** functions 387 ⟩ +≡
```
int list_records(
unsigned short sort_field,
unsigned short sort_order,
unsigned short timespan,
BOOL suppress_duplicate_records);
```

**418.**

⟨ Define **Selector** functions 388 ⟩ +≡

```
int Selector::list_records(unsigned short sort_field, unsigned short sort_order, unsigned short
        timespan, BOOL suppress_duplicate_records){
    if (sort_field < SORT_FIELD_RECORD_ID ∨ sort_field > SORT_FIELD_HEADER_DATESTAMP) {
      temp_strm ≪ "WARNING!␣␣In␣`Selector::list_records':" ≪ endl ≪
          "`sort_field'␣has␣invalid␣value:␣" ≪ sort_field ≪ endl ≪
          "Setting␣it␣to␣`SORT_FIELD_RECORD_ID'␣and␣continuing.";
      AfxMessageBox(temp_strm.str().c_str());
      temp_strm.str("");
      sort_field = SORT_FIELD_RECORD_ID;
    }
    stringstream sql_strm;
    sql_strm ≪ "delete␣temp_ids␣insert␣temp_ids␣" ≪
        "select␣R.record_id␣from␣records␣as␣R␣";
```

**419.** Timespan.

⟨ Define **Selector** functions 388 ⟩ +≡

```
int day_of_month;
int day_of_week;
int month_of_year;
CTime t0;
stringstream timespan_strm;
timespan_strm.str("");
string date_type_str;
if (use_date_type ≡ USE_DC_DATE) date_type_str = "dc_date";
else date_type_str = "header_datestamp";
if (timespan ≤ NULL_TIMESPAN ∨ timespan > ALL_RECORDS) {
  temp_strm ≪ "WARNING!␣␣In␣‘Selector::list_records’:" ≪ endl ≪
      "‘timespan’␣has␣invalid␣value:␣" ≪ timespan ≪ endl ≪
      "Setting␣‘timespan’␣to␣‘ALL_RECORDS’␣and␣continuing.";
  AfxMessageBox(temp_strm.str().c_str());
  temp_strm.str("");
  timespan = ALL_RECORDS;
}
if (timespan ≡ ALL_RECORDS) ;    /* Do nothing. */
else if (timespan ≡ TODAY) {
  t0 = CTime::GetCurrentTime();
  timespan_strm ≪ date_type_str ≪ "␣>=␣convert(datetime,␣'" ≪
      t0.Format("%Y-%m-%d␣00:00:00") ≪ "',␣121)␣";
}
else if (timespan ≡ YESTERDAY) {
  t0 = CTime::GetCurrentTime();
  t0 −= CTimeSpan(1, 0, 0, 0);
  timespan_strm ≪ date_type_str ≪ "␣>=␣convert(datetime,␣'" ≪
      t0.Format("%Y-%m-%d␣00:00:00") ≪ "',␣121)␣";
}
else if (timespan ≡ THIS_WEEK) {
  t0 = CTime::GetCurrentTime();
  day_of_week = t0.GetDayOfWeek() − 1;
  t0 −= CTimeSpan(day_of_week, 0, 0, 0);
  timespan_strm ≪ date_type_str ≪ "␣>=␣convert(datetime,␣'" ≪
      t0.Format("%Y-%m-%d␣00:00:00") ≪ "',␣121)␣";
}
else if (timespan ≡ LAST_WEEK) {
  t0 = CTime::GetCurrentTime();
  day_of_week = t0.GetDayOfWeek() − 1;
  t0 −= CTimeSpan(7 + day_of_week, 0, 0, 0);
  timespan_strm ≪ date_type_str ≪ "␣>=␣convert(datetime,␣'" ≪
      t0.Format("%Y-%m-%d␣00:00:00") ≪ "',␣121)␣";
}
else if (timespan ≡ THIS_MONTH) {
  t0 = CTime::GetCurrentTime();
  day_of_month = t0.GetDay();
  t0 −= CTimeSpan(day_of_month − 1, 0, 0, 0);
  timespan_strm ≪ date_type_str ≪ "␣>=␣convert(datetime,␣'" ≪
      t0.Format("%Y-%m-%d␣00:00:00") ≪ "',␣121)␣";
```

```
}
else if (timespan ≡ LAST_MONTH) {
  t0 = CTime::GetCurrentTime();
  day_of_month = t0.GetDay();
  t0 −= CTimeSpan(day_of_month, 0, 0, 0);
  day_of_month = t0.GetDay();
  t0 −= CTimeSpan(day_of_month − 1, 0, 0, 0);
  timespan_strm ≪ date_type_str ≪ "␣>=␣convert(datetime,␣'" ≪
      t0.Format("%Y-%m-%d␣00:00:00") ≪ "',␣121)␣";
}
else if (timespan ≡ LAST_6_MONTHS) {
  t0 = CTime::GetCurrentTime();
  month_of_year = t0.GetMonth();
  if (month_of_year > 5) {
    CTime t1(t0.GetYear(), month_of_year − 5, 1, 0, 0, 0);
    t0 = t1;
  }
  else {
    CTime t1(t0.GetYear() − 1, month_of_year + 7, 1, 0, 0, 0);
    t0 = t1;
  }
  timespan_strm ≪ date_type_str ≪ "␣>=␣convert(datetime,␣'" ≪
      t0.Format("%Y-%m-%d␣00:00:00") ≪ "',␣121)␣";
}
else if (timespan ≡ THIS_YEAR) {
  t0 = CTime::GetCurrentTime();
  CTime t1(t0.GetYear(), 1, 1, 0, 0, 0);
  t0 = t1;
  timespan_strm ≪ date_type_str ≪ "␣>=␣convert(datetime,␣'" ≪
      t0.Format("%Y-%m-%d␣00:00:00") ≪ "',␣121)␣";
}
else if (timespan ≡ LAST_YEAR) {
  t0 = CTime::GetCurrentTime();
  CTime t1(t0.GetYear() − 1, 1, 1, 0, 0, 0);
  t0 = t1;
  timespan_strm ≪ date_type_str ≪ "␣>=␣convert(datetime,␣'" ≪
      t0.Format("%Y-%m-%d␣00:00:00") ≪ "',␣121)␣";
}
else if (timespan ≡ LAST_2_YEARS) {
  t0 = CTime::GetCurrentTime();
  CTime t1(t0.GetYear() − 2, 1, 1, 0, 0, 0);
  t0 = t1;
  timespan_strm ≪ date_type_str ≪ "␣>=␣convert(datetime,␣'" ≪
      t0.Format("%Y-%m-%d␣00:00:00") ≪ "',␣121)␣";
}
else if (timespan ≡ LAST_5_YEARS) {
  t0 = CTime::GetCurrentTime();
  CTime t1(t0.GetYear() − 5, 1, 1, 0, 0, 0);
  t0 = t1;
```

```
    timespan_strm << date_type_str << " >= convert(datetime, '" <<
        t0.Format("%Y-%m-%d 00:00:00") << "', 121) ";
  }
  else if (timespan ≡ LAST_10_YEARS) {
    t0 = CTime::GetCurrentTime();
    CTime t1(t0.GetYear() − 10, 1, 1, 0, 0, 0);
    t0 = t1;
    timespan_strm << date_type_str << " >= convert(datetime, '" <<
        t0.Format("%Y-%m-%d 00:00:00") << "', 121) ";
  }
  else if (timespan ≡ LAST_20_YEARS) {
    t0 = CTime::GetCurrentTime();
    CTime t1(t0.GetYear() − 20, 1, 1, 0, 0, 0);
    t0 = t1;
    timespan_strm << date_type_str << " >= convert(datetime, '" <<
        t0.Format("%Y-%m-%d 00:00:00") << "', 121) ";
  }
  unsigned int select_value = 0;
  if (sort_field ≡ SORT_FIELD_RECORD_ID) {
#if 0
    temp_strm << "'sort_field' == 'SORT_FIELD_RECORD_ID'.";
    AfxMessageBox(temp_strm.str().c_str());
    temp_strm.str("");
#endif
    if (timespan ≠ ALL_RECORDS) sql_strm << "where " << timespan_strm.str();
    sql_strm << "order by record_id ";
  }
  else if (sort_field ≡ SORT_FIELD_CREATOR) {
#if 0
    temp_strm << "'sort_field' == 'SORT_FIELD_CREATOR.";
    AfxMessageBox(temp_strm.str().c_str());
    temp_strm.str("");
#endif
    sql_strm << ", Creators as C, Records_Creators as RC " <<
        "where R.record_id = RC.record_id " << "and C.creator_id = RC.creator_id ";
    if (timespan ≠ ALL_RECORDS) sql_strm << "and " << timespan_strm.str();
    sql_strm << "order by C.dc_creator ";
    select_value = CREATORS;
  }
  else if (sort_field ≡ SORT_FIELD_TITLE) {
#if 0
    temp_strm << "'sort_field' == 'SORT_FIELD_TITLE.";
    AfxMessageBox(temp_strm.str().c_str());
    temp_strm.str("");
#endif
    sql_strm << ", Titles as T " << "where R.record_id = T.record_id ";
    if (timespan ≠ ALL_RECORDS) sql_strm << "and " << timespan_strm.str();
    sql_strm << "order by T.dc_title ";
    select_value = TITLES;
  }
  else if (sort_field ≡ SORT_FIELD_DC_DATE) {
```

```
#if 0
    temp_strm ≪ "‘sort_field’␣==␣‘SORT_FIELD_DC_DATE.";
    AfxMessageBox(temp_strm.str().c_str());
    temp_strm.str("");
#endif
    if (timespan ≠ ALL_RECORDS) sql_strm ≪ "where␣" ≪ timespan_strm.str();
    sql_strm ≪ "order␣by␣dc_date␣";
    select_value = TITLES;
  }
  else if (sort_field ≡ SORT_FIELD_HEADER_DATESTAMP) {
#if 0
    temp_strm ≪ "‘sort_field’␣==␣‘SORT_FIELD_HEADER_DATESTAMP.";
    AfxMessageBox(temp_strm.str().c_str());
    temp_strm.str("");
#endif
    if (timespan ≠ ALL_RECORDS) sql_strm ≪ "where␣" ≪ timespan_strm.str();
    sql_strm ≪ "order␣by␣header_datestamp␣";
  }
```

**420.** Sort order.

```
⟨ Define Selector functions 388 ⟩ +≡
  if (¬(sort_order ≡ SORT_ASCENDING ∨ sort_order ≡ SORT_DESCENDING)) {
    temp_strm ≪ "WARNING!␣␣In␣‘Selector::list_records’:" ≪ endl ≪
        "‘sort_order’␣has␣invalid␣value:␣" ≪ sort_order ≪ endl ≪
        "Setting␣it␣to␣‘SORT_ASCENDING’␣and␣continuing.";
    AfxMessageBox(temp_strm.str().c_str());
    temp_strm.str("");
    sort_order = SORT_ASCENDING;
  }
  if (sort_order ≡ SORT_ASCENDING) sql_strm ≪ "asc␣";
  else if (sort_order ≡ SORT_DESCENDING) sql_strm ≪ "desc␣";
```

**421.** Pass the command stored in *sql_strm* to SQL-Server.

```
⟨ Define Selector functions 388 ⟩ +≡
#if 0
  AfxMessageBox("Error␣after␣here␣0.");
  temp_strm ≪ "sql_strm.str()␣==␣" ≪ sql_strm.str();
  AfxMessageBox(temp_strm.str().c_str());
  temp_strm.str("");
#endif
  try {
    cdb→ExecuteSQL(sql_strm.str().c_str());
  }
```

**422.** The error code is in *e→m_nRetCode*. [LDF Undated.]

⟨ Define **Selector** functions 388 ⟩ +≡

```
catch(CDBException *e)
{
  temp_strm << "Exception when calling `cdb->ExecuteSQL'." << endl << "Error code = " <<
      e->m_nRetCode << endl << "Exception == " << e->m_strError << endl <<
      "Exiting `list_records' unsuccessfully with return value -1.";
  AfxMessageBox(temp_strm.str().c_str());
  temp_strm.str("");
  e->Delete();
  return -1;
}   /* catch */
sql_strm.str("");
temp_ids.Open();
long record_ctr = 0;
while (!temp_ids.IsEOF()) {
  temp_ids.MoveNext();
}
record_ctr = temp_ids.GetRecordCount();
#if 1
  temp_strm << "Total records:  " << record_ctr;
  AfxMessageBox(temp_strm.str().c_str());
  temp_strm.str("");
#endif
```

**423.** Open *html_strm* and write header.

⟨ Define **Selector** functions 388 ⟩ +≡

```
html_strm.open("results.html");
html_strm << "<!-- results.html -->" << endl << endl << "<!DOCTYPE HTML PUBLIC " <<
    "\"-//W3C//DTD HTML 4.01 Transitional//EN\"" << endl <<
    "\"http://www.w3.org/TR/html4/loose.dtd\">" << endl << "<! file:///u|/results.html>" <<
    "<html>\n<head>\n<title>\nRecords List\n</title>" << endl <<
    "</head>\n<body>\n<font color=\"black\">" << "\n<h1 align=\"center\">\n<a name=\"Top\\
    "> Records List" << "\n</a>\n</h1>\n</font>\n\n" << copyright_html_str <<
    "<hr size=\"2\" color=\"black\">\n\n" << "<p>\n<b>Sort Field:</b> " <<
    sort_field_map[sort_field] << "\n<br>\n<br>\n" << "<b>Sort Order:</b> ";
if (sort_order == SORT_ASCENDING) html_strm << "Ascending";
else html_strm << "Descending";
html_strm << "\n<br>\n<br>\n" << "<b>Timespan:</b> " << timespan_map[timespan] <<
    "\n<br>\n<br>\n";
```

**424.** Duplicate Records. [LDF Undated.]
Currently, there can only be duplicate records if sorting by creators. [LDF 2006.07.03.]

⟨ Define **Selector** functions 388 ⟩ +≡

```
html_strm ≪ "<b>Duplicate␣records:</b>␣";
if (sort_field ≡ SORT_FIELD_CREATOR) {
  if (suppress_duplicate_records) html_strm ≪ "Suppressed";
  else html_strm ≪ "Not␣suppressed";
}   /* if (sort_field ≡ SORT_FIELD_CREATOR) */
else   /* sort_field ≠ SORT_FIELD_CREATOR */
{
  html_strm ≪ "Not␣applicable";
}   /* else (sort_field ≠ SORT_FIELD_CREATOR) */
html_strm ≪ "\n<br>\n<br>\n";
if (record_ctr ≤ 0) {
  html_strm ≪ "No␣records␣found.\n</p><br><br>\n" ≪
      "<hr␣size=\"2\"␣color=\"black\">\n\n" ≪ "</font>\n</body>\n</html>\n";
  temp_ids.Close();
#if 0
  records_temp.Close();
    /* Closing records_temp here causes a run-time error. I don't know why. LDF 2006.06.28. */
#endif
  html_strm.close();
  return 0;
}
else   /* (record_ctr > 0) */
{
```

**425.** Recount records, if if *suppress_duplicate_records* ≡ `TRUE`. [LDF Undated.]
Currently, there can only be duplicate records if sorting by creators. [LDF 2006.07.03.]

⟨ Define **Selector** functions 388 ⟩ +≡

```
if (suppress_duplicate_records ∧ sort_field ≡ SORT_FIELD_CREATOR) {
  temp_ids_1.Open(CRecordset::dynaset, "select␣distinct␣temp_id␣from␣temp_ids");
  while (¬temp_ids_1.IsEOF()) {
    temp_ids_1.MoveNext();
  }
  record_ctr = temp_ids_1.GetRecordCount();
  temp_strm ≪ "Unique␣records␣found:␣␣" ≪ record_ctr;
  AfxMessageBox(temp_strm.str().c_str());
  temp_strm.str("");
  temp_ids_1.Close();
}   /* if (suppress_duplicate_records ∧ sort_field ≡ SORT_FIELD_CREATOR) */
html_strm ≪ "<b>Records␣found:</b>␣" ≪ record_ctr ≪ "</p>\n\n";
temp_ids.MoveFirst(); }   /* else ((record_ctr > 0)) */
html_strm ≪ "<hr␣size=\"2\"␣color=\"black\">\n\n";
```

**426.** Open record sets for the temporary tables. [LDF Undated.]

⟨ Define **Selector** functions 388 ⟩ +≡
```
records_temp.Open();
contributors_temp.Open();
creators_temp.Open();
descriptions_temp.Open();
identifiers_temp.Open();
languages_temp.Open();
publishers_temp.Open();
rights_temp.Open();
subjects_temp.Open();
titles_temp.Open();
types_temp.Open();
```

**427.** Loop through *temp_ids* and fill the other temporary tables. [LDF Undated.]

⟨ Define **Selector** functions 388 ⟩ +≡
```
int ret_val;
unsigned long temp_ctr = 1;
set<unsigned long> used_ids;
set<unsigned long>::iterator iter; while (¬temp_ids.IsEOF()) {
```

**428.** Eliminate duplicates if *suppress_duplicate_records* ≡ `TRUE`∧*sort_field* ≡ `SORT_FIELD_CREATOR`. [LDF Undated.]
Currently, there can only be duplicate records if sorting by creators. [LDF 2006.07.03.]

⟨ Define **Selector** functions 388 ⟩ +≡
```
if (suppress_duplicate_records ∧ sort_field ≡ SORT_FIELD_CREATOR) {
    iter = used_ids.find(temp_ids.m_temp_id);
```

**429.** Record not used yet. [LDF Undated.]

⟨ Define **Selector** functions 388 ⟩ +≡
```
  if (iter ≡ used_ids.end()) {
    used_ids.insert(temp_ids.m_temp_id);
#if 0
    temp_strm << "id␣" << temp_ids.m_temp_id << "␣not␣used␣yet.";
    AfxMessageBox(temp_strm.str().c_str());
    temp_strm.str("");
#endif
  }    /* if (iter ≡ used_ids.end()) */
```

**430.** Record already used. [LDF Undated.]

```
⟨ Define Selector functions 388 ⟩ +≡
  else    /* (iter ≠ used_ids.end()) */
  {
#if 0
    temp_strm << "id␣" << temp_ids.m_temp_id << "␣used␣already.␣␣Continuing";
    AfxMessageBox(temp_strm.str().c_str());
    temp_strm.str("");
#endif
    temp_ids.MoveNext();
    continue;
  }    /* else (iter ≠ used_ids.end()) */
  }    /* if (suppress_duplicate_records ∧ sort_field ≡ SORT_FIELD_CREATOR) */
  sql_strm << "delete␣records_temp␣insert␣records_temp␣" <<
      "select␣*␣from␣records␣where␣record_id␣=␣" << temp_ids.m_temp_id;
#if 0
  temp_strm << "Error␣after␣here␣1.\n";
  temp_strm << "sql_strm␣==␣" << sql_strm.str();
  AfxMessageBox(temp_strm.str().c_str());
  temp_strm.str("");
#endif
  try {
    cdb→ExecuteSQL(sql_strm.str().c_str());
  }
```

**431.** The error code is in $e \rightarrow m\_nRetCode$.

```
⟨ Define Selector functions 388 ⟩ +≡
  catch(CDBException *e)
  {
    temp_strm << "Exception␣when␣calling␣'cdb->ExecuteSQL'." << endl << "Error␣code␣=␣" <<
        e→m_nRetCode << endl << "Exception␣==␣" << e→m_strError << endl <<
        "Exiting␣'list_records'␣unsuccessfully␣with␣return␣value␣-1.";
    AfxMessageBox(temp_strm.str().c_str());
    temp_strm.str("");
    e→Delete();
    temp_ids.Close();
    records_temp.Close();
    contributors_temp.Close();
    creators_temp.Close();
    descriptions_temp.Close();
    identifiers_temp.Close();
    languages_temp.Close();
    publishers_temp.Close();
    rights_temp.Close();
    subjects_temp.Close();
    titles_temp.Close();
    types_temp.Close();
    return −1;
  }    /* catch */
```

**432.**

**Log**

[LDF 2006.08.29.] !! BUG FIX: Now closing and reopening *records_temp* instead of calling **CRecordset** :: *Requery*. I've had problems with the latter function.

---

⟨ Define **Selector** functions 388 ⟩ +≡

```
sql_strm.str("");
records_temp.Close();
records_temp.Open();
html_strm ≪ "<h3>\n<a␣name=\"Result_" ≪ temp_ctr ≪ "\">␣Result␣" ≪ temp_ctr ≪
    ".</a>\n</h3>\n\n" ≪ "<p>\n<b>Record␣" ≪ records_temp.m_record_id ≪
    ".</b>\n<br>\n<br>\n\n";
++temp_ctr;
ret_val = fill_table_streams(cdb);
```

**433.** Error handling: *fill_table_streams* failed. [LDF Undated.]

⟨ Define **Selector** functions 388 ⟩ +≡

```
if (ret_val ≡ −1) {
  temp_strm ≪ "ERROR!␣␣In␣‘Selector::list_records’:" ≪ endl ≪
      "‘Selector::fill_table_streams’␣failed,␣returning␣-1." ≪ endl ≪
      "Exiting␣function␣unsuccessfully␣with␣return␣value␣-1.";
  AfxMessageBox(temp_strm.str().c_str());
  temp_strm.str("");
  temp_ids.Close();
  html_strm.close();
  return −1;
}
#if 0
temp_strm ≪ "records_temp.m_record_id␣==␣" ≪ records_temp.m_record_id ≪ endl;
AfxMessageBox(temp_strm.str().c_str());
temp_strm.str("");
#endif
```

**434.** Write from streams to *html_strm*. [LDF Undated.]

⟨ Define **Selector** functions 388 ⟩ +≡

```
ret_val = write_html_from_streams(QUERY_LISTING, select_value);
```

**435.** Error handling: *write_html_from_stream* failed. [LDF Undated.]

⟨ Define **Selector** functions 388 ⟩ +≡

```
if (ret_val ≠ 0) {
    temp_strm ≪ "ERROR!␣␣In␣‘Selector::list_records’:" ≪ endl ≪
        "‘Selector::write_html_from_streams’␣failed,␣returning␣" ≪ ret_val ≪
        "." ≪ endl ≪ "Exiting␣function␣unsuccessfully␣with␣return␣value␣-1.";
    AfxMessageBox(temp_strm.str().c_str());
    temp_strm.str("");
    temp_ids.Close();
    html_strm.close();    /* Currently, there is no error return from write_html_from_streams. If I add
        one, and I close these record sets, I'll have to remove this code. LDF 2006.06.27. */
    records_temp.Close();
    contributors_temp.Close();
    creators_temp.Close();
    descriptions_temp.Close();
    identifiers_temp.Close();
    languages_temp.Close();
    publishers_temp.Close();
    rights_temp.Close();
    subjects_temp.Close();
    titles_temp.Close();
    types_temp.Close();
    return −1;
}
```

**436.** Move to next record. [LDF Undated.]

⟨ Define **Selector** functions 388 ⟩ +≡

```
temp_ids.MoveNext(); }    /* while (Iterating through temp_ids) */
```

**437.** Write final code to *html_strm* and close it. [LDF Undated.]

⟨ Define **Selector** functions 388 ⟩ +≡

```
html_strm ≪ "\n</font>\n</body>\n</html>\n";
html_strm.close();
```

**438.** Close record sets. [LDF Undated.]

⟨ Define **Selector** functions 388 ⟩ +≡

```
temp_ids.Close();
records_temp.Close();
contributors_temp.Close();
creators_temp.Close();
descriptions_temp.Close();
identifiers_temp.Close();
languages_temp.Close();
publishers_temp.Close();
rights_temp.Close();
subjects_temp.Close();
titles_temp.Close();
types_temp.Close();
```

**439.** Exit function successfully with *record_ctr*. [LDF Undated.]

⟨ Define **Selector** functions 388 ⟩ +≡
  **return** *record_ctr*; }   /* End of **Selector** :: *list_records* definition. */

**440. Fill table streams.** [LDF Undated.]

This function is called by **Selector** :: *select_from_database*, which is defined in `select_from_database.cpp`. [LDF 2006.08.29.]

**Log**

[LDF 2006.08.29.] !! BUG FIX: Now closing and reopening the "*_temp*" objects that reference the various record sets, e.g., *contributors_temp*, *creators_temp*, etc. Formerly, I called **CRecordset** :: *Requery* on them. However, I had problems with the latter function in ZTest. *Requery* was causing the problem that large data sets weren't sorted completely until the search had been performed two or three times. Closing and reopening the objects solved the problem.

---

⟨ Declare **Selector** functions 387 ⟩ +≡
  **int** *fill_table_streams*(**CDatabase** **cdb*);

**441.**

⟨ Define **Selector** functions 388 ⟩ +≡
  **int Selector** :: *fill_table_streams*(**CDatabase** **cdb*){

**442.** Clear streams. [LDF Undated.]

⟨ Define **Selector** functions 388 ⟩ +≡

```
contributor_strm.str("");
creator_strm.str("");
dc_date_strm.str("");
description_strm.str("");
header_datestamp_strm.str("");
identifier_strm.str("");
language_strm.str("");
publisher_strm.str("");
rights_strm.str("");
subject_strm.str("");
title_strm.str("");
type_strm.str("");
temp_strm << "exec␣store_in_temp_tables␣" << records_temp.m_record_id;
try {
  cdb→ExecuteSQL(temp_strm.str().c_str());
}
```

**443.** The error code is in *e→m_nRetCode*. [LDF Undated.]

⟨ Define **Selector** functions 388 ⟩ +≡
```
catch (CDBException *e)
{
  temp_strm << "Exception when calling `cdb->ExecuteSQL'." << endl << "Error code = " <<
      e->m_nRetCode << endl << "Exception == " << e->m_strError << endl <<
      "Exiting `select_from_database' unsuccessfully with return value -1.";
  AfxMessageBox(temp_strm.str().c_str());
  temp_strm.str("");
  e->Delete();
  records_temp.Close();
  contributors_temp.Close();
  creators_temp.Close();
  descriptions_temp.Close();
  identifiers_temp.Close();
  languages_temp.Close();
  publishers_temp.Close();
  rights_temp.Close();
  subjects_temp.Close();
  titles_temp.Close();
  types_temp.Close();
  return -1;
}   /* catch */
temp_strm.str("");
```

**444.** Write *dc_date* to *dc_date_strm*. [LDF Undated.]

⟨ Define **Selector** functions 388 ⟩ +≡
```
dc_date_strm << records_temp.m_dc_date.Format("%Y-%m-%d %H:%M:%S") << "<br>\n";
```

**445.** Write *header_datestamp* to *header_datestamp_strm*. [LDF Undated.]

⟨ Define **Selector** functions 388 ⟩ +≡
```
header_datestamp_strm << records_temp.m_header_datestamp.Format("%Y-%m-%d %H:%M:%S") << "<br>\n";
```

**446.** Iterate through contributors for current record. [LDF Undated.]

⟨ Define **Selector** functions 388 ⟩ +≡
```
#if 0   /* 1 */
  contributors_temp.Requery();
#else
  contributors_temp.Close();
  contributors_temp.Open();
#endif
  while (!contributors_temp.IsEOF()) {
    temp_strm << "contributor id == " << contributors_temp.m_contributor_id << endl <<
        "dc_contributor == " << contributors_temp.m_dc_contributor << endl;
    contributor_strm << contributors_temp.m_dc_contributor << "<br>\n";
    contributors_temp.MoveNext();
  }   /* while */
#if 0
  AfxMessageBox(temp_strm.str().c_str());
#endif
  temp_strm.str("");
```

**447.** Iterate through creators for current record. [LDF Undated.]

```
⟨ Define Selector functions 388 ⟩ +≡
#if 0   /* 1 */
  creators_temp.Requery( );
#else
  creators_temp.Close( );
  creators_temp.Open( );
#endif
  while (¬creators_temp.IsEOF( )) {
    temp_strm ≪ "creator␣id␣==␣" ≪ creators_temp.m_creator_id ≪ endl ≪ "dc_creator␣==␣" ≪
        creators_temp.m_dc_creator ≪ endl;
    creator_strm ≪ creators_temp.m_dc_creator ≪ "<br>\n";
    creators_temp.MoveNext( );
  }   /* while */
```

**448.** Iterate through descriptions for current record. [LDF Undated.]

```
⟨ Define Selector functions 388 ⟩ +≡
#if 0   /* 1 */
  descriptions_temp.Requery( );
#else
  descriptions_temp.Close( );
  descriptions_temp.Open( );
#endif
  while (¬descriptions_temp.IsEOF( )) {
    temp_strm ≪ "description␣id␣==␣" ≪ descriptions_temp.m_description_id ≪ endl ≪
        "dc_description␣==␣" ≪ descriptions_temp.m_dc_description ≪ endl;
    description_strm ≪ descriptions_temp.m_dc_description ≪ "<br>\n";
    descriptions_temp.MoveNext( );
  }   /* while */
#if 0
  AfxMessageBox(temp_strm.str( ).c_str( ));
#endif
  temp_strm.str("");
```

**449.** Iterate through identifiers for current record. [LDF Undated.]

```
⟨ Define Selector functions 388 ⟩ +≡
#if 0    /* 1 */
  identifiers_temp.Requery( );
#else
  identifiers_temp.Close( );
  identifiers_temp.Open( );
#endif
  while (¬identifiers_temp.IsEOF( )) {
    temp_strm ≪ "identifier␣id␣==␣" ≪ identifiers_temp.m_identifier_id ≪ endl ≪
        "dc_identifier␣==␣" ≪ identifiers_temp.m_dc_identifier ≪ endl;
    identifier_strm ≪ "<a␣href=\"" ≪ identifiers_temp.m_dc_identifier ≪ "\">␣" ≪
        identifiers_temp.m_dc_identifier ≪ "␣</a>\n<br>\n";
    identifiers_temp.MoveNext( );
  }    /* while */
#if 0
  AfxMessageBox(temp_strm.str( ).c_str( ));
#endif
  temp_strm.str("");
```

**450.** Iterate through languages for current record. [LDF Undated.]

```
⟨ Define Selector functions 388 ⟩ +≡
#if 0    /* 1 */
  languages_temp.Requery( );
#else
  languages_temp.Close( );
  languages_temp.Open( );
#endif
  while (¬languages_temp.IsEOF( )) {
    temp_strm ≪ "language␣id␣==␣" ≪ languages_temp.m_language_id ≪ endl ≪ "dc_language␣==␣" ≪
        languages_temp.m_dc_language ≪ endl;
    language_strm ≪ languages_temp.m_dc_language ≪ "<br>\n";
    languages_temp.MoveNext( );
  }    /* while */
#if 0
  AfxMessageBox(temp_strm.str( ).c_str( ));
#endif
  temp_strm.str("");
```

**451.** Iterate through publishers for current record. [LDF Undated.]

⟨ Define **Selector** functions 388 ⟩ +≡
```
#if 0    /* 1 */
  publishers_temp.Requery();
#else
  publishers_temp.Close();
  publishers_temp.Open();
#endif
  while (¬publishers_temp.IsEOF()) {
    temp_strm ≪ "publisher␣id␣==␣" ≪ publishers_temp.m_publisher_id ≪ endl ≪
        "dc_publisher␣==␣" ≪ publishers_temp.m_dc_publisher ≪ endl;
    publisher_strm ≪ publishers_temp.m_dc_publisher ≪ "<br>\n";
    publishers_temp.MoveNext();
  }    /* while */
#if 0
  AfxMessageBox(temp_strm.str().c_str());
#endif
  temp_strm.str("");
```

**452.** Iterate through rights for current record. [LDF Undated.]

⟨ Define **Selector** functions 388 ⟩ +≡
```
#if 0    /* 1 */
  rights_temp.Requery();
#else
  rights_temp.Close();
  rights_temp.Open();
#endif
  while (¬rights_temp.IsEOF()) {
    temp_strm ≪ "rights␣id␣==␣" ≪ rights_temp.m_rights_id ≪ endl ≪ "dc_rights␣==␣" ≪
        rights_temp.m_dc_rights ≪ endl;
    rights_strm ≪ "<a␣href=\"" ≪ rights_temp.m_dc_rights ≪ "\">␣" ≪ rights_temp.m_dc_rights ≪
        "␣</a>\n<br>\n";
    rights_temp.MoveNext();
  }    /* while */
#if 0
  AfxMesageBox(temp_strm.str().c_str());
#endif
  temp_strm.str("");
```

**453.** Iterate through subjects for current record. [LDF Undated.]

```
⟨ Define Selector functions 388 ⟩ +≡
#if 0    /* 1 */
  subjects_temp.Requery();
#else
  subjects_temp.Close();
  subjects_temp.Open();
#endif
  while (¬subjects_temp.IsEOF()) {
    temp_strm ≪ "subject␣id␣==␣" ≪ subjects_temp.m_subject_id ≪ endl ≪ "dc_subject␣==␣" ≪
        subjects_temp.m_dc_subject ≪ endl;
    subject_strm ≪ subjects_temp.m_dc_subject ≪ "<br>\n";
    subjects_temp.MoveNext();
  }    /* while */
#if 0
  AfxMessageBox(temp_strm.str().c_str());
#endif
  temp_strm.str("");
```

**454.** Iterate through titles for current record. [LDF Undated.]

```
⟨ Define Selector functions 388 ⟩ +≡
#if 0    /* 1 */
  titles_temp.Requery();
#else
  titles_temp.Close();
  titles_temp.Open();
#endif
  while (¬titles_temp.IsEOF()) {
    temp_strm ≪ "title␣id␣==␣" ≪ titles_temp.m_title_id ≪ endl ≪ "dc_title␣==␣" ≪
        titles_temp.m_dc_title ≪ endl;
    title_strm ≪ titles_temp.m_dc_title ≪ "<br>\n";
    titles_temp.MoveNext();
  }    /* while */
#if 0
  AfxMessageBox(temp_strm.str().c_str());
#endif
  temp_strm.str("");
```

**455.** Iterate through types for current record. [LDF Undated.]

```
⟨ Define Selector functions 388 ⟩ +≡
#if 0    /* 1 */
  types_temp.Requery();
#else
  types_temp.Close();
  types_temp.Open();
#endif
  while (¬types_temp.IsEOF()) {
    temp_strm ≪ "type␣id␣==␣" ≪ types_temp.m_type_id ≪ endl ≪ "dc_type␣==␣" ≪
        types_temp.m_dc_type ≪ endl;
    type_strm ≪ types_temp.m_dc_type ≪ "<br>\n";
    types_temp.MoveNext();
  }    /* while */
#if 0
  AfxMessageBox(temp_strm.str().c_str());
#endif
  temp_strm.str("");
  return 0; }    /* End of Selector::fill_table_streams definition. */
```

**456. Write HTML from streams.** [LDF Undated.]

```
⟨ Declare Selector functions 387 ⟩ +≡
  int write_html_from_streams(
  unsigned short query_type,
  unsigned int select_value = 0,
  CString *search_str = 0);
```

**457.**

```
⟨ Define Selector functions 388 ⟩ +≡
  int Selector::write_html_from_streams(
  unsigned short query_type,
  unsigned int select_value,
  CString *search_str){ string contributor_str;
      string creator_str;
      string dc_date_str;
      string description_str;
      string header_datestamp_str;
      string identifier_str;
      string language_str;
      string publisher_str;
      string rights_str;
      string subject_str;
      string title_str;
      string type_str;
      contributor_str = contributor_strm.str();
      creator_str = creator_strm.str();
      dc_date_str = dc_date_strm.str();
      description_str = description_strm.str();
      header_datestamp_str = header_datestamp_strm.str();
      identifier_str = identifier_strm.str();
      language_str = language_strm.str();
      publisher_str = publisher_strm.str();
      rights_str = rights_strm.str();
      subject_str = subject_strm.str();
      title_str = title_strm.str();
      type_str = type_strm.str();
      string::iterator iter;
      string::size_type pos = 0;
      string search_color_str = "\"red\"";
      string non_search_color_str = "\"purple\"";
      string font_begin_str;
      string font_begin_str_save = "<font␣color=";
      string::size_type font_begin_str_len;
      string::size_type search_str_len = 0;
      if (search_str) search_str_len = search_str->GetLength();
      string font_end_str = "</font>";
      string replacement_str = "";
      if (search_str) {
        replacement_str = font_begin_str;
        replacement_str += *search_str;
        replacement_str += font_end_str;
      }
```

**458.** *erase_value* is the length of the HTML code appended to *contributor_str*, *creator_str*, and the other *strings*. If the code is changed, the value of *erase_value* may have to be changed, too! [LDF 2006.06.27.]

⟨ Define **Selector** functions 388 ⟩ +≡
**string** :: **size_type** *erase_value* = 5;
**string** *temp_find_str*;
**string** *temp_search_str*;
**long** *contributor_ctr* = 0;
**long** *creator_ctr* = 0;
**long** *description_ctr* = 0;
**long** *identifier_ctr* = 0;
**long** *language_ctr* = 0;
**long** *publisher_ctr* = 0;
**long** *rights_ctr* = 0;
**long** *subject_ctr* = 0;
**long** *title_ctr* = 0;
**long** *type_ctr* = 0;

**459.**

⟨ Define **Selector** functions 388 ⟩ +≡
*contributor_ctr* = *contributors_temp*.*GetRecordCount*( );
*creator_ctr* = *creators_temp*.*GetRecordCount*( );
*description_ctr* = *descriptions_temp*.*GetRecordCount*( );
*identifier_ctr* = *identifiers_temp*.*GetRecordCount*( );
*language_ctr* = *languages_temp*.*GetRecordCount*( );
*publisher_ctr* = *publishers_temp*.*GetRecordCount*( );
*rights_ctr* = *rights_temp*.*GetRecordCount*( );
*subject_ctr* = *subjects_temp*.*GetRecordCount*( );
*title_ctr* = *titles_temp*.*GetRecordCount*( );
*type_ctr* = *types_temp*.*GetRecordCount*( );

**460.** Find *search_str* and color it, if we performed a search for it. [LDF Undated.]

⟨ Define **Selector** functions 388 ⟩ +≡
**if** (*query_type* ≡ QUERY_SEARCH ∧ *select_value* ∧ *search_str*) { *temp_search_str* = ∗*search_str*;
*strlwr*(**const_cast**⟨**char** ∗⟩(*temp_search_str*.*c_str*( )));

**461.** Contributors. [LDF Undated.]

```
⟨ Define Selector functions 388 ⟩ +≡
  if (contributor_ctr ≤ 0) ;   /* Do nothing. */
  else   /* contributor_ctr > 0 */
  {
    if (select_value & CONTRIBUTORS) font_begin_str = font_begin_str_save + search_color_str + ">";
    else font_begin_str = font_begin_str_save + non_search_color_str + ">";
    font_begin_str_len = font_begin_str.length();
    pos = 0;
    temp_find_str = contributor_str.erase(contributor_str.length() − erase_value);
    strlwr(const_cast⟨char *⟩(temp_find_str.c_str()));
    while (TRUE) {
      pos = temp_find_str.find(temp_search_str, pos);
      if (pos ≡ string::npos) break;
      else {
        contributor_str.insert(pos, font_begin_str);
        temp_find_str.insert(pos, font_begin_str);
        pos += search_str_len + font_begin_str_len;
        contributor_str.insert(pos, font_end_str);
        temp_find_str.insert(pos, font_end_str);
        pos += font_end_str.length();
        continue;
      }   /* else */
    }   /* while */
  }   /* else (contributor_ctr > 0) */
  temp_find_str = "";
```

**462.** Creators. [LDF Undated.]

⟨ Define **Selector** functions 388 ⟩ +≡

```
if (creator_ctr ≤ 0) ;   /* Do nothing. */
else   /* creator_ctr > 0 */
{
  if (select_value & CREATORS) font_begin_str = font_begin_str_save + search_color_str + ">";
  else font_begin_str = font_begin_str_save + non_search_color_str + ">";
  font_begin_str_len = font_begin_str.length();
  pos = 0;
  temp_find_str = creator_str.erase(creator_str.length() − erase_value);
  strlwr(const_cast⟨char *⟩(temp_find_str.c_str()));
  while (TRUE) {
    pos = temp_find_str.find(temp_search_str, pos);
    if (pos ≡ string::npos) break;
    else {
      creator_str.insert(pos, font_begin_str);
      temp_find_str.insert(pos, font_begin_str);
      pos += search_str_len + font_begin_str_len;
      creator_str.insert(pos, font_end_str);
      temp_find_str.insert(pos, font_end_str);
      pos += font_end_str.length();
      continue;
    }   /* else */
  }   /* while */
}   /* else (creator_ctr > 0) */
temp_find_str = "";
```

**463.** Descriptions. [LDF Undated.]

⟨ Define **Selector** functions 388 ⟩ +≡

```
description_ctr = descriptions_temp.GetRecordCount();
if (description_ctr ≤ 0) ;    /* Do nothing. */
else    /* description_ctr > 0 */
{
  if (select_value & DESCRIPTIONS) font_begin_str = font_begin_str_save + search_color_str + ">";
  else font_begin_str = font_begin_str_save + non_search_color_str + ">";
  font_begin_str_len = font_begin_str.length();
  pos = 0;
  temp_find_str = description_str.erase(description_str.length() − erase_value);
  strlwr(const_cast⟨char *⟩(temp_find_str.c_str()));
  while (TRUE) {
    pos = temp_find_str.find(temp_search_str, pos);
    if (pos ≡ string::npos) break;
    else {
      description_str.insert(pos, font_begin_str);
      temp_find_str.insert(pos, font_begin_str);
      pos += search_str_len + font_begin_str_len;
      description_str.insert(pos, font_end_str);
      temp_find_str.insert(pos, font_end_str);
      pos += font_end_str.length();
      continue;
    }    /* else */
  }    /* while */
}    /* else (description_ctr > 0) */
temp_find_str = "";
```

**464.** Identifiers. [LDF Undated.]

Don't color the search string in *identifier_str*. [LDF 2006.06.26.]

```
⟨ Define Selector functions 388 ⟩ +≡
#if 0     /* 1 */
  if (identifier_ctr ≤ 0) ;     /* Do nothing. */
  else     /* identifier_ctr > 0 */
  {
    if (select_value & IDENTIFIERS) font_begin_str = font_begin_str_save + search_color_str + ">";
    else font_begin_str = font_begin_str_save + non_search_color_str + ">";
    font_begin_str_len = font_begin_str.length( );
    pos = 0;
    temp_find_str = identifier_str.erase(identifier_str.length( ) − erase_value);
    strlwr(const_cast⟨char *⟩(temp_find_str.c_str( )));
    while (TRUE) {
      pos = temp_find_str.find(temp_search_str, pos);
      if (pos ≡ string::npos) break;
      else {
        identifier_str.insert(pos, font_begin_str);
        temp_find_str.insert(pos, font_begin_str);
        pos += search_str_len + font_begin_str_len;
        identifier_str.insert(pos, font_end_str);
        temp_find_str.insert(pos, font_end_str);
        pos += font_end_str.length( );
        continue;
      }     /* else */
    }     /* while */
  }     /* else (identifier_ctr > 0) */
  temp_find_str = "";
#endif
```

**465.** Languages. [LDF Undated.]

```
⟨ Define Selector functions 388 ⟩ +≡
  language_ctr = languages_temp.GetRecordCount();
  if (language_ctr ≤ 0) ;    /* Do nothing. */
  else    /* language_ctr > 0 */
  {
    if (select_value & LANGUAGES) font_begin_str = font_begin_str_save + search_color_str + ">";
    else font_begin_str = font_begin_str_save + non_search_color_str + ">";
    font_begin_str_len = font_begin_str.length();
    pos = 0;
    temp_find_str = language_str.erase(language_str.length() − erase_value);
    strlwr(const_cast⟨char *⟩(temp_find_str.c_str()));
    while (TRUE) {
      pos = temp_find_str.find(temp_search_str, pos);
      if (pos ≡ string::npos) break;
      else {
        language_str.insert(pos, font_begin_str);
        temp_find_str.insert(pos, font_begin_str);
        pos += search_str_len + font_begin_str_len;
        language_str.insert(pos, font_end_str);
        temp_find_str.insert(pos, font_end_str);
        pos += font_end_str.length();
        continue;
      }    /* else */
    }    /* while */
  }    /* else (language_ctr > 0) */
  temp_find_str = "";
```

**466.** Publishers. [LDF Undated.]

⟨ Define **Selector** functions 388 ⟩ +≡

```
publisher_ctr = publishers_temp.GetRecordCount();
if (publisher_ctr ≤ 0) ;     /* Do nothing. */
else     /* publisher_ctr > 0 */
{
  if (select_value & PUBLISHERS) font_begin_str = font_begin_str_save + search_color_str + ">";
  else font_begin_str = font_begin_str_save + non_search_color_str + ">";
  font_begin_str_len = font_begin_str.length();
  pos = 0;
  temp_find_str = publisher_str.erase(publisher_str.length() − erase_value);
  strlwr(const_cast⟨char *⟩(temp_find_str.c_str()));
  while (TRUE) {
    pos = temp_find_str.find(temp_search_str, pos);
    if (pos ≡ string::npos) break;
    else {
      publisher_str.insert(pos, font_begin_str);
      temp_find_str.insert(pos, font_begin_str);
      pos += search_str_len + font_begin_str_len;
      publisher_str.insert(pos, font_end_str);
      temp_find_str.insert(pos, font_end_str);
      pos += font_end_str.length();
      continue;
    }     /* else */
  }     /* while */
}     /* else (publisher_ctr > 0) */
temp_find_str = "";
```

**467.** Rights. [LDF Undated.]

```
⟨ Define Selector functions 388 ⟩ +≡
  rights_ctr = rights_temp.GetRecordCount();
  if (rights_ctr ≤ 0) ;    /* Do nothing. */
  else    /* rights_ctr > 0 */
  {
    if (select_value & RIGHTS) font_begin_str = font_begin_str_save + search_color_str + ">";
    else font_begin_str = font_begin_str_save + non_search_color_str + ">";
    font_begin_str_len = font_begin_str.length();
    pos = 0;
    temp_find_str = rights_str.erase(rights_str.length() − erase_value);
    strlwr(const_cast⟨char *⟩(temp_find_str.c_str()));
    while (TRUE) {
      pos = temp_find_str.find(temp_search_str, pos);
      if (pos ≡ string::npos) break;
      else {
        rights_str.insert(pos, font_begin_str);
        temp_find_str.insert(pos, font_begin_str);
        pos += search_str_len + font_begin_str_len;
        rights_str.insert(pos, font_end_str);
        temp_find_str.insert(pos, font_end_str);
        pos += font_end_str.length();
        continue;
      }    /* else */
    }    /* while */
  }    /* else (rights_ctr > 0) */
  temp_find_str = "";
```

**468.** Subjects. [LDF Undated.]

```
⟨ Define Selector functions 388 ⟩ +≡
  subject_ctr = subjects_temp.GetRecordCount();
  if (subject_ctr ≤ 0) ;    /* Do nothing. */
  else    /* subject_ctr > 0 */
  {
    if (select_value & SUBJECTS) font_begin_str = font_begin_str_save + search_color_str + ">";
    else font_begin_str = font_begin_str_save + non_search_color_str + ">";
    font_begin_str_len = font_begin_str.length();
    pos = 0;
    temp_find_str = subject_str.erase(subject_str.length() − erase_value);
    strlwr(const_cast⟨char *⟩(temp_find_str.c_str()));
    while (TRUE) {
      pos = temp_find_str.find(temp_search_str, pos);
      if (pos ≡ string::npos) break;
      else {
        subject_str.insert(pos, font_begin_str);
        temp_find_str.insert(pos, font_begin_str);
        pos += search_str_len + font_begin_str_len;
        subject_str.insert(pos, font_end_str);
        temp_find_str.insert(pos, font_end_str);
        pos += font_end_str.length();
        continue;
      }    /* else */
    }    /* while */
  }    /* else (subject_ctr > 0) */
  temp_find_str = "";
```

**469.** Titles. [LDF Undated.]

⟨ Define **Selector** functions 388 ⟩ +≡

```
title_ctr = titles_temp.GetRecordCount();
if (title_ctr <= 0) ;  /* Do nothing. */
else  /* title_ctr > 0 */
{
  if (select_value & TITLES) font_begin_str = font_begin_str_save + search_color_str + ">";
  else font_begin_str = font_begin_str_save + non_search_color_str + ">";
  font_begin_str_len = font_begin_str.length();
  pos = 0;
  temp_find_str = title_str.erase(title_str.length() - erase_value);
  strlwr(const_cast<char *>(temp_find_str.c_str()));
  while (TRUE) {
    pos = temp_find_str.find(temp_search_str, pos);
    if (pos == string::npos) break;
    else {
      title_str.insert(pos, font_begin_str);
      temp_find_str.insert(pos, font_begin_str);
      pos += search_str_len + font_begin_str_len;
      title_str.insert(pos, font_end_str);
      temp_find_str.insert(pos, font_end_str);
      pos += font_end_str.length();
      continue;
    }  /* else */
  }  /* while */
}  /* else (title_ctr > 0) */
temp_find_str = "";
```

**470.** Types. [LDF Undated.]

⟨ Define **Selector** functions 388 ⟩ +≡

```
type_ctr = types_temp.GetRecordCount( );
if (type_ctr ≤ 0) ;    /* Do nothing. */
else    /* type_ctr > 0 */
{
  if (select_value & TYPES) font_begin_str = font_begin_str_save + search_color_str + ">";
  else font_begin_str = font_begin_str_save + non_search_color_str + ">";
  font_begin_str_len = font_begin_str.length( );
  pos = 0;
  temp_find_str = type_str.erase(type_str.length( ) − erase_value);
  strlwr(const_cast⟨char *⟩(temp_find_str.c_str( )));
  while (TRUE) {
    pos = temp_find_str.find(temp_search_str, pos);
    if (pos ≡ string::npos) break;
    else {
      type_str.insert(pos, font_begin_str);
      temp_find_str.insert(pos, font_begin_str);
      pos += search_str_len + font_begin_str_len;
      type_str.insert(pos, font_end_str);
      temp_find_str.insert(pos, font_end_str);
      pos += font_end_str.length( );
      continue;
    }    /* else */
  }    /* while */
}    /* else (type_ctr > 0) */
temp_find_str = ""; }    /* if (query_type ≡ QUERY_SEARCH ∧ select_value ∧ search_str) */
else
  if (query_type ≡ QUERY_LISTING ∧ select_value) {
    font_begin_str = font_begin_str_save + search_color_str + ">";
  }
```

**471.** Output strings to *html_strm*. [LDF Undated.]

⟨ Define **Selector** functions 388 ⟩ +≡

**472.** Title(s). [LDF Undated.]

⟨ Define **Selector** functions 388 ⟩ +≡

```
if (title_ctr ≤ 0 ∨ title_strm.tellp( ) ≤ 0) html_strm ≪ "\n<b>No␣Titles.</b>\n<br>\n<br>\n";
else {
  if (title_ctr ≡ 1) html_strm ≪ "\n\n<b>Title:</b>␣";
  else html_strm ≪ "\n\n<b>Titles:</b>\n<br>\n";
  if (query_type ≡ QUERY_LISTING ∧ select_value ≡ TITLES) {
    html_strm ≪ font_begin_str;
  }
  html_strm ≪ title_str ≪ "\n<br>\n\n";
  if (query_type ≡ QUERY_LISTING ∧ select_value ≡ TITLES) {
    html_strm ≪ font_end_str;
  }
  html_strm ≪ "\n<br>\n";
}
```

**473.** Creator(s). [LDF Undated.]

```
⟨ Define Selector functions 388 ⟩ +≡
  if (creator_ctr ≤ 0 ∨ creator_strm.tellp( ) ≤ 0)
    html_strm ≪ "\n<b>No␣Creators.</b>\n<br>\n<br>\n";
  else {
    if (creator_ctr ≡ 1) html_strm ≪ "\n\n<b>Creator:</b>␣";
    else html_strm ≪ "\n\n<b>Creators:</b>\n<br>\n";
    if (query_type ≡ QUERY_LISTING ∧ select_value ≡ CREATORS) {
      html_strm ≪ font_begin_str;
    }
    html_strm ≪ creator_str ≪ "\n<br>\n\n";
    if (query_type ≡ QUERY_LISTING ∧ select_value ≡ CREATORS) {
      html_strm ≪ font_end_str;
    }
    html_strm ≪ "\n<br>\n";
  }
```

**474.** Description(s). [LDF Undated.]

```
⟨ Define Selector functions 388 ⟩ +≡
  if (description_ctr ≤ 0 ∨ description_strm.tellp( ) ≤ 0)
    html_strm ≪ "\n\n<b>No␣Descriptions.</b>\n<br>\n<br>\n";
  else {
    if (description_ctr ≡ 1) html_strm ≪ "\n\n<b>Description:</b>␣";
    else html_strm ≪ "\n\n<b>Descriptions:</b>\n<br>\n";
    if (query_type ≡ QUERY_LISTING ∧ select_value ≡ DESCRIPTIONS) {
      html_strm ≪ font_begin_str;
    }
    html_strm ≪ description_str ≪ "\n<br>\n\n";
    if (query_type ≡ QUERY_LISTING ∧ select_value ≡ DESCRIPTIONS) {
      html_strm ≪ font_end_str;
    }
    html_strm ≪ "\n<br>\n";
  }
```

**475.** Contributor(s). [LDF Undated.]

```
⟨ Define Selector functions 388 ⟩ +≡
  if (contributor_ctr ≤ 0 ∨ contributor_strm.tellp( ) ≤ 0)
    html_strm ≪ "\n\n<b>No␣Contributors.</b>\n<br>\n<br>\n";
  else {
    if (contributor_ctr ≡ 1) html_strm ≪ "\n\n<b>Contributor:</b>␣";
    else html_strm ≪ "\n\n<b>Contributors:</b>\n<br>\n";
    if (query_type ≡ QUERY_LISTING ∧ select_value ≡ CONTRIBUTORS) {
      html_strm ≪ font_begin_str;
    }
    html_strm ≪ contributor_str ≪ "\n<br>\n\n";
    if (query_type ≡ QUERY_LISTING ∧ select_value ≡ CONTRIBUTORS) {
      html_strm ≪ font_end_str;
    }
    html_strm ≪ "\n<br>\n";
  }
```

**476.** Subject(s). [LDF Undated.]

⟨ Define **Selector** functions 388 ⟩ +≡

```
if (subject_ctr ≤ 0 ∨ subject_strm.tellp( ) ≤ 0)
    html_strm ≪ "\n\n<b>No␣Subjects.</b>\n<br>\n<br>\n";
else {
    if (subject_ctr ≡ 1) html_strm ≪ "\n\n<b>Subject:</b>␣";
    else html_strm ≪ "\n\n<b>Subjects:</b>\n<br>\n";
    if (query_type ≡ QUERY_LISTING ∧ select_value ≡ SUBJECTS) {
        html_strm ≪ font_begin_str;
    }
    html_strm ≪ subject_str ≪ "\n<br>\n\n";
    if (query_type ≡ QUERY_LISTING ∧ select_value ≡ SUBJECTS) {
        html_strm ≪ font_end_str;
    }
    html_strm ≪ "\n<br>\n";
}
```

**477.** Identifier(s). [LDF Undated.]

⟨ Define **Selector** functions 388 ⟩ +≡

```
if (identifier_ctr ≤ 0 ∨ identifier_strm.tellp( ) ≤ 0)
    html_strm ≪ "\n\n<b>No␣Identifiers.</b>\n<br>\n<br>\n";
else {
    if (identifier_ctr ≡ 1) html_strm ≪ "\n\n<b>Identifier:</b>␣";
    else html_strm ≪ "\n\n<b>Identifiers:</b>\n<br>\n";
    if (query_type ≡ QUERY_LISTING ∧ select_value ≡ IDENTIFIERS) {
        html_strm ≪ font_begin_str;
    }
    html_strm ≪ identifier_str ≪ "\n<br>\n\n";
    if (query_type ≡ QUERY_LISTING ∧ select_value ≡ IDENTIFIERS) {
        html_strm ≪ font_end_str;
    }
    html_strm ≪ "\n<br>\n";
}
```

**478.** Publisher(s). [LDF Undated.]

⟨ Define **Selector** functions 388 ⟩ +≡

```
if (publisher_ctr ≤ 0 ∨ publisher_strm.tellp( ) ≤ 0)
    html_strm ≪ "\n\n<b>No␣Publishers.</b>\n<br>\n<br>\n";
else {
    if (publisher_ctr ≡ 1) html_strm ≪ "\n\n<b>Publisher:</b>␣";
    else html_strm ≪ "\n\n<b>Publishers:</b>\n<br>\n";
    if (query_type ≡ QUERY_LISTING ∧ select_value ≡ PUBLISHERS) {
        html_strm ≪ font_begin_str;
    }
    html_strm ≪ publisher_str ≪ "\n<br>\n\n";
    if (query_type ≡ QUERY_LISTING ∧ select_value ≡ PUBLISHERS) {
        html_strm ≪ font_end_str;
    }
    html_strm ≪ "\n<br>\n";
}
```

**479.** *dc_date*. [LDF Undated.]

```
⟨ Define Selector functions 388 ⟩ +≡
  if (query_type ≡ QUERY_LISTING ∧ select_value ≡ DC_DATES) {
    html_strm ≪ font_begin_str;
  }
  html_strm ≪ "\n\n<b>Date␣(dc:date):</b>␣" ≪ dc_date_str ≪ "\n\n";
  if (query_type ≡ QUERY_LISTING ∧ select_value ≡ DC_DATES) {
    html_strm ≪ font_end_str;
  }
  html_strm ≪ "\n<br>\n";
```

**480.** *header_datestamp*. [LDF Undated.]

```
⟨ Define Selector functions 388 ⟩ +≡
  if (query_type ≡ QUERY_LISTING ∧ select_value ≡ HEADER_DATESTAMPS) {
    html_strm ≪ font_begin_str;
  }
  html_strm ≪ "\n\n<b>Date␣(header␣datestamp):</b>␣" ≪ header_datestamp_str ≪ "\n\n";
  if (query_type ≡ QUERY_LISTING ∧ select_value ≡ HEADER_DATESTAMPS) {
    html_strm ≪ font_end_str;
  }
  html_strm ≪ "\n<br>\n";
```

**481.** Language(s). [LDF Undated.]

```
⟨ Define Selector functions 388 ⟩ +≡
  if (language_ctr ≤ 0 ∨ language_strm.tellp( ) ≤ 0)
    html_strm ≪ "\n\n<b>No␣Languages.</b>\n<br>\n<br>\n";
  else {
    if (language_ctr ≡ 1) html_strm ≪ "\n\n<b>Language:</b>␣";
    else html_strm ≪ "\n\n<b>Languages:</b>\n<br>\n";
    html_strm ≪ language_str ≪ "\n<br>\n\n";
    html_strm ≪ "\n<br>\n";
  }
```

**482.** Type(s). [LDF Undated.]

```
⟨ Define Selector functions 388 ⟩ +≡
  if (type_ctr ≤ 0 ∨ type_strm.tellp( ) ≤ 0) html_strm ≪ "\n\n<b>No␣Types.</b>\n<br>\n<br>\n";
  else {
    if (type_ctr ≡ 1) html_strm ≪ "\n\n<b>Type:</b>␣";
    else html_strm ≪ "\n\n<b>Types:</b>\n<br>\n";
    if (query_type ≡ QUERY_LISTING ∧ select_value ≡ TYPES) {
      html_strm ≪ font_begin_str;
    }
    html_strm ≪ type_str ≪ "\n<br>\n\n";
    if (query_type ≡ QUERY_LISTING ∧ select_value ≡ TYPES) {
      html_strm ≪ font_end_str;
    }
    html_strm ≪ "\n<br>\n";
  }
```

**483.** Rights. [LDF Undated.]

⟨ Define **Selector** functions 388 ⟩ +≡

```
if (rights_ctr ≤ 0 ∨ rights_strm.tellp() ≤ 0) html_strm ≪ "\n\n<b>No␣Rights.</b>\n<br>\n<br>\n";
else {
  if (rights_ctr ≡ 1) html_strm ≪ "\n\n<b>Rights:</b>␣";
  else html_strm ≪ "\n\n<b>Rights:</b>\n<br>\n";
  if (query_type ≡ QUERY_LISTING ∧ select_value ≡ RIGHTS) {
    html_strm ≪ font_begin_str;
  }
  html_strm ≪ rights_str ≪ "\n<br>\n\n";
  if (query_type ≡ QUERY_LISTING ∧ select_value ≡ RIGHTS) {
    html_strm ≪ font_end_str;
  }
  html_strm ≪ "\n<br>\n";
}
html_strm ≪ "\n</p>\n\n<hr␣size=\"2\"␣color=\"black\">\n\n";
```

**484.** Clear streams and strings. [LDF Undated.]

⟨ Define **Selector** functions 388 ⟩ +≡

```
contributor_strm.str("");
dc_date_strm.str("");
creator_strm.str("");
description_strm.str("");
header_datestamp_strm.str("");
identifier_strm.str("");
language_strm.str("");
publisher_strm.str("");
rights_strm.str("");
subject_strm.str("");
title_strm.str("");
type_strm.str("");
contributor_str = "";
creator_str = "";
dc_date_str = "";
description_str = "";
header_datestamp_str = "";
identifier_str = "";
language_str = "";
publisher_str = "";
rights_str = "";
subject_str = "";
title_str = "";
type_str = "";
return 0; }   /* End of Selector::write_html_from_streams definition. */
```

**485. List table.** [LDF Undated.]

⟨ Declare **Selector** functions 387 ⟩ +≡

```
int list_table(
unsigned short table,
unsigned short timespan,
unsigned short sort_order);
```

**486.**

⟨ Define **Selector** functions 388 ⟩ +≡
```
int Selector::list_table(
unsigned short table,
unsigned short timespan,
unsigned short sort_order){
    int ret_val = 0;
```

**487.** Timespan. [LDF Undated.]

⟨ Define **Selector** functions 388 ⟩ +≡
```
int day_of_month;
int day_of_week;
int month_of_year;
CTime t0;
stringstream timespan_strm;
timespan_strm.str("");
string date_type_str;
```

**488.**

⟨ Define **Selector** functions 388 ⟩ +≡
```
if (use_date_type ≡ USE_DC_DATE) date_type_str = "R.dc_date";
else date_type_str = "R.header_datestamp";
string sort_order_str;
sort_order_str = (sort_order ≡ SORT_DESCENDING) ? "desc" : "asc";
```

**489.**

⟨ Define **Selector** functions 388 ⟩ +≡

```
if (timespan ≤ NULL_TIMESPAN ∨ timespan > ALL_RECORDS) {
  temp_strm ≪ "WARNING!␣␣In␣`Selector::list_table':" ≪ endl ≪
      "`timespan'␣has␣invalid␣value:␣" ≪ timespan ≪ endl ≪
      "Setting␣`timespan'␣to␣`ALL_RECORDS'␣and␣continuing.";
  AfxMessageBox(temp_strm.str().c_str());
  temp_strm.str("");
  timespan = ALL_RECORDS;
}
if (timespan ≡ ALL_RECORDS) ;   /* Do nothing. */
else if (timespan ≡ TODAY) {
  t0 = CTime::GetCurrentTime();
  timespan_strm ≪ date_type_str ≪ "␣>=␣convert(datetime,␣'" ≪
      t0.Format("%Y-%m-%d␣00:00:00") ≪ "',␣121)␣";
}
else if (timespan ≡ YESTERDAY) {
  t0 = CTime::GetCurrentTime();
  t0 −= CTimeSpan(1, 0, 0, 0);
  timespan_strm ≪ date_type_str ≪ "␣>=␣convert(datetime,␣'" ≪
      t0.Format("%Y-%m-%d␣00:00:00") ≪ "',␣121)␣";
}
else if (timespan ≡ THIS_WEEK) {
  t0 = CTime::GetCurrentTime();
  day_of_week = t0.GetDayOfWeek() − 1;
  t0 −= CTimeSpan(day_of_week, 0, 0, 0);
  timespan_strm ≪ date_type_str ≪ "␣>=␣convert(datetime,␣'" ≪
      t0.Format("%Y-%m-%d␣00:00:00") ≪ "',␣121)␣";
}
else if (timespan ≡ LAST_WEEK) {
  t0 = CTime::GetCurrentTime();
  day_of_week = t0.GetDayOfWeek() − 1;
  t0 −= CTimeSpan(7 + day_of_week, 0, 0, 0);
  timespan_strm ≪ date_type_str ≪ "␣>=␣convert(datetime,␣'" ≪
      t0.Format("%Y-%m-%d␣00:00:00") ≪ "',␣121)␣";
}
```

**490.**

```
⟨ Define Selector functions 388 ⟩ +≡
  else
  if (timespan ≡ THIS_MONTH) {
    t0 = CTime::GetCurrentTime();
    day_of_month = t0.GetDay();
    t0 −= CTimeSpan(day_of_month − 1, 0, 0, 0);
    timespan_strm ≪ date_type_str ≪ "␣>=␣convert(datetime,␣'" ≪
        t0.Format("%Y-%m-%d␣00:00:00") ≪ "',␣121)␣";
  }
  else if (timespan ≡ LAST_MONTH) {
    t0 = CTime::GetCurrentTime();
    day_of_month = t0.GetDay();
    t0 −= CTimeSpan(day_of_month, 0, 0, 0);
    day_of_month = t0.GetDay();
    t0 −= CTimeSpan(day_of_month − 1, 0, 0, 0);
    timespan_strm ≪ date_type_str ≪ "␣>=␣convert(datetime,␣'" ≪
        t0.Format("%Y-%m-%d␣00:00:00") ≪ "',␣121)␣";
  }
  else if (timespan ≡ LAST_6_MONTHS) { t0 = CTime::GetCurrentTime();
  month_of_year = t0.GetMonth();
  if (month_of_year > 5) {
    CTime t1(t0.GetYear(), month_of_year − 5, 1, 0, 0, 0);
    t0 = t1;
  }
  else {
    CTime t1(t0.GetYear() − 1, month_of_year + 7, 1, 0, 0, 0);
    t0 = t1;
  }
```

**491.**

```
⟨ Define Selector functions 388 ⟩ +≡
  timespan_strm ≪ date_type_str ≪ "␣>=␣convert(datetime,␣'" ≪ t0.Format("%Y-%m-%d␣00:00:00") ≪
      "',␣121)␣"; }
  else
    if (timespan ≡ THIS_YEAR) {
      t0 = CTime::GetCurrentTime();
      CTime t1(t0.GetYear(), 1, 1, 0, 0, 0);
      t0 = t1;
      timespan_strm ≪ date_type_str ≪ "␣>=␣convert(datetime,␣'" ≪
          t0.Format("%Y-%m-%d␣00:00:00") ≪ "',␣121)␣";
    }
    else if (timespan ≡ LAST_YEAR) {
      t0 = CTime::GetCurrentTime();
      CTime t1(t0.GetYear() − 1, 1, 1, 0, 0, 0);
      t0 = t1;
      timespan_strm ≪ date_type_str ≪ "␣>=␣convert(datetime,␣'" ≪
          t0.Format("%Y-%m-%d␣00:00:00") ≪ "',␣121)␣";
    }
    else if (timespan ≡ LAST_2_YEARS) {
      t0 = CTime::GetCurrentTime();
      CTime t1(t0.GetYear() − 2, 1, 1, 0, 0, 0);
      t0 = t1;
      timespan_strm ≪ date_type_str ≪ "␣>=␣convert(datetime,␣'" ≪
          t0.Format("%Y-%m-%d␣00:00:00") ≪ "',␣121)␣";
    }
    else if (timespan ≡ LAST_5_YEARS) {
      t0 = CTime::GetCurrentTime();
      CTime t1(t0.GetYear() − 5, 1, 1, 0, 0, 0);
      t0 = t1;
      timespan_strm ≪ date_type_str ≪ "␣>=␣convert(datetime,␣'" ≪
          t0.Format("%Y-%m-%d␣00:00:00") ≪ "',␣121)␣";
    }
    else if (timespan ≡ LAST_10_YEARS) {
      t0 = CTime::GetCurrentTime();
      CTime t1(t0.GetYear() − 10, 1, 1, 0, 0, 0);
      t0 = t1;
      timespan_strm ≪ date_type_str ≪ "␣>=␣convert(datetime,␣'" ≪
          t0.Format("%Y-%m-%d␣00:00:00") ≪ "',␣121)␣";
    }
    else if (timespan ≡ LAST_20_YEARS) {
      t0 = CTime::GetCurrentTime();
      CTime t1(t0.GetYear() − 20, 1, 1, 0, 0, 0);
      t0 = t1;
      timespan_strm ≪ date_type_str ≪ "␣>=␣convert(datetime,␣'" ≪
          t0.Format("%Y-%m-%d␣00:00:00") ≪ "',␣121)␣";
    }
  if (table ≡ CONTRIBUTORS) { html_strm.open("contributors.html");
```

**492.** Write header to *html_strm*. [LDF Undated.]

⟨ Define **Selector** functions 388 ⟩ +≡

```
html_strm ≪ "<!--␣contributors.html␣-->" ≪ endl ≪ endl ≪
    "<!DOCTYPE␣HTML␣PUBLIC␣" ≪ "\"-//W3C//DTD␣HTML␣4.01␣Transitional//EN\"" ≪
    endl ≪ "\"http://www.w3.org/TR/html4/loose.dtd\">" ≪ endl ≪
    "<!␣file:///u|/contributors.html>" ≪ "<html>\n<head>\n<title>\nContributors␣L\
    ist\n</title>\n" ≪ "</head>\n<body>\n<font␣color=\"black\">\n" ≪
    "\n<h1␣align=\"center\">\n<a␣name=\"Top\">␣Contributors␣List" ≪
    "\n</a>\n</h1>\n</font>\n\n" ≪ copyright_html_str ≪
    "<hr␣size=\"2\"␣color=\"black\">\n\n" ≪ "\n<br>\n<br>\n" ≪ "<b>Sort␣Order:</b>␣" ≪
    "\n<br>\n<br>\n";
temp_strm ≪ "delete␣contributors_temp␣insert␣" ≪
    "contributors_temp␣select␣distinct␣C.contributor_id,␣" ≪
    "C.dc_contributor,␣C.institution_id,␣C.person_id␣" ≪
    "from␣Contributors␣as␣C,␣Records␣as␣R,␣Records_Contributors␣as␣RC␣" ≪
    "where␣C.contributor_id␣>␣0␣" ≪ "and␣RC.contributor_id␣=␣C.contributor_id␣" ≪
    "and␣RC.record_id␣=␣R.record_id␣";
if (timespan ≠ ALL_RECORDS) temp_strm ≪ "and␣" ≪ timespan_strm.str();
temp_strm ≪ "order␣by␣C.dc_contributor␣" ≪ sort_order_str;
cdb→ExecuteSQL(temp_strm.str().c_str());
temp_strm.str("");
contributors_temp.Open();
unsigned long contributor_ctr;
while (¬contributors_temp.IsEOF()) {
  contributors_temp.MoveNext();
}
contributor_ctr = contributors_temp.GetRecordCount();
if (contributor_ctr ≡ 0) {
  html_strm ≪ "<b>No␣Contributors␣found.</b>\n<br>\n<br>\n" ≪
      "<hr␣size=\"2\"␣color=\"black\">\n\n<br>\n";
  return 0;
}
else ret_val = contributor_ctr;
html_strm ≪ "<b>Contributors␣found:</b>␣" ≪ contributor_ctr ≪ "\n<br>\n<br>\n" ≪
    "<hr␣size=\"2\"␣color=\"black\">\n\n<br>\n";
contributors_temp.MoveFirst();
contributor_ctr = 1;
while (¬contributors_temp.IsEOF()) {
  html_strm ≪ contributor_ctr++ ≪ ".␣" ≪ contributors_temp.m_dc_contributor ≪ "␣";
  temp_strm ≪ "delete␣temp_ids␣" ≪ "insert␣temp_ids␣select␣R.record_id␣" ≪
      "from␣records␣as␣R,␣records_contributors␣as␣RC,␣" ≪ "Contributors␣as␣␣C␣" ≪
      "where␣C.contributor_id␣=␣" ≪ contributors_temp.m_contributor_id ≪ "␣" ≪
      "and␣RC.contributor_id␣=␣" ≪ contributors_temp.m_contributor_id ≪ "␣" ≪
      "and␣RC.record_id␣=␣R.record_id";
  cdb→ExecuteSQL(temp_strm.str().c_str());
  temp_strm.str("");
  temp_ids.Open();
  while (¬temp_ids.IsEOF()) {
    temp_ids.MoveNext();
  }   /* while (¬temp_ids.IsEOF()) */
  if (temp_ids.GetRecordCount() > 0) {
```

```
      temp_ids.MoveFirst();
      while (¬temp_ids.IsEOF()) {
        html_strm ≪ "[" ≪ "<a␣href=\"./listing_complete.html#Result_" ≪ temp_ids.m_temp_id ≪
            "\">" ≪ temp_ids.m_temp_id ≪ "</a>]␣";
        temp_ids.MoveNext();
      }    /* while (¬temp_ids.IsEOF()) */
    }    /* if (temp_ids.GetRecordCount() > 0) */
    temp_ids.Close();
    html_strm ≪ "\n<br>\n<br>\n\n";
    contributors_temp.MoveNext();
  }    /* while (¬contributors_temp.IsEOF()) */
  contributors_temp.Close();
```

**493.** Write trailer to *html_strm* and close it. [LDF Undated.]

```
⟨Define Selector functions 388⟩ +≡
  html_strm ≪ "\n\n<br>\n<br>\n" ≪ "<hr␣size=\"2\"␣color=\"black\">\n\n" ≪
      "</font>\n</body>\n</html>\n";
  html_strm.close(); }    /* if (table ≡ CONTRIBUTORS) */
  else if (table ≡ CREATORS) { html_strm.open("creators.html");
```

**494.** Write header to *html_strm*. [LDF Undated.]

⟨Define **Selector** functions 388⟩ +≡

```
html_strm ≪ "<!--␣creators.html␣-->" ≪ endl ≪ endl ≪ "<!DOCTYPE␣HTML␣PUBLIC␣" ≪
    "\"-//W3C//DTD␣HTML␣4.01␣Transitional//EN\"" ≪ endl ≪
    "\"http://www.w3.org/TR/html4/loose.dtd\">" ≪ endl ≪ "<!␣file:///u|/creators.html>" ≪
    "<html>\n<head>\n<title>\nCreators␣List\n</title>" ≪ endl ≪
    "</head>\n<body>\n<font␣color=\"black\">" ≪ "\n<h1␣align=\"cente\
    r\">\n<a␣name=\"Top\">␣Creators␣List" ≪ "\n</a>\n</h1>\n</font>\n\n" ≪
    copyright_html_str ≪ "<hr␣size=\"2\"␣color=\"black\">\n\n" ≪ "\n<br>\n<br>\n" ≪
    "<b>Sort␣Order:</b>␣" ≪ "\n<br>\n<br>\n";
temp_strm ≪ "delete␣Creators_Temp␣insert␣" ≪ "Creators_Temp␣selec\
    t␣distinct␣C.creator_id,␣C.dc_creator,␣" ≪ "C.institution_id,␣C.person_id␣" ≪
    "from␣Creators␣as␣C,␣Records␣as␣R,␣Records_Creators␣as␣RC␣" ≪
    "where␣C.creator_id␣>␣0␣and␣RC.creator_id␣=␣C.creator_id␣" ≪
    "and␣RC.record_id␣=␣R.record_id␣";
if (timespan ≠ ALL_RECORDS) temp_strm ≪ "and␣" ≪ timespan_strm.str();
temp_strm ≪ "order␣by␣dc_creator␣" ≪ sort_order_str;
cdb→ExecuteSQL(temp_strm.str().c_str());
temp_strm.str("");
creators_temp.Open();
unsigned long creator_ctr;
while (¬creators_temp.IsEOF()) {
  creators_temp.MoveNext();
}
creator_ctr = creators_temp.GetRecordCount();
if (creator_ctr ≡ 0) {
  html_strm ≪ "<b>No␣Creators␣found.</b>\n<br>\n<br>\n" ≪
      "<hr␣size=\"2\"␣color=\"black\">\n\n<br>\n";
  return 0;
}
else ret_val = creator_ctr;
html_strm ≪ "<b>Creators␣found:</b>␣" ≪ creator_ctr ≪ "\n<br>\n<br>\n" ≪
    "<hr␣size=\"2\"␣color=\"black\">\n\n<br>\n";
creators_temp.MoveFirst();
creator_ctr = 1;
while (¬creators_temp.IsEOF()) {
  html_strm ≪ creator_ctr++ ≪ ".␣" ≪ creators_temp.m_dc_creator ≪ "␣";
  temp_strm ≪ "delete␣temp_ids␣" ≪ "insert␣temp_ids␣select␣R.record_id␣" ≪
      "from␣records␣as␣R,␣records_creators␣as␣RC,␣" ≪ "Creators␣as␣␣C␣" ≪
      "where␣C.creator_id␣=␣" ≪ creators_temp.m_creator_id ≪ "␣" ≪ "and␣RC.creator_id␣=␣" ≪
      creators_temp.m_creator_id ≪ "␣" ≪ "and␣RC.record_id␣=␣R.record_id";
  cdb→ExecuteSQL(temp_strm.str().c_str());
  temp_strm.str("");
  temp_ids.Open();
  while (¬temp_ids.IsEOF()) {
    temp_ids.MoveNext();
  }   /* while (¬temp_ids.IsEOF()) */
  if (temp_ids.GetRecordCount() > 0) {
    temp_ids.MoveFirst();
    while (¬temp_ids.IsEOF()) {
```

```
        html_strm ≪ "[" ≪ "<a␣href=\"./listing_complete.html#Result_" ≪ temp_ids.m_temp_id ≪
            "\">" ≪ temp_ids.m_temp_id ≪ "</a>]␣";
        temp_ids.MoveNext();
      }   /* while (¬temp_ids.IsEOF()) */
    }   /* if (temp_ids.GetRecordCount() > 0) */
    temp_ids.Close();
    html_strm ≪ "\n<br>\n<br>\n\n";
    creators_temp.MoveNext();
  }   /* while (¬creators_temp.IsEOF()) */
  creators_temp.Close();
```

**495.** Write trailer to *html_strm* and close it. [LDF Undated.]

⟨ Define **Selector** functions 388 ⟩ +≡

```
  html_strm ≪ "\n\n<br>\n<br>\n" ≪ "<hr␣size=\"2\"␣color=\"black\">\n\n" ≪
      "</font>\n</body>\n</html>\n";
  html_strm.close(); }   /* if (table ≡ CREATORS) */
  else if (table ≡ SUBJECTS) { html_strm.open("subjects.html");
```

**496.** Write header to *html_strm*. [LDF Undated.]

⟨ Define **Selector** functions 388 ⟩ +≡

```
  html_strm ≪ "<!--␣subjects.html␣-->" ≪ endl ≪ endl ≪ "<!DOCTYPE␣HTML␣PUBLIC␣" ≪
      "\"-//W3C//DTD␣HTML␣4.01␣Transitional//EN\"" ≪ endl ≪
      "\"http://www.w3.org/TR/html4/loose.dtd\">" ≪ endl ≪ "<!␣file:///u|/subjects.html>" ≪
      "<html>\n<head>\n<title>\nSubjects␣List\n</title>" ≪ endl ≪
      "</head>\n<body>\n<font␣color=\"black\">" ≪ "\n<h1␣align=\"cente\
      r\">\n<a␣name=\"Top\">␣Subjects␣List" ≪ "\n</a>\n</h1>\n</font>\n\n" ≪
      copyright_html_str ≪ "<hr␣size=\"2\"␣color=\"black\">\n\n" ≪ "\n<br>\n<br>\n" ≪
      "<b>Sort␣Order:</b>␣" ≪ "\n<br>\n<br>\n";
  if (timespan ≡ ALL_RECORDS)
    temp_strm ≪ "delete␣Temp_Ids␣insert␣Temp_Ids␣" ≪ "select␣subject_id␣from␣" ≪
        "Subjects␣" ≪ "order␣by␣dc_subject";
  else temp_strm ≪ "delete␣Temp_Ids␣insert␣Temp_Ids␣" ≪ "select␣S.subject_id␣from␣" ≪
        "Subjects␣as␣S␣" ≪ ",␣Records␣as␣R,␣Records_Subjects␣as␣RS␣" ≪
        "where␣R.record_id␣=␣RS.record_id␣" ≪ "and␣S.subject_id␣=␣RS.subject_id␣" ≪
        "and␣" ≪ timespan_strm.str() ≪ "order␣by␣S.dc_subject";
#if 0
  AfxMessageBox(temp_strm.str().c_str());
#endif
  cdb→ExecuteSQL(temp_strm.str().c_str());
  temp_strm.str("");
  temp_ids.Open();
  unsigned long subject_ctr = 0;
  set⟨unsigned long⟩ used_ids;
  set⟨unsigned long⟩::iterator iter;
  while (¬temp_ids.IsEOF()) {
    iter = used_ids.find(temp_ids.m_temp_id);
    if (iter ≡ used_ids.end()) {
      ++subject_ctr;
      used_ids.insert(temp_ids.m_temp_id);
    }
    temp_ids.MoveNext();
  }
  if (subject_ctr ≡ 0) {
    html_strm ≪ "<b>No␣Subjects␣found.</b>\n<br>\n<br>\n" ≪
        "<hr␣size=\"2\"␣color=\"black\">\n\n<br>\n";
    return 0;
  }
  else ret_val = subject_ctr;
  html_strm ≪ "<b>Subjects␣found:</b>␣" ≪ subject_ctr ≪ "\n<br>\n<br>\n" ≪
      "<hr␣size=\"2\"␣color=\"black\">\n\n<br>\n";
  temp_ids.MoveFirst();
#if 0
  temp_strm ≪ "Unique␣subjects␣found:␣" ≪ subject_ctr;
  AfxMessageBox(temp_strm.str().c_str());
  temp_strm.str("");
#endif
  subject_ctr = 1;
  used_ids.clear();
  subjects.Open();
```

```
while (¬temp_ids.IsEOF()) {
  iter = used_ids.find(temp_ids.m_temp_id);
  if (iter ≡ used_ids.end()) {
    used_ids.insert(temp_ids.m_temp_id);
  }
  else {
    temp_ids.MoveNext();
    continue;
  }
  subjects.SetAbsolutePosition(temp_ids.m_temp_id);
  html_strm ≪ subject_ctr++ ≪ ".␣" ≪ subjects.m_dc_subject ≪ "␣";
  temp_strm ≪ "delete␣temp_ids_1␣" ≪ "insert␣temp_ids_1␣select␣R.record_id␣" ≪
      "from␣records␣as␣R,␣records_subjects␣as␣RS,␣" ≪ "Subjects␣as␣␣S␣" ≪
      "where␣S.subject_id␣=␣" ≪ temp_ids.m_temp_id ≪ "␣" ≪ "and␣RS.subject_id␣=␣" ≪
      temp_ids.m_temp_id ≪ "␣" ≪ "and␣RS.record_id␣=␣R.record_id";
  cdb→ExecuteSQL(temp_strm.str().c_str());
  temp_strm.str("");
  temp_ids_1.Open();
  while (¬temp_ids_1.IsEOF()) {
    temp_ids_1.MoveNext();
  }    /* while (¬temp_ids_1.IsEOF()) */
  if (temp_ids_1.GetRecordCount() > 0) {
    temp_ids_1.MoveFirst();
    while (¬temp_ids_1.IsEOF()) {
      html_strm ≪ "[" ≪ "<a␣href=\"./listing_complete.html#Result_" ≪
          temp_ids_1.m_temp_id ≪ "\">" ≪ temp_ids_1.m_temp_id ≪ "</a>]␣";
      temp_ids_1.MoveNext();
    }    /* while (¬temp_ids_1.IsEOF()) */
  }    /* if (temp_ids_1.GetRecordCount() > 0) */
  html_strm ≪ "\n<br>\n<br>\n\n";
  temp_ids.MoveNext();
  temp_ids_1.Close();
}    /* while (¬temp_ids.IsEOF()) */
temp_ids.Close();
subjects.Close();
```

**497.** Write trailer to *html_strm* and close it. [LDF Undated.]

⟨ Define **Selector** functions 388 ⟩ +≡

```
html_strm ≪ "\n\n<br>\n<br>\n" ≪ "<hr␣size=\"2\"␣color=\"black\">\n\n" ≪
    "</font>\n</body>\n</html>\n";
html_strm.close(); }    /* if (table ≡ SUBJECTS) */
else if (table ≡ TITLES) { html_strm.open("titles.html");
```

**498.** Write header to *html_strm*. [LDF Undated.]

⟨ Define **Selector** functions 388 ⟩ +≡

```
html_strm ≪ "<!--␣titles.html␣-->" ≪ endl ≪ endl ≪ "<!DOCTYPE␣HTML␣PUBLIC␣" ≪
    "\"-//W3C//DTD␣HTML␣4.01␣Transitional//EN\"" ≪ endl ≪
    "\"http://www.w3.org/TR/html4/loose.dtd\">" ≪ endl ≪
    "<!␣file:///u|/titles.html>" ≪ "<html>\n<head>\n<title>\nTitles␣List\n</title>\n" ≪
    "</head>\n<body>\n<font␣color=\"black\">" ≪ "\n<h1␣align=\"cente\
    r\">\n<a␣name=\"Top\">␣Titles␣List" ≪ "\n</a>\n</h1>\n</font>\n\n" ≪
    copyright_html_str ≪ "<hr␣size=\"2\"␣color=\"black\">\n\n" ≪ "\n<br>\n<br>\n" ≪
    "<b>Sort␣Order:</b>␣" ≪ "\n<br>\n<br>\n";
temp_strm ≪ "delete␣temp_ids␣insert␣temp_ids␣select␣T.title_id␣" ≪
    "from␣titles␣as␣T,␣Records␣as␣R␣where␣T.record_id␣=␣R.record_id␣";
if (timespan ≠ ALL_RECORDS) temp_strm ≪ "and␣" ≪ timespan_strm.str() ≪ "␣";
temp_strm ≪ "order␣by␣dc_title␣" ≪ sort_order_str;
cdb→ExecuteSQL(temp_strm.str().c_str());
temp_strm.str("");
temp_ids.Open();
unsigned long title_ctr;
while (¬temp_ids.IsEOF()) {
  temp_ids.MoveNext();
}
title_ctr = temp_ids.GetRecordCount();
if (title_ctr ≡ 0) {
  html_strm ≪ "<b>No␣Titles␣found.</b>\n<br>\n<br>\n" ≪
      "<hr␣size=\"2\"␣color=\"black\">\n\n<br>\n";
  return 0;
}
else ret_val = title_ctr;
html_strm ≪ "<b>Titles␣found:</b>␣" ≪ title_ctr ≪ "\n<br>\n<br>\n" ≪
    "<hr␣size=\"2\"␣color=\"black\">\n\n<br>\n";
temp_ids.MoveFirst();
title_ctr = 1;
titles.Open();
while (¬temp_ids.IsEOF()) {
  titles.SetAbsolutePosition(temp_ids.m_temp_id);
  html_strm ≪ title_ctr++ ≪ ".␣" ≪ "<a␣href=\"./listing_complete.html#Result_" ≪
      temp_ids.m_temp_id ≪ "\">␣" ≪ titles.m_dc_title ≪ "</a>\n<br>\n<br>\n";
  temp_ids.MoveNext();
}
titles.Close();
temp_ids.Close();
```

**499.** Write trailer to *html_strm* and close it. [LDF Undated.]

⟨ Define **Selector** functions 388 ⟩ +≡

```
html_strm ≪ "\n\n<br>\n<br>\n" ≪ "<hr␣size=\"2\"␣color=\"black\">\n\n" ≪
    "</font>\n</body>\n</html>\n";
html_strm.close(); }    /* if (table ≡ TITLES) */
return ret_val; }    /* End of int Selector::list_table definition. */
```

**500. Putting Selector together.**

**501.** This is what's compiled.

⟨ Include files 20 ⟩
⟨ Preprocessor macro definitions 18 ⟩
⟨ Initialize **static** constants in **class Selector** 385 ⟩
⟨ Define **Selector** functions 388 ⟩

**502.** This is what's written to `selector.h`.

⟨ `selector.h` 502 ⟩ ≡
  ⟨ Preprocessor macro calls 17 ⟩
  ⟨ Forward declarations 383 ⟩
  ⟨ Declare **class Selector** 384 ⟩

**503. Records (`records.web`).**

**Log**

[LDF 2006.10.04.] Created this file. It contains code formerly in `Records.h` and `Records.cpp`.

---

⟨ `records.web` 503 ⟩ ≡ /* Empty section for use in cross-references. */

This code is cited in sections 7 and 9.

This code is used in section 881.

**504. Preprocessor macro calls.**

⟨ Preprocessor macro calls 17 ⟩ +≡
**#pragma** *once*

**505. using declarations for namespaces.**

⟨ **using** declarations for namespaces 505 ⟩ ≡
  **using namespace std**;

See also sections 526, 547, 568, 589, 610, 631, 652, 673, 694, 715, 736, 757, 778, 799, 820, 841, and 862.

This code is used in sections 522, 543, 564, 585, 606, 627, 648, 669, 690, 711, 732, 753, 774, 795, 816, 837, 858, and 879.

**506. Include files.**

⟨ Include files 20 ⟩ +≡
**#include** `"stdafx.h"`

**507. Records class declaration.**

**Log**

[LDF 2006.06.16.] Added this declaration. It was generated by Visual Studio.

---

⟨ Declare **class Records** 507 ⟩ ≡
  **class Records** : **public CRecordset** {
  **public**: **long** *m_record_id*;
    **CStringA** *m_header_identifier*;
    **CTime** *m_header_datestamp*;
    **CStringA** *m_header_status*;
    **CTime** *m_dc_date*;
    ⟨ Declare **Records** functions 509 ⟩
  };

This code is used in section 522.

**508. Functions.**

**509. Constructor.**

⟨ Declare **Records** functions 509 ⟩ ≡

```
Records(CDatabase *pDatabase = NULL);
DECLARE_DYNAMIC(Records)
```

See also sections 511, 513, 515, 517, and 519.
This code is used in section 507.

**510.**

```
⟨ Define Records functions 510 ⟩ ≡
  IMPLEMENT_DYNAMIC(Records, CRecordset)
  Records::Records(CDatabase *pdb)
  : CRecordset(pdb) {
    m_record_id = 0;
    m_header_identifier = "";
    m_header_datestamp;
    m_header_status = "";
    m_dc_date;
    m_nFields = 5;
    m_nDefaultType = dynaset;
  }
```

See also sections 512, 514, 516, 518, and 520.
This code is used in section 523.

**511. Get default connect.** [LDF 2006.06.16.]

```
⟨ Declare Records functions 509 ⟩ +≡
  virtual CString GetDefaultConnect();
```

**512.**

```
⟨ Define Records functions 510 ⟩ +≡
  CString Records::GetDefaultConnect()
  {
    return _T(DATABASE_CONNECTION_STRING);
  }
```

**513. Standard-SQL for Recordset.** [LDF 2006.06.16.]

```
⟨ Declare Records functions 509 ⟩ +≡
  virtual CString GetDefaultSQL();
```

**514.**

```
⟨ Define Records functions 510 ⟩ +≡
  CString Records::GetDefaultSQL()
  {
    return _T("[dbo].[Records]");
  }
```

**515. Do Field Exchange.** RFX-Unterstützung. [LDF 2006.06.16.]

```
⟨ Declare Records functions 509 ⟩ +≡
  virtual void DoFieldExchange(CFieldExchange *pFX);
```

**516.**

⟨ Define **Records** functions 510 ⟩ +≡

```
void Records::DoFieldExchange(CFieldExchange *pFX)
{
  pFX->SetFieldType(CFieldExchange::outputColumn);
  RFX_Long(pFX, _T("[record_id]"), m_record_id);
  RFX_Text(pFX, _T("[header_identifier]"), m_header_identifier, 1023);
  RFX_Date(pFX, _T("[header_datestamp]"), m_header_datestamp);
  RFX_Text(pFX, _T("[header_status]"), m_header_status, 1023);
  RFX_Date(pFX, _T("[dc_date]"), m_dc_date);
}
```

**517. Assert Valid.** [LDF 2006.06.16.]

⟨ Declare **Records** functions 509 ⟩ +≡

```
#ifdef _DEBUG
  virtual void AssertValid() const;
#endif
```

**518.**

⟨ Define **Records** functions 510 ⟩ +≡

```
#ifdef _DEBUG
  void Records::AssertValid() const
  {
    CRecordset::AssertValid();
  }
#endif
```

**519. Dump.** [LDF 2006.06.16.]

⟨ Declare **Records** functions 509 ⟩ +≡

```
#ifdef _DEBUG
  virtual void Dump(CDumpContext &dc) const;
#endif
```

**520.**

⟨ Define **Records** functions 510 ⟩ +≡

```
#ifdef _DEBUG
  void Records::Dump(CDumpContext &dc) const
  {
    CRecordset::Dump(dc);
  }
#endif
```

**521. Putting Records together.**

**522.** This is what gets written to `records.h`.

⟨ `records.h` 522 ⟩ ≡
  ⟨ Preprocessor macro calls 17 ⟩
  ⟨ **using** declarations for namespaces 505 ⟩
  ⟨ Declare **class Records** 507 ⟩

**523.** This is what gets compiled.

⟨ Include files 20 ⟩
⟨ Define **Records** functions 510 ⟩

**524. Records_Temp** (`rcrdstmp.web`).

**Log**

[LDF 2006.10.09.] Created this file. It contains code formerly in `Records_Temp.h` and `Records_Temp.cpp`.

---

⟨ `rcrdstmp.web` 524 ⟩ ≡ /* Empty section for use in cross-references. */

This code is cited in sections 7 and 9.

This code is used in section 881.

**525. Preprocessor macro calls.**

⟨ Preprocessor macro calls 17 ⟩ +≡
**#pragma** *once*

**526. using declarations for namespaces.**

⟨ **using** declarations for namespaces 505 ⟩ +≡
**using namespace std**;

**527. Include files.**

⟨ Include files 20 ⟩ +≡
**#include** `"stdafx.h"`

**528. Records_Temp class declaration.**

**Log**

[LDF 2006.06.20.] Added this declaration. It was generated by Visual Studio.

---

⟨ Declare **class Records_Temp** 528 ⟩ ≡
**class Records_Temp** : **public CRecordset** {
**public**: **long** *m_record_id*;
**CStringA** *m_header_identifier*;
**CTime** *m_header_datestamp*;
**CStringA** *m_header_status*;
**CTime** *m_dc_date*;
⟨ Declare **Records_Temp** functions 530 ⟩
};

This code is used in section 543.

**529. Functions.**

**530. Constructor.**

⟨ Declare **Records_Temp** functions 530 ⟩ ≡
**Records_Temp**(**CDatabase** **pDatabase* = `NULL`);
`DECLARE_DYNAMIC`(**Records_Temp**)

See also sections 532, 534, 536, 538, and 540.

This code is used in section 528.

**531.**

⟨ Define **Records_Temp** functions 531 ⟩ ≡

```
IMPLEMENT_DYNAMIC(Records_Temp, CRecordset)
Records_Temp :: Records_Temp(CDatabase *pdb)
: CRecordset(pdb) {
  m_record_id = 0;
  m_header_identifier = "";
  m_header_datestamp;
  m_header_status = "";
  m_dc_date;
  m_nFields = 5;
  m_nDefaultType = dynaset;
}
```

See also sections 533, 535, 537, 539, and 541.

This code is used in section 544.

**532. Get default connect.** [LDF 2006.06.20.]

⟨ Declare **Records_Temp** functions 530 ⟩ +≡

```
virtual CString GetDefaultConnect( );
```

**533.**

⟨ Define **Records_Temp** functions 531 ⟩ +≡

```
CString Records_Temp :: GetDefaultConnect( )
{
  return _T(DATABASE_CONNECTION_STRING);
}
```

**534. Standard-SQL for Recordset.** [LDF 2006.06.20.]

⟨ Declare **Records_Temp** functions 530 ⟩ +≡

```
virtual CString GetDefaultSQL( );
```

**535.**

⟨ Define **Records_Temp** functions 531 ⟩ +≡

```
CString Records_Temp :: GetDefaultSQL( )
{
  return _T("[dbo].[Records_Temp]");
}
```

**536. Do Field Exchange.** RFX-Unterstützung. [LDF 2006.06.20.]

⟨ Declare **Records_Temp** functions 530 ⟩ +≡

```
virtual void DoFieldExchange(CFieldExchange *pFX);
```

**537.**

⟨ Define **Records_Temp** functions 531 ⟩ +≡

```
void Records_Temp::DoFieldExchange(CFieldExchange *pFX)
{
  pFX->SetFieldType(CFieldExchange::outputColumn);
  RFX_Long(pFX, _T("[record_id]"), m_record_id);
  RFX_Text(pFX, _T("[header_identifier]"), m_header_identifier, 1023);
  RFX_Date(pFX, _T("[header_datestamp]"), m_header_datestamp);
  RFX_Text(pFX, _T("[header_status]"), m_header_status, 1023);
  RFX_Date(pFX, _T("[dc_date]"), m_dc_date);
}
```

**538. Assert Valid.** [LDF 2006.06.20.]

⟨ Declare **Records_Temp** functions 530 ⟩ +≡

```
#ifdef _DEBUG
  virtual void AssertValid() const;
#endif
```

**539.**

⟨ Define **Records_Temp** functions 531 ⟩ +≡

```
#ifdef _DEBUG
  void Records_Temp::AssertValid() const
  {
    CRecordset::AssertValid();
  }
#endif
```

**540. Dump.** [LDF 2006.06.20.]

⟨ Declare **Records_Temp** functions 530 ⟩ +≡

```
#ifdef _DEBUG
  virtual void Dump(CDumpContext &dc) const;
#endif
```

**541.**

⟨ Define **Records_Temp** functions 531 ⟩ +≡

```
#ifdef _DEBUG
  void Records_Temp::Dump(CDumpContext &dc) const
  {
    CRecordset::Dump(dc);
  }
#endif
```

**542. Putting Records_Temp together.**

**543.** This is what gets written to `rcrdstmp.h`.

⟨ `rcrdstmp.h` 543 ⟩ ≡
  ⟨ Preprocessor macro calls 17 ⟩
  ⟨ **using** declarations for namespaces 505 ⟩
  ⟨ Declare **class Records_Temp** 528 ⟩

**544.** This is what gets compiled.

⟨ Include files 20 ⟩
⟨ Define **Records_Temp** functions 531 ⟩

**545. Creators (`creators.web`).**

**Log**

[LDF 2006.10.04.] Created this file. It contains code formerly in `Creators.h` and `Creators.cpp`.

---

⟨ `creators.web` 545 ⟩ ≡ /* Empty section for use in cross-references. */

This code is cited in sections 7 and 9.

This code is used in section 881.

**546. Preprocessor macro calls.**

⟨ Preprocessor macro calls 17 ⟩ +≡
**#pragma** *once*

**547. using declarations for namespaces.**

⟨ **using** declarations for namespaces 505 ⟩ +≡
**using namespace std**;

**548. Include files.**

⟨ Include files 20 ⟩ +≡
**#include** `"stdafx.h"`

**549. Creators class declaration.**

**Log**

[LDF 2006.06.30.] Added this declaration. It was generated by Visual Studio.

---

⟨ Declare **class Creators** 549 ⟩ ≡

```
class Creators : public CRecordset {
public: long m_creator_id;
  CStringA m_dc_creator;
  long m_institution_id;
  long m_person_id;
  ⟨ Declare Creators functions 551 ⟩
};
```

This code is used in section 564.

**550. Functions.**

**551. Constructor.**

⟨ Declare **Creators** functions 551 ⟩ ≡

```
Creators(CDatabase *pDatabase = NULL);
DECLARE_DYNAMIC(Creators)
```

See also sections 553, 555, 557, 559, and 561.

This code is used in section 549.

**552.**

⟨ Define **Creators** functions 552 ⟩ ≡
```
IMPLEMENT_DYNAMIC(Creators, CRecordset)
Creators::Creators(CDatabase *pdb)
: CRecordset(pdb) {
  m_creator_id = 0;
  m_dc_creator = "";
  m_institution_id = 0;
  m_person_id = 0;
  m_nFields = 4;
  m_nDefaultType = dynaset;
}
```

See also sections 554, 556, 558, 560, and 562.

This code is used in section 565.

**553. Get default connect.** [LDF 2006.06.30.]

⟨ Declare **Creators** functions 551 ⟩ +≡
```
virtual CString GetDefaultConnect( );
```

**554.**

⟨ Define **Creators** functions 552 ⟩ +≡
```
CString Creators::GetDefaultConnect( )
{
  return _T(DATABASE_CONNECTION_STRING);
}
```

**555. Standard-SQL for Recordset.** [LDF 2006.06.30.]

⟨ Declare **Creators** functions 551 ⟩ +≡
```
virtual CString GetDefaultSQL( );
```

**556.**

⟨ Define **Creators** functions 552 ⟩ +≡
```
CString Creators::GetDefaultSQL( )
{
  return _T("[dbo].[Creators]");
}
```

**557. Do Field Exchange.** RFX-Unterstützung. [LDF 2006.06.30.]

⟨ Declare **Creators** functions 551 ⟩ +≡
```
virtual void DoFieldExchange(CFieldExchange *pFX);
```

**558.**

⟨ Define **Creators** functions 552 ⟩ +≡

```
void Creators::DoFieldExchange(CFieldExchange *pFX)
{
  pFX→SetFieldType(CFieldExchange::outputColumn);
  RFX_Long(pFX, _T("[creator_id]"), m_creator_id);
  RFX_Text(pFX, _T("[dc_creator]"), m_dc_creator, 1023);
  RFX_Long(pFX, _T("[institution_id]"), m_institution_id);
  RFX_Long(pFX, _T("[person_id]"), m_person_id);
}
```

**559. Assert Valid.** [LDF 2006.06.30.]

⟨ Declare **Creators** functions 551 ⟩ +≡

```
#ifdef _DEBUG
  virtual void AssertValid() const;
#endif
```

**560.**

⟨ Define **Creators** functions 552 ⟩ +≡

```
#ifdef _DEBUG
  void Creators::AssertValid() const
  {
    CRecordset::AssertValid();
  }
#endif
```

**561. Dump.** [LDF 2006.06.30.]

⟨ Declare **Creators** functions 551 ⟩ +≡

```
#ifdef _DEBUG
  virtual void Dump(CDumpContext &dc) const;
#endif
```

**562.**

⟨ Define **Creators** functions 552 ⟩ +≡

```
#ifdef _DEBUG
  void Creators::Dump(CDumpContext &dc) const
  {
    CRecordset::Dump(dc);
  }
#endif
```

**563. Putting Creators together.**

**564.** This is what gets written to `creators.h`.

⟨ `creators.h` 564 ⟩ ≡

⟨ Preprocessor macro calls 17 ⟩
⟨ **using** declarations for namespaces 505 ⟩
⟨ Declare **class Creators** 549 ⟩

**565.** This is what gets compiled.

⟨ Include files 20 ⟩
⟨ Define **Creators** functions 552 ⟩

**566. Creators_Temp (`crtrstmp.web`).**

**Log**

[LDF 2006.10.04.] Created this file. It contains code formerly in `Creators_Temp.h` and `Creators_Temp.cpp`.

---

⟨ `crtrstmp.web` 566 ⟩ ≡ /* Empty section for use in cross-references. */

This code is cited in sections 7 and 9.

This code is used in section 881.

**567. Preprocessor macro calls.**

⟨ Preprocessor macro calls 17 ⟩ +≡
**#pragma** *once*

**568. using declarations for namespaces.**

⟨ **using** declarations for namespaces 505 ⟩ +≡
**using namespace std**;

**569. Include files.**

⟨ Include files 20 ⟩ +≡
**#include** `"stdafx.h"`

**570. Creators_Temp class declaration.**

**Log**

[LDF 2006.06.21.] Added this declaration. It was generated by Visual Studio.

---

⟨ Declare **class Creators_Temp** 570 ⟩ ≡
**class Creators_Temp** : **public CRecordset** {
**public**: **long** *m_creator_id*;
**CStringA** *m_dc_creator*;
**long** *m_institution_id*;
**long** *m_person_id*;
⟨ Declare **Creators_Temp** functions 572 ⟩
};

This code is used in section 585.

**571. Functions.**

**572. Constructor.**

⟨ Declare **Creators_Temp** functions 572 ⟩ ≡
**Creators_Temp**(**CDatabase** **pDatabase* = `NULL`);
`DECLARE_DYNAMIC`(**Creators_Temp**)

See also sections 574, 576, 578, 580, and 582.

This code is used in section 570.

**573.**

⟨ Define **Creators_Temp** functions 573 ⟩ ≡

```
IMPLEMENT_DYNAMIC(Creators_Temp, CRecordset)
Creators_Temp::Creators_Temp(CDatabase *pdb)
: CRecordset(pdb) {
  m_creator_id = 0;
  m_dc_creator = "";
  m_institution_id = 0;
  m_person_id = 0;
  m_nFields = 4;
  m_nDefaultType = dynaset;
}
```

See also sections 575, 577, 579, 581, and 583.

This code is used in section 586.

**574. Get default connect.** [LDF 2006.06.21.]

⟨ Declare **Creators_Temp** functions 572 ⟩ +≡

```
virtual CString GetDefaultConnect( );
```

**575.**

⟨ Define **Creators_Temp** functions 573 ⟩ +≡

```
CString Creators_Temp::GetDefaultConnect( )
{
  return _T(DATABASE_CONNECTION_STRING);
}
```

**576. Standard-SQL for Recordset.** [LDF 2006.06.21.]

⟨ Declare **Creators_Temp** functions 572 ⟩ +≡

```
virtual CString GetDefaultSQL( );
```

**577.**

⟨ Define **Creators_Temp** functions 573 ⟩ +≡

```
CString Creators_Temp::GetDefaultSQL( )
{
  return _T("[dbo].[Creators_Temp]");
}
```

**578. Do Field Exchange.** RFX-Unterstützung. [LDF 2006.06.21.]

⟨ Declare **Creators_Temp** functions 572 ⟩ +≡

```
virtual void DoFieldExchange(CFieldExchange *pFX);
```

**579.**

⟨ Define **Creators_Temp** functions 573 ⟩ +≡

```
void Creators_Temp::DoFieldExchange(CFieldExchange *pFX)
{
  pFX->SetFieldType(CFieldExchange::outputColumn);
  RFX_Long(pFX, _T("[creator_id]"), m_creator_id);
  RFX_Text(pFX, _T("[dc_creator]"), m_dc_creator, 1023);
  RFX_Long(pFX, _T("[institution_id]"), m_institution_id);
  RFX_Long(pFX, _T("[person_id]"), m_person_id);
}
```

**580. Assert Valid.** [LDF 2006.06.21.]

⟨ Declare **Creators_Temp** functions 572 ⟩ +≡

```
#ifdef _DEBUG
  virtual void AssertValid() const;
#endif
```

**581.**

⟨ Define **Creators_Temp** functions 573 ⟩ +≡

```
#ifdef _DEBUG
  void Creators_Temp::AssertValid() const
  {
    CRecordset::AssertValid();
  }
#endif
```

**582. Dump.** [LDF 2006.06.21.]

⟨ Declare **Creators_Temp** functions 572 ⟩ +≡

```
#ifdef _DEBUG
  virtual void Dump(CDumpContext &dc) const;
#endif
```

**583.**

⟨ Define **Creators_Temp** functions 573 ⟩ +≡

```
#ifdef _DEBUG
  void Creators_Temp::Dump(CDumpContext &dc) const
  {
    CRecordset::Dump(dc);
  }
#endif
```

**584. Putting Creators_Temp together.**

**585.** This is what gets written to `crtrstmp.h`.

⟨ `crtrstmp.h` 585 ⟩ ≡
  ⟨ Preprocessor macro calls 17 ⟩
  ⟨ **using** declarations for namespaces 505 ⟩
  ⟨ Declare **class Creators_Temp** 570 ⟩

**586.** This is what gets compiled.

⟨ Include files 20 ⟩
⟨ Define **Creators_Temp** functions 573 ⟩

**587. Contributors (`cntrbtrs.web`).**

**Log**

[LDF 2006.10.04.] Created this file. It contains code formerly in `Contributors.h` and `Contributors.cpp`.

---

⟨ `cntrbtrs.web` 587 ⟩ ≡ /* Empty section for use in cross-references. */

This code is cited in sections 7 and 9.

This code is used in section 881.

**588. Preprocessor macro calls.**

⟨ Preprocessor macro calls 17 ⟩ +≡
**#pragma** *once*

**589. using declarations for namespaces.**

⟨ **using** declarations for namespaces 505 ⟩ +≡
**using namespace std**;

**590. Include files.**

⟨ Include files 20 ⟩ +≡
**#include** `"stdafx.h"`

**591. Contributors class declaration.**

**Log**

[LDF 2006.06.30.] Added this declaration. It was generated by Visual Studio.

---

⟨ Declare **class Contributors** 591 ⟩ ≡

```
class Contributors : public CRecordset {
public: long m_contributor_id;
  CStringA m_dc_contributor;
  long m_institution_id;
  long m_person_id;
  ⟨ Declare Contributors functions 593 ⟩
};
```

This code is used in section 606.

**592. Functions.**

**593. Constructor.**

⟨ Declare **Contributors** functions 593 ⟩ ≡

```
Contributors(CDatabase *pDatabase = NULL);
DECLARE_DYNAMIC(Contributors)
```

See also sections 595, 597, 599, 601, and 603.

This code is used in section 591.

**594.**

⟨ Define **Contributors** functions 594 ⟩ ≡
```
IMPLEMENT_DYNAMIC(Contributors, CRecordset)
Contributors::Contributors(CDatabase *pdb)
: CRecordset(pdb) {
  m_contributor_id = 0;
  m_dc_contributor = "";
  m_institution_id = 0;
  m_person_id = 0;
  m_nFields = 4;
  m_nDefaultType = dynaset;
}
```
See also sections 596, 598, 600, 602, and 604.

This code is used in section 607.

**595. Get default connect.** [LDF 2006.06.30.]

⟨ Declare **Contributors** functions 593 ⟩ +≡
```
virtual CString GetDefaultConnect( );
```

**596.**

⟨ Define **Contributors** functions 594 ⟩ +≡
```
CString Contributors::GetDefaultConnect( )
{
  return _T(DATABASE_CONNECTION_STRING);
}
```

**597. Standard-SQL for Recordset.** [LDF 2006.06.30.]

⟨ Declare **Contributors** functions 593 ⟩ +≡
```
virtual CString GetDefaultSQL( );
```

**598.**

⟨ Define **Contributors** functions 594 ⟩ +≡
```
CString Contributors::GetDefaultSQL( )
{
  return _T("[dbo].[Contributors]");
}
```

**599. Do Field Exchange.** RFX-Unterstützung. [LDF 2006.06.30.]

⟨ Declare **Contributors** functions 593 ⟩ +≡
```
virtual void DoFieldExchange(CFieldExchange *pFX);
```

**600.**

⟨ Define **Contributors** functions 594 ⟩ +≡

```
void Contributors::DoFieldExchange(CFieldExchange *pFX)
{
  pFX→SetFieldType(CFieldExchange::outputColumn);
  RFX_Long(pFX, _T("[contributor_id]"), m_contributor_id);
  RFX_Text(pFX, _T("[dc_contributor]"), m_dc_contributor, 1023);
  RFX_Long(pFX, _T("[institution_id]"), m_institution_id);
  RFX_Long(pFX, _T("[person_id]"), m_person_id);
}
```

**601. Assert Valid.** [LDF 2006.06.30.]

⟨ Declare **Contributors** functions 593 ⟩ +≡

```
#ifdef _DEBUG
  virtual void AssertValid() const;
#endif
```

**602.**

⟨ Define **Contributors** functions 594 ⟩ +≡

```
#ifdef _DEBUG
  void Contributors::AssertValid() const
  {
    CRecordset::AssertValid();
  }
#endif
```

**603. Dump.** [LDF 2006.06.30.]

⟨ Declare **Contributors** functions 593 ⟩ +≡

```
#ifdef _DEBUG
  virtual void Dump(CDumpContext &dc) const;
#endif
```

**604.**

⟨ Define **Contributors** functions 594 ⟩ +≡

```
#ifdef _DEBUG
  void Contributors::Dump(CDumpContext &dc) const
  {
    CRecordset::Dump(dc);
  }
#endif
```

**605. Putting Contributors together.**

**606.** This is what gets written to `cntrbtrs.h`.

⟨ `cntrbtrs.h` 606 ⟩ ≡
  ⟨ Preprocessor macro calls 17 ⟩
  ⟨ **using** declarations for namespaces 505 ⟩
  ⟨ Declare **class Contributors** 591 ⟩

**607.** This is what gets compiled.

⟨ Include files 20 ⟩
⟨ Define **Contributors** functions 594 ⟩

**608. Contributors_Temp (`cntrbtmp.web`).**

**Log**

[LDF 2006.10.04.] Created this file. It contains code formerly in `Contributors_Temp.h` and `Contributors_Temp.cpp`.

---

⟨ `cntrbtmp.web` 608 ⟩ ≡ /* Empty section for use in cross-references. */

This code is cited in sections 7 and 9.

This code is used in section 881.

**609. Preprocessor macro calls.**

⟨ Preprocessor macro calls 17 ⟩ +≡
**#pragma** *once*

**610. using declarations for namespaces.**

⟨ **using** declarations for namespaces 505 ⟩ +≡
**using namespace std**;

**611. Include files.**

⟨ Include files 20 ⟩ +≡
**#include** `"stdafx.h"`

**612. Contributors_Temp class declaration.**

**Log**

[LDF 2006.06.22.] Added this declaration. It was generated by Visual Studio.

---

⟨ Declare **class Contributors_Temp** 612 ⟩ ≡

```
class Contributors_Temp : public CRecordset {
public: long m_contributor_id;
    CStringA m_dc_contributor;
    long m_institution_id;
    long m_person_id;
    ⟨ Declare Contributors_Temp functions 614 ⟩
};
```

This code is used in section 627.

**613. Functions.**

**614. Constructor.**

⟨ Declare **Contributors_Temp** functions 614 ⟩ ≡

```
Contributors_Temp(CDatabase *pDatabase = NULL);
DECLARE_DYNAMIC(Contributors_Temp)
```

See also sections 616, 618, 620, 622, and 624.

This code is used in section 612.

**615.**

⟨ Define **Contributors_Temp** functions 615 ⟩ ≡
```
IMPLEMENT_DYNAMIC(Contributors_Temp, CRecordset)
Contributors_Temp::Contributors_Temp(CDatabase *pdb)
: CRecordset(pdb) {
  m_contributor_id = 0;
  m_dc_contributor = "";
  m_institution_id = 0;
  m_person_id = 0;
  m_nFields = 4;
  m_nDefaultType = dynaset;
}
```
See also sections 617, 619, 621, 623, and 625.
This code is used in section 628.

**616. Get default connect.** [LDF 2006.06.22.]

⟨ Declare **Contributors_Temp** functions 614 ⟩ +≡
```
virtual CString GetDefaultConnect();
```

**617.**

⟨ Define **Contributors_Temp** functions 615 ⟩ +≡
```
CString Contributors_Temp::GetDefaultConnect()
{
  return _T(DATABASE_CONNECTION_STRING);
}
```

**618. Standard-SQL for Recordset.** [LDF 2006.06.22.]

⟨ Declare **Contributors_Temp** functions 614 ⟩ +≡
```
virtual CString GetDefaultSQL();
```

**619.**

⟨ Define **Contributors_Temp** functions 615 ⟩ +≡
```
CString Contributors_Temp::GetDefaultSQL()
{
  return _T("[dbo].[Contributors_Temp]");
}
```

**620. Do Field Exchange.** RFX-Unterstützung. [LDF 2006.06.22.]

⟨ Declare **Contributors_Temp** functions 614 ⟩ +≡
```
virtual void DoFieldExchange(CFieldExchange *pFX);
```

**621.**

⟨ Define **Contributors_Temp** functions 615 ⟩ +≡

```
void Contributors_Temp::DoFieldExchange(CFieldExchange *pFX)
{
  pFX->SetFieldType(CFieldExchange::outputColumn);
  RFX_Long(pFX, _T("[contributor_id]"), m_contributor_id);
  RFX_Text(pFX, _T("[dc_contributor]"), m_dc_contributor, 1023);
  RFX_Long(pFX, _T("[institution_id]"), m_institution_id);
  RFX_Long(pFX, _T("[person_id]"), m_person_id);
}
```

**622. Assert Valid.** [LDF 2006.06.22.]

⟨ Declare **Contributors_Temp** functions 614 ⟩ +≡

```
#ifdef _DEBUG
  virtual void AssertValid() const;
#endif
```

**623.**

⟨ Define **Contributors_Temp** functions 615 ⟩ +≡

```
#ifdef _DEBUG
  void Contributors_Temp::AssertValid() const
  {
    CRecordset::AssertValid();
  }
#endif
```

**624. Dump.** [LDF 2006.06.22.]

⟨ Declare **Contributors_Temp** functions 614 ⟩ +≡

```
#ifdef _DEBUG
  virtual void Dump(CDumpContext &dc) const;
#endif
```

**625.**

⟨ Define **Contributors_Temp** functions 615 ⟩ +≡

```
#ifdef _DEBUG
  void Contributors_Temp::Dump(CDumpContext &dc) const
  {
    CRecordset::Dump(dc);
  }
#endif
```

**626. Putting Contributors_Temp together.**

**627.** This is what gets written to `cntrbtmp.h`.

⟨ `cntrbtmp.h` 627 ⟩ ≡

⟨ Preprocessor macro calls 17 ⟩
⟨ **using** declarations for namespaces 505 ⟩
⟨ Declare **class Contributors_Temp** 612 ⟩

**628.** This is what gets compiled.

⟨ Include files 20 ⟩
⟨ Define **Contributors_Temp** functions 615 ⟩

**629. Titles (`titles.web`).**

**Log**

[LDF 2006.10.05.] Created this file. It contains code formerly in `Titles.h` and `Titles.cpp`.

---

⟨ `titles.web` 629 ⟩ ≡ /* Empty section for use in cross-references. */

This code is cited in sections 7 and 9.

This code is used in section 881.

**630. Preprocessor macro calls.**

⟨ Preprocessor macro calls 17 ⟩ +≡
**#pragma** *once*

**631. using declarations for namespaces.**

⟨ **using** declarations for namespaces 505 ⟩ +≡
**using namespace std**;

**632. Include files.**

⟨ Include files 20 ⟩ +≡
**#include** `"stdafx.h"`

**633. Titles class declaration.**

**Log**

[LDF 2006.06.30.] Added this declaration. It was generated by Visual Studio.

---

⟨ Declare **class Titles** 633 ⟩ ≡
**class Titles** : **public CRecordset** {
**public**: **long** *m_title_id*;
**long** *m_record_id*;
**CStringA** *m_dc_title*;
⟨ Declare **Titles** functions 635 ⟩
};

This code is used in section 648.

**634. Functions.**

**635. Constructor.**

⟨ Declare **Titles** functions 635 ⟩ ≡
**Titles**(**CDatabase** **pDatabase* = `NULL`);
`DECLARE_DYNAMIC`(**Titles**)

See also sections 637, 639, 641, 643, and 645.

This code is used in section 633.

**636.**

⟨ Define **Titles** functions 636 ⟩ ≡
```
IMPLEMENT_DYNAMIC(Titles, CRecordset)
Titles::Titles(CDatabase *pdb)
: CRecordset(pdb) {
  m_title_id = 0;
  m_record_id = 0;
  m_dc_title = "";
  m_nFields = 3;
  m_nDefaultType = dynaset;
}
```
See also sections 638, 640, 642, 644, and 646.
This code is used in section 649.

**637. Get default connect.** [LDF 2006.06.30.]

⟨ Declare **Titles** functions 635 ⟩ +≡
```
virtual CString GetDefaultConnect( );
```

**638.**

⟨ Define **Titles** functions 636 ⟩ +≡
```
CString Titles::GetDefaultConnect( )
{
  return _T(DATABASE_CONNECTION_STRING);
}
```

**639. Standard-SQL for Recordset.** [LDF 2006.06.30.]

⟨ Declare **Titles** functions 635 ⟩ +≡
```
virtual CString GetDefaultSQL( );
```

**640.**

⟨ Define **Titles** functions 636 ⟩ +≡
```
CString Titles::GetDefaultSQL( )
{
  return _T("[dbo].[Titles]");
}
```

**641. Do Field Exchange.** RFX-Unterstützung. [LDF 2006.06.30.]

⟨ Declare **Titles** functions 635 ⟩ +≡
```
virtual void DoFieldExchange(CFieldExchange *pFX);
```

**642.**

⟨ Define **Titles** functions 636 ⟩ +≡
```
void Titles::DoFieldExchange(CFieldExchange *pFX)
{
  pFX->SetFieldType(CFieldExchange::outputColumn);
  RFX_Long(pFX, _T("[title_id]"), m_title_id);
  RFX_Long(pFX, _T("[record_id]"), m_record_id);
  RFX_Text(pFX, _T("[dc_title]"), m_dc_title, 1023);
}
```

**643. Assert Valid.** [LDF 2006.06.30.]

⟨ Declare **Titles** functions 635 ⟩ +≡
```
#ifdef _DEBUG
  virtual void AssertValid() const;
#endif
```

**644.**

⟨ Define **Titles** functions 636 ⟩ +≡
```
#ifdef _DEBUG
  void Titles::AssertValid() const
  {
    CRecordset::AssertValid();
  }
#endif
```

**645. Dump.** [LDF 2006.06.30.]

⟨ Declare **Titles** functions 635 ⟩ +≡
```
#ifdef _DEBUG
  virtual void Dump(CDumpContext &dc) const;
#endif
```

**646.**

⟨ Define **Titles** functions 636 ⟩ +≡
```
#ifdef _DEBUG
  void Titles::Dump(CDumpContext &dc) const
  {
    CRecordset::Dump(dc);
  }
#endif
```

**647. Putting Titles together.**

**648.** This is what gets written to `titles.h`.

⟨ `titles.h` 648 ⟩ ≡
  ⟨ Preprocessor macro calls 17 ⟩
  ⟨ **using** declarations for namespaces 505 ⟩
  ⟨ Declare **class Titles** 633 ⟩

**649.** This is what gets compiled.

⟨ Include files 20 ⟩
⟨ Define **Titles** functions 636 ⟩

**650. Titles_Temp (`ttlstmp.web`).**

---

**Log**

[LDF 2006.10.05.] Created this file. It contains code formerly in `Titles_Temp.h` and `Titles_Temp.cpp`.

---

⟨ `ttlstmp.web` 650 ⟩ ≡ /* Empty section for use in cross-references. */

This code is cited in sections 7 and 9.

This code is used in section 881.

**651. Preprocessor macro calls.**

⟨ Preprocessor macro calls 17 ⟩ +≡
**#pragma** *once*

**652. using declarations for namespaces.**

⟨ **using** declarations for namespaces 505 ⟩ +≡
**using namespace std**;

**653. Include files.**

⟨ Include files 20 ⟩ +≡
**#include** `"stdafx.h"`

**654. Titles_Temp class declaration.**

---

**Log**

[LDF 2006.06.20.] Added this declaration. It was generated by Visual Studio.

---

⟨ Declare **class Titles_Temp** 654 ⟩ ≡
**class Titles_Temp** : **public CRecordset** {
**public**: **long** *m_title_id*;
**long** *m_record_id*;
**CStringA** *m_dc_title*;
⟨ Declare **Titles_Temp** functions 656 ⟩
};

This code is used in section 669.

**655. Functions.**

**656. Constructor.**

⟨ Declare **Titles_Temp** functions 656 ⟩ ≡
**Titles_Temp**(**CDatabase** **pDatabase* = `NULL`);
`DECLARE_DYNAMIC`(**Titles_Temp**)

See also sections 658, 660, 662, 664, and 666.

This code is used in section 654.

**657.**

⟨ Define **Titles_Temp** functions 657 ⟩ ≡
```
IMPLEMENT_DYNAMIC(Titles_Temp, CRecordset)
Titles_Temp::Titles_Temp(CDatabase *pdb)
: CRecordset(pdb) {
  m_title_id = 0;
  m_record_id = 0;
  m_dc_title = "";
  m_nFields = 3;
  m_nDefaultType = dynaset;
}
```
See also sections 659, 661, 663, 665, and 667.

This code is used in section 670.

**658. Get default connect.** [LDF 2006.06.20.]

⟨ Declare **Titles_Temp** functions 656 ⟩ +≡
```
virtual CString GetDefaultConnect( );
```

**659.**

⟨ Define **Titles_Temp** functions 657 ⟩ +≡
```
CString Titles_Temp::GetDefaultConnect( )
{
  return _T(DATABASE_CONNECTION_STRING);
}
```

**660. Standard-SQL for Recordset.** [LDF 2006.06.20.]

⟨ Declare **Titles_Temp** functions 656 ⟩ +≡
```
virtual CString GetDefaultSQL( );
```

**661.**

⟨ Define **Titles_Temp** functions 657 ⟩ +≡
```
CString Titles_Temp::GetDefaultSQL( )
{
  return _T("[dbo].[Titles_Temp]");
}
```

**662. Do Field Exchange.** RFX-Unterstützung. [LDF 2006.06.20.]

⟨ Declare **Titles_Temp** functions 656 ⟩ +≡
```
virtual void DoFieldExchange(CFieldExchange *pFX);
```

**663.**

⟨ Define **Titles_Temp** functions 657 ⟩ +≡
  **void Titles_Temp** :: *DoFieldExchange* (**CFieldExchange** **pFX* )
  {
    *pFX*→*SetFieldType* (**CFieldExchange** :: *outputColumn*);
    *RFX_Long* (*pFX* , `_T("[title_id]")`, *m_title_id* );
    *RFX_Long* (*pFX* , `_T("[record_id]")`, *m_record_id* );
    *RFX_Text* (*pFX* , `_T("[dc_title]")`, *m_dc_title* , 1023);
  }

**664. Assert Valid.** [LDF 2006.06.20.]

⟨ Declare **Titles_Temp** functions 656 ⟩ +≡
**#ifdef** `_DEBUG`
  **virtual void** *AssertValid* ( ) **const**;
**#endif**

**665.**

⟨ Define **Titles_Temp** functions 657 ⟩ +≡
**#ifdef** `_DEBUG`
  **void Titles_Temp** :: *AssertValid* ( ) **const**
  {
    **CRecordset** :: *AssertValid* ( );
  }
**#endif**

**666. Dump.** [LDF 2006.06.20.]

⟨ Declare **Titles_Temp** functions 656 ⟩ +≡
**#ifdef** `_DEBUG`
  **virtual void** *Dump* (**CDumpContext** &*dc*) **const**;
**#endif**

**667.**

⟨ Define **Titles_Temp** functions 657 ⟩ +≡
**#ifdef** `_DEBUG`
  **void Titles_Temp** :: *Dump* (**CDumpContext** &*dc*) **const**
  {
    **CRecordset** :: *Dump* (*dc*);
  }
**#endif**

**668. Putting Titles_Temp together.**

**669.** This is what gets written to `ttlstmp.h`.

⟨ `ttlstmp.h` 669 ⟩ ≡
  ⟨ Preprocessor macro calls 17 ⟩
  ⟨ **using** declarations for namespaces 505 ⟩
  ⟨ Declare **class Titles_Temp** 654 ⟩

**670.** This is what gets compiled.

⟨ Include files 20 ⟩
⟨ Define **Titles_Temp** functions 657 ⟩

**671. Descriptions_Temp (`dscrptmp.web`).**

**Log**

[LDF 2006.10.05.] Created this file. It contains code formerly in `Descriptions_Temp.h` and `Descriptions_Temp.cpp`.

---

⟨ `dscrptmp.web` 671 ⟩ ≡ /* Empty section for use in cross-references. */

This code is cited in sections 7 and 9.

This code is used in section 881.

**672. Preprocessor macro calls.**

⟨ Preprocessor macro calls 17 ⟩ +≡
**#pragma** *once*

**673. using declarations for namespaces.**

⟨ **using** declarations for namespaces 505 ⟩ +≡
**using namespace std**;

**674. Include files.**

⟨ Include files 20 ⟩ +≡
**#include** `"stdafx.h"`

**675. Descriptions_Temp class declaration.**

**Log**

[LDF 2066.06.22.] Added this declaration. It was generated by Visual Studio.

---

⟨ Declare **class Descriptions_Temp** 675 ⟩ ≡
**class Descriptions_Temp** : **public CRecordset** {
**public**: **long** *m_description_id*;
**long** *m_record_id*;
**CStringA** *m_dc_description*;
⟨ Declare **Descriptions_Temp** functions 677 ⟩
};

This code is used in section 690.

**676. Functions.**

**677. Constructor.**

⟨ Declare **Descriptions_Temp** functions 677 ⟩ ≡
**Descriptions_Temp**(**CDatabase** **pDatabase* = NULL);
DECLARE_DYNAMIC(**Descriptions_Temp**)

See also sections 679, 681, 683, 685, and 687.

This code is used in section 675.

**678.**

⟨ Define **Descriptions_Temp** functions 678 ⟩ ≡
```
IMPLEMENT_DYNAMIC(Descriptions_Temp, CRecordset)
Descriptions_Temp::Descriptions_Temp(CDatabase *pdb)
: CRecordset(pdb) {
  m_description_id = 0;
  m_record_id = 0;
  m_dc_description = "";
  m_nFields = 3;
  m_nDefaultType = dynaset;
}
```
See also sections 680, 682, 684, 686, and 688.

This code is used in section 691.

**679. Get default connect.** [LDF 2066.06.22.]

⟨ Declare **Descriptions_Temp** functions 677 ⟩ +≡
```
virtual CString GetDefaultConnect();
```

**680.**

⟨ Define **Descriptions_Temp** functions 678 ⟩ +≡
```
CString Descriptions_Temp::GetDefaultConnect()
{
  return _T(DATABASE_CONNECTION_STRING);
}
```

**681. Standard-SQL for Recordset.** [LDF 2066.06.22.]

⟨ Declare **Descriptions_Temp** functions 677 ⟩ +≡
```
virtual CString GetDefaultSQL();
```

**682.**

⟨ Define **Descriptions_Temp** functions 678 ⟩ +≡
```
CString Descriptions_Temp::GetDefaultSQL()
{
  return _T("[dbo].[Descriptions_Temp]");
}
```

**683. Do Field Exchange.** RFX-Unterstützung. [LDF 2066.06.22.]

⟨ Declare **Descriptions_Temp** functions 677 ⟩ +≡
```
virtual void DoFieldExchange(CFieldExchange *pFX);
```

**684.**

⟨ Define **Descriptions_Temp** functions 678 ⟩ +≡
**void Descriptions_Temp** :: *DoFieldExchange* (**CFieldExchange** ∗*pFX* )
{
*pFX*→*SetFieldType*(**CFieldExchange** :: *outputColumn*);
*RFX_Long*(*pFX* , `_T("[description_id]")`, *m_description_id* );
*RFX_Long*(*pFX* , `_T("[record_id]")`, *m_record_id* );
*RFX_Text*(*pFX* , `_T("[dc_description]")`, *m_dc_description*, 2048);
}

**685. Assert Valid.** [LDF 2066.06.22.]

⟨ Declare **Descriptions_Temp** functions 677 ⟩ +≡
**#ifdef** `_DEBUG`
**virtual void** *AssertValid*( ) **const**;
**#endif**

**686.**

⟨ Define **Descriptions_Temp** functions 678 ⟩ +≡
**#ifdef** `_DEBUG`
**void Descriptions_Temp** :: *AssertValid*( ) **const**
{
**CRecordset** :: *AssertValid*( );
}
**#endif**

**687. Dump.** [LDF 2066.06.22.]

⟨ Declare **Descriptions_Temp** functions 677 ⟩ +≡
**#ifdef** `_DEBUG`
**virtual void** *Dump*(**CDumpContext** &*dc*) **const**;
**#endif**

**688.**

⟨ Define **Descriptions_Temp** functions 678 ⟩ +≡
**#ifdef** `_DEBUG`
**void Descriptions_Temp** :: *Dump*(**CDumpContext** &*dc*) **const**
{
**CRecordset** :: *Dump*(*dc*);
}
**#endif**

**689. Putting Descriptions_Temp together.**

**690.** This is what gets written to `dscrptmp.h`.

⟨ `dscrptmp.h` 690 ⟩ ≡
⟨ Preprocessor macro calls 17 ⟩
⟨ **using** declarations for namespaces 505 ⟩
⟨ Declare **class Descriptions_Temp** 675 ⟩

**691.** This is what gets compiled.

⟨ Include files 20 ⟩
⟨ Define **Descriptions_Temp** functions 678 ⟩

**692. Subjects (`subjects.web`).**

---

**Log**

[LDF 2006.10.05.] Created this file. It contains code formerly in `Subjects.h` and `Subjects.cpp`.

---

⟨ `subjects.web` 692 ⟩ ≡ /* Empty section for use in cross-references. */

This code is cited in sections 7 and 9.

This code is used in section 881.

**693. Preprocessor macro calls.**

⟨ Preprocessor macro calls 17 ⟩ +≡
**#pragma** *once*

**694. using declarations for namespaces.**

⟨ **using** declarations for namespaces 505 ⟩ +≡
**using namespace std**;

**695. Include files.**

⟨ Include files 20 ⟩ +≡
**#include** "stdafx.h"

**696. Subjects class declaration.**

---

**Log**

[LDF 2006.06.30.] Added this declaration. It was generated by Visual Studio.

---

⟨ Declare **class Subjects** 696 ⟩ ≡
```
class Subjects : public CRecordset {
public: long m_subject_id;
  CStringA m_dc_subject;
  ⟨ Declare Subjects functions 698 ⟩
};
```
This code is used in section 711.

**697. Functions.**

**698. Constructor.**

⟨ Declare **Subjects** functions 698 ⟩ ≡
```
Subjects(CDatabase *pDatabase = NULL);
DECLARE_DYNAMIC(Subjects)
```
See also sections 700, 702, 704, 706, and 708.

This code is used in section 696.

**699.**

⟨ Define **Subjects** functions 699 ⟩ ≡
```
IMPLEMENT_DYNAMIC(Subjects, CRecordset)
Subjects::Subjects(CDatabase *pdb)
: CRecordset(pdb) {
  m_subject_id = 0;
  m_dc_subject = "";
  m_nFields = 2;
  m_nDefaultType = dynaset;
}
```
See also sections 701, 703, 705, 707, and 709.

This code is used in section 712.

**700. Get default connect.** [LDF 2006.06.30.]

⟨ Declare **Subjects** functions 698 ⟩ +≡
```
virtual CString GetDefaultConnect();
```

**701.**

⟨ Define **Subjects** functions 699 ⟩ +≡
```
CString Subjects::GetDefaultConnect()
{
  return _T(DATABASE_CONNECTION_STRING);
}
```

**702. Standard-SQL for Recordset.** [LDF 2006.06.30.]

⟨ Declare **Subjects** functions 698 ⟩ +≡
```
virtual CString GetDefaultSQL();
```

**703.**

⟨ Define **Subjects** functions 699 ⟩ +≡
```
CString Subjects::GetDefaultSQL()
{
  return _T("[dbo].[Subjects]");
}
```

**704. Do Field Exchange.** RFX-Unterstützung. [LDF 2006.06.30.]

⟨ Declare **Subjects** functions 698 ⟩ +≡
```
virtual void DoFieldExchange(CFieldExchange *pFX);
```

**705.**

⟨ Define **Subjects** functions 699 ⟩ +≡
```
void Subjects::DoFieldExchange(CFieldExchange *pFX)
{
   pFX->SetFieldType(CFieldExchange::outputColumn);
   RFX_Long(pFX, _T("[subject_id]"), m_subject_id);
   RFX_Text(pFX, _T("[dc_subject]"), m_dc_subject, 1023);
}
```

**706. Assert Valid.** [LDF 2006.06.30.]

⟨ Declare **Subjects** functions 698 ⟩ +≡
```
#ifdef _DEBUG
  virtual void AssertValid() const;
#endif
```

**707.**

⟨ Define **Subjects** functions 699 ⟩ +≡
```
#ifdef _DEBUG
  void Subjects::AssertValid() const
  {
     CRecordset::AssertValid();
  }
#endif
```

**708. Dump.** [LDF 2006.06.30.]

⟨ Declare **Subjects** functions 698 ⟩ +≡
```
#ifdef _DEBUG
  virtual void Dump(CDumpContext &dc) const;
#endif
```

**709.**

⟨ Define **Subjects** functions 699 ⟩ +≡
```
#ifdef _DEBUG
  void Subjects::Dump(CDumpContext &dc) const
  {
     CRecordset::Dump(dc);
  }
#endif
```

**710. Putting Subjects together.**

**711.** This is what gets written to `subjects.h`.

⟨ `subjects.h` 711 ⟩ ≡
  ⟨ Preprocessor macro calls 17 ⟩
  ⟨ **using** declarations for namespaces 505 ⟩
  ⟨ Declare **class Subjects** 696 ⟩

**712.** This is what gets compiled.

⟨ Include files 20 ⟩
⟨ Define **Subjects** functions 699 ⟩

**713. Subjects_Temp (`sbjcttmp.web`).**

---
**Log**

[LDF 2006.10.05.] Created this file. It contains code formerly in `Subjects_Temp.h` and `Subjects_Temp.cpp`.

---

⟨ `sbjcttmp.web` 713 ⟩ ≡ /* Empty section for use in cross-references. */

This code is cited in sections 7 and 9.

This code is used in section 881.

**714. Preprocessor macro calls.**

⟨ Preprocessor macro calls 17 ⟩ +≡
**#pragma** *once*

**715. using declarations for namespaces.**

⟨ **using** declarations for namespaces 505 ⟩ +≡
**using namespace std**;

**716. Include files.**

⟨ Include files 20 ⟩ +≡
**#include** `"stdafx.h"`

**717. Subjects_Temp class declaration.**

---
**Log**

[LDF 2006.06.20.] Added this declaration. It was generated by Visual Studio.

---

⟨ Declare **class Subjects_Temp** 717 ⟩ ≡
**class Subjects_Temp** : **public CRecordset** {
**public**: **long** *m_subject_id*;
**CStringA** *m_dc_subject*;
⟨ Declare **Subjects_Temp** functions 719 ⟩
};

This code is used in section 732.

**718. Functions.**

**719. Constructor.**

⟨ Declare **Subjects_Temp** functions 719 ⟩ ≡
**Subjects_Temp**(**CDatabase** **pDatabase* = `NULL`);
`DECLARE_DYNAMIC`(**Subjects_Temp**)

See also sections 721, 723, 725, 727, and 729.

This code is used in section 717.

**720.**

⟨ Define **Subjects_Temp** functions 720 ⟩ ≡
```
IMPLEMENT_DYNAMIC(Subjects_Temp, CRecordset)
Subjects_Temp::Subjects_Temp(CDatabase *pdb)
: CRecordset(pdb) {
  m_subject_id = 0;
  m_dc_subject = "";
  m_nFields = 2;
  m_nDefaultType = dynaset;
}
```
See also sections 722, 724, 726, 728, and 730.

This code is used in section 733.

**721. Get default connect.** [LDF 2006.06.20.]

⟨ Declare **Subjects_Temp** functions 719 ⟩ +≡
```
virtual CString GetDefaultConnect();
```

**722.**

⟨ Define **Subjects_Temp** functions 720 ⟩ +≡
```
CString Subjects_Temp::GetDefaultConnect()
{
  return _T(DATABASE_CONNECTION_STRING);
}
```

**723. Standard-SQL for Recordset.** [LDF 2006.06.20.]

⟨ Declare **Subjects_Temp** functions 719 ⟩ +≡
```
virtual CString GetDefaultSQL();
```

**724.**

⟨ Define **Subjects_Temp** functions 720 ⟩ +≡
```
CString Subjects_Temp::GetDefaultSQL()
{
  return _T("[dbo].[Subjects_Temp]");
}
```

**725. Do Field Exchange.** RFX-Unterstützung. [LDF 2006.06.20.]

⟨ Declare **Subjects_Temp** functions 719 ⟩ +≡
```
virtual void DoFieldExchange(CFieldExchange *pFX);
```

**726.**

⟨ Define **Subjects_Temp** functions 720 ⟩ +≡
  **void Subjects_Temp** :: *DoFieldExchange* (**CFieldExchange** ∗*pFX* )
  {
    *pFX*→*SetFieldType* (**CFieldExchange** :: *outputColumn*);
    *RFX_Long* (*pFX* , `_T("[subject_id]")`, *m_subject_id* );
    *RFX_Text* (*pFX* , `_T("[dc_subject]")`, *m_dc_subject*, 1023);
  }

**727. Assert Valid.** [LDF 2006.06.20.]

⟨ Declare **Subjects_Temp** functions 719 ⟩ +≡
**#ifdef** `_DEBUG`
  **virtual void** *AssertValid* ( ) **const**;
**#endif**

**728.**

⟨ Define **Subjects_Temp** functions 720 ⟩ +≡
**#ifdef** `_DEBUG`
  **void Subjects_Temp** :: *AssertValid* ( ) **const**
  {
    **CRecordset** :: *AssertValid* ( );
  }
**#endif**

**729. Dump.** [LDF 2006.06.20.]

⟨ Declare **Subjects_Temp** functions 719 ⟩ +≡
**#ifdef** `_DEBUG`
  **virtual void** *Dump* (**CDumpContext** &*dc*) **const**;
**#endif**

**730.**

⟨ Define **Subjects_Temp** functions 720 ⟩ +≡
**#ifdef** `_DEBUG`
  **void Subjects_Temp** :: *Dump* (**CDumpContext** &*dc*) **const**
  {
    **CRecordset** :: *Dump* (*dc*);
  }
**#endif**

**731. Putting Subjects_Temp together.**

**732.** This is what gets written to `sbjcttmp.h`.

⟨ `sbjcttmp.h` 732 ⟩ ≡
  ⟨ Preprocessor macro calls 17 ⟩
  ⟨ **using** declarations for namespaces 505 ⟩
  ⟨ Declare **class Subjects_Temp** 717 ⟩

**733.** This is what gets compiled.

⟨ Include files 20 ⟩
⟨ Define **Subjects_Temp** functions 720 ⟩

**734. Identifiers_Temp (`identtmp.web`).**

**Log**

[LDF 2006.10.05.] Created this file. It contains code formerly in `Identifiers_Temp.h` and `Identifiers_Temp.cpp`.

---

⟨ `identtmp.web` 734 ⟩ ≡ /* Empty section for use in cross-references. */

This code is cited in sections 7 and 9.

This code is used in section 881.

**735. Preprocessor macro calls.**

⟨ Preprocessor macro calls 17 ⟩ +≡
**#pragma** *once*

**736. using declarations for namespaces.**

⟨ **using** declarations for namespaces 505 ⟩ +≡
**using namespace std**;

**737. Include files.**

⟨ Include files 20 ⟩ +≡
**#include** "stdafx.h"

**738. Identifiers_Temp class declaration.**

**Log**

[LDF 2006.06.22.] Added this declaration. It was generated by Visual Studio.

---

⟨ Declare **class Identifiers_Temp** 738 ⟩ ≡
```
class Identifiers_Temp : public CRecordset {
public: long m_identifier_id;
  CStringA m_dc_identifier;
  ⟨ Declare Identifiers_Temp functions 740 ⟩
};
```
This code is used in section 753.

**739. Functions.**

**740. Constructor.**

⟨ Declare **Identifiers_Temp** functions 740 ⟩ ≡
```
Identifiers_Temp(CDatabase *pDatabase = NULL);
DECLARE_DYNAMIC(Identifiers_Temp)
```
See also sections 742, 744, 746, 748, and 750.

This code is used in section 738.

**741.**

⟨ Define **Identifiers_Temp** functions 741 ⟩ ≡

```
IMPLEMENT_DYNAMIC(Identifiers_Temp, CRecordset)
Identifiers_Temp::Identifiers_Temp(CDatabase *pdb)
: CRecordset(pdb) {
  m_identifier_id = 0;
  m_dc_identifier = "";
  m_nFields = 2;
  m_nDefaultType = dynaset;
}
```

See also sections 743, 745, 747, 749, and 751.

This code is used in section 754.

**742. Get default connect.** [LDF 2006.06.22.]

⟨ Declare **Identifiers_Temp** functions 740 ⟩ +≡

```
virtual CString GetDefaultConnect( );
```

**743.**

⟨ Define **Identifiers_Temp** functions 741 ⟩ +≡

```
CString Identifiers_Temp::GetDefaultConnect( )
{
  return _T(DATABASE_CONNECTION_STRING);
}
```

**744. Standard-SQL for Recordset.** [LDF 2006.06.22.]

⟨ Declare **Identifiers_Temp** functions 740 ⟩ +≡

```
virtual CString GetDefaultSQL( );
```

**745.**

⟨ Define **Identifiers_Temp** functions 741 ⟩ +≡

```
CString Identifiers_Temp::GetDefaultSQL( )
{
  return _T("[dbo].[Identifiers_Temp]");
}
```

**746. Do Field Exchange.** RFX-Unterstützung. [LDF 2006.06.22.]

⟨ Declare **Identifiers_Temp** functions 740 ⟩ +≡

```
virtual void DoFieldExchange(CFieldExchange *pFX);
```

**747.**

⟨ Define **Identifiers_Temp** functions 741 ⟩ +≡
```
void Identifiers_Temp::DoFieldExchange(CFieldExchange *pFX)
{
  pFX->SetFieldType(CFieldExchange::outputColumn);
  RFX_Long(pFX, _T("[identifier_id]"), m_identifier_id);
  RFX_Text(pFX, _T("[dc_identifier]"), m_dc_identifier, 1024);
}
```

**748. Assert Valid.** [LDF 2006.06.22.]

⟨ Declare **Identifiers_Temp** functions 740 ⟩ +≡
```
#ifdef _DEBUG
  virtual void AssertValid() const;
#endif
```

**749.**

⟨ Define **Identifiers_Temp** functions 741 ⟩ +≡
```
#ifdef _DEBUG
  void Identifiers_Temp::AssertValid() const
  {
    CRecordset::AssertValid();
  }
#endif
```

**750. Dump.** [LDF 2006.06.22.]

⟨ Declare **Identifiers_Temp** functions 740 ⟩ +≡
```
#ifdef _DEBUG
  virtual void Dump(CDumpContext &dc) const;
#endif
```

**751.**

⟨ Define **Identifiers_Temp** functions 741 ⟩ +≡
```
#ifdef _DEBUG
  void Identifiers_Temp::Dump(CDumpContext &dc) const
  {
    CRecordset::Dump(dc);
  }
#endif
```

**752. Putting Identifiers_Temp together.**

**753.** This is what gets written to `identtmp.h`.

⟨ `identtmp.h` 753 ⟩ ≡
  ⟨ Preprocessor macro calls 17 ⟩
  ⟨ **using** declarations for namespaces 505 ⟩
  ⟨ Declare **class Identifiers_Temp** 738 ⟩

**754.** This is what gets compiled.

⟨ Include files 20 ⟩
⟨ Define **Identifiers_Temp** functions 741 ⟩

**755. Languages_Temp (`langtmp.web`).**

---
**Log**

[LDF 2006.10.05.] Created this file. It contains code formerly in `Languages_Temp.h` and `Languages_Temp.cpp`.

---

⟨ `langtmp.web` 755 ⟩ ≡ /* Empty section for use in cross-references. */

This code is cited in sections 7 and 9.

This code is used in section 881.

**756. Preprocessor macro calls.**

⟨ Preprocessor macro calls 17 ⟩ +≡
**#pragma** *once*

**757. using declarations for namespaces.**

⟨ **using** declarations for namespaces 505 ⟩ +≡
**using namespace std**;

**758. Include files.**

⟨ Include files 20 ⟩ +≡
**#include** "stdafx.h"

**759. Languages_Temp class declaration.**

---
**Log**

[LDF 2006.06.22.] Added this declaration. It was generated by Visual Studio.

---

⟨ Declare **class Languages_Temp** 759 ⟩ ≡
```
class Languages_Temp : public CRecordset {
public: long m_language_id;
  CStringA m_dc_language;
  ⟨ Declare Languages_Temp functions 761 ⟩
};
```

This code is used in section 774.

**760. Functions.**

**761. Constructor.**

⟨ Declare **Languages_Temp** functions 761 ⟩ ≡
```
Languages_Temp(CDatabase *pDatabase = NULL);
DECLARE_DYNAMIC(Languages_Temp)
```

See also sections 763, 765, 767, 769, and 771.

This code is used in section 759.

**762.**

⟨ Define **Languages_Temp** functions 762 ⟩ ≡

```
IMPLEMENT_DYNAMIC(Languages_Temp, CRecordset)
Languages_Temp::Languages_Temp(CDatabase *pdb)
: CRecordset(pdb) {
   m_language_id = 0;
   m_dc_language = "";
   m_nFields = 2;
   m_nDefaultType = dynaset;
}
```

See also sections 764, 766, 768, 770, and 772.

This code is used in section 775.

**763. Get default connect.** [LDF 2006.06.22.]

⟨ Declare **Languages_Temp** functions 761 ⟩ +≡

```
virtual CString GetDefaultConnect();
```

**764.**

⟨ Define **Languages_Temp** functions 762 ⟩ +≡

```
CString Languages_Temp::GetDefaultConnect()
{
   return _T(DATABASE_CONNECTION_STRING);
}
```

**765. Standard-SQL for Recordset.** [LDF 2006.06.22.]

⟨ Declare **Languages_Temp** functions 761 ⟩ +≡

```
virtual CString GetDefaultSQL();
```

**766.**

⟨ Define **Languages_Temp** functions 762 ⟩ +≡

```
CString Languages_Temp::GetDefaultSQL()
{
   return _T("[dbo].[Languages_Temp]");
}
```

**767. Do Field Exchange.** RFX-Unterstützung. [LDF 2006.06.22.]

⟨ Declare **Languages_Temp** functions 761 ⟩ +≡

```
virtual void DoFieldExchange(CFieldExchange *pFX);
```

**768.**

⟨ Define **Languages_Temp** functions 762 ⟩ +≡
**void Languages_Temp** :: *DoFieldExchange*(**CFieldExchange** **pFX*)
{
*pFX*→*SetFieldType*(**CFieldExchange** :: *outputColumn*);
*RFX_Long*(*pFX*, `_T("[language_id]")`, *m_language_id*);
*RFX_Text*(*pFX*, `_T("[dc_language]")`, *m_dc_language*, 1023);
}

**769. Assert Valid.** [LDF 2006.06.22.]

⟨ Declare **Languages_Temp** functions 761 ⟩ +≡
**#ifdef** `_DEBUG`
**virtual void** *AssertValid*( ) **const**;
**#endif**

**770.**

⟨ Define **Languages_Temp** functions 762 ⟩ +≡
**#ifdef** `_DEBUG`
**void Languages_Temp** :: *AssertValid*( ) **const**
{
**CRecordset** :: *AssertValid*( );
}
**#endif**

**771. Dump.** [LDF 2006.06.22.]

⟨ Declare **Languages_Temp** functions 761 ⟩ +≡
**#ifdef** `_DEBUG`
**virtual void** *Dump*(**CDumpContext** &*dc*) **const**;
**#endif**

**772.**

⟨ Define **Languages_Temp** functions 762 ⟩ +≡
**#ifdef** `_DEBUG`
**void Languages_Temp** :: *Dump*(**CDumpContext** &*dc*) **const**
{
**CRecordset** :: *Dump*(*dc*);
}
**#endif**

**773. Putting Languages_Temp together.**

**774.** This is what gets written to `langtmp.h`.

⟨ `langtmp.h` 774 ⟩ ≡
⟨ Preprocessor macro calls 17 ⟩
⟨ **using** declarations for namespaces 505 ⟩
⟨ Declare **class Languages_Temp** 759 ⟩

**775.** This is what gets compiled.

⟨ Include files 20 ⟩
⟨ Define **Languages_Temp** functions 762 ⟩

**776. Publishers_Temp (`pblshtmp.web`).**

**Log**

---

[LDF 2006.10.05.] Created this file. It contains code formerly in `Publishers_Temp.h` and `Publishers_Temp.cpp`.

---

⟨ `pblshtmp.web` 776 ⟩ ≡ /* Empty section for use in cross-references. */

This code is cited in sections 7 and 9.

This code is used in section 881.

**777. Preprocessor macro calls.**

⟨ Preprocessor macro calls 17 ⟩ +≡
#**pragma** *once*

**778. using declarations for namespaces.**

⟨ **using** declarations for namespaces 505 ⟩ +≡
**using namespace std**;

**779. Include files.**

⟨ Include files 20 ⟩ +≡
#**include** `"stdafx.h"`

**780. Publishers_Temp class declaration.**

**Log**

---

[LDF 2006.06.22.] Added this declaration. It was generated by Visual Studio.

---

⟨ Declare **class Publishers_Temp** 780 ⟩ ≡
**class Publishers_Temp** : **public CRecordset** {
**public**: **long** *m_publisher_id*;
**CStringA** *m_dc_publisher*;
**long** *m_person_id*;
**long** *m_institution_id*;
**long** *m_company_id*;
⟨ Declare **Publishers_Temp** functions 782 ⟩
};

This code is used in section 795.

**781. Functions.**

**782. Constructor.**

⟨ Declare **Publishers_Temp** functions 782 ⟩ ≡
**Publishers_Temp**(**CDatabase** **pDatabase* = `NULL`);
`DECLARE_DYNAMIC`(**Publishers_Temp**)

See also sections 784, 786, 788, 790, and 792.

This code is used in section 780.

**783.**

⟨ Define **Publishers_Temp** functions 783 ⟩ ≡
```
IMPLEMENT_DYNAMIC(Publishers_Temp, CRecordset)
Publishers_Temp::Publishers_Temp(CDatabase *pdb)
: CRecordset(pdb) {
  m_publisher_id = 0;
  m_dc_publisher = "";
  m_person_id = 0;
  m_institution_id = 0;
  m_company_id = 0;
  m_nFields = 5;
  m_nDefaultType = dynaset;
}
```
See also sections 785, 787, 789, 791, and 793.

This code is used in section 796.

**784. Get default connect.** [LDF 2006.06.22.]

⟨ Declare **Publishers_Temp** functions 782 ⟩ +≡
```
virtual CString GetDefaultConnect( );
```

**785.**

⟨ Define **Publishers_Temp** functions 783 ⟩ +≡
```
CString Publishers_Temp::GetDefaultConnect( )
{
  return _T(DATABASE_CONNECTION_STRING);
}
```

**786. Standard-SQL for Recordset.** [LDF 2006.06.22.]

⟨ Declare **Publishers_Temp** functions 782 ⟩ +≡
```
virtual CString GetDefaultSQL( );
```

**787.**

⟨ Define **Publishers_Temp** functions 783 ⟩ +≡
```
CString Publishers_Temp::GetDefaultSQL( )
{
  return _T("[dbo].[Publishers_Temp]");
}
```

**788. Do Field Exchange.** RFX-Unterstützung. [LDF 2006.06.22.]

⟨ Declare **Publishers_Temp** functions 782 ⟩ +≡
```
virtual void DoFieldExchange(CFieldExchange *pFX);
```

**789.**

⟨ Define **Publishers_Temp** functions 783 ⟩ +≡

```
void Publishers_Temp::DoFieldExchange(CFieldExchange *pFX)
{
  pFX->SetFieldType(CFieldExchange::outputColumn);
  RFX_Long(pFX, _T("[publisher_id]"), m_publisher_id);
  RFX_Text(pFX, _T("[dc_publisher]"), m_dc_publisher, 1023);
  RFX_Long(pFX, _T("[person_id]"), m_person_id);
  RFX_Long(pFX, _T("[institution_id]"), m_institution_id);
  RFX_Long(pFX, _T("[company_id]"), m_company_id);
}
```

**790. Assert Valid.** [LDF 2006.06.22.]

⟨ Declare **Publishers_Temp** functions 782 ⟩ +≡

```
#ifdef _DEBUG
  virtual void AssertValid() const;
#endif
```

**791.**

⟨ Define **Publishers_Temp** functions 783 ⟩ +≡

```
#ifdef _DEBUG
  void Publishers_Temp::AssertValid() const
  {
    CRecordset::AssertValid();
  }
#endif
```

**792. Dump.** [LDF 2006.06.22.]

⟨ Declare **Publishers_Temp** functions 782 ⟩ +≡

```
#ifdef _DEBUG
  virtual void Dump(CDumpContext &dc) const;
#endif
```

**793.**

⟨ Define **Publishers_Temp** functions 783 ⟩ +≡

```
#ifdef _DEBUG
  void Publishers_Temp::Dump(CDumpContext &dc) const
  {
    CRecordset::Dump(dc);
  }
#endif
```

**794. Putting Publishers_Temp together.**

**795.** This is what gets written to `pblshtmp.h`.

⟨ `pblshtmp.h` 795 ⟩ ≡

  ⟨ Preprocessor macro calls 17 ⟩
  ⟨ **using** declarations for namespaces 505 ⟩
  ⟨ Declare **class Publishers_Temp** 780 ⟩

**796.** This is what gets compiled.

⟨ Include files 20 ⟩
⟨ Define **Publishers_Temp** functions 783 ⟩

**797. Rights_Temp (`rghtstmp.web`).**

**Log**

[LDF 2006.10.05.] Created this file. It contains code formerly in `Rights_Temp.h` and `Rights_Temp.cpp`.

⟨ `rghtstmp.web` 797 ⟩ ≡ /∗ Empty section for use in cross-references. ∗/

This code is cited in sections 7 and 9.

This code is used in section 881.

**798. Preprocessor macro calls.**

⟨ Preprocessor macro calls 17 ⟩ +≡
**#pragma** *once*

**799. using declarations for namespaces.**

⟨ **using** declarations for namespaces 505 ⟩ +≡
**using namespace std**;

**800. Include files.**

⟨ Include files 20 ⟩ +≡
**#include** `"stdafx.h"`

**801. Rights_Temp class declaration.**

**Log**

[LDF 2006.06.22.] Added this declaration. It was generated by Visual Studio.

⟨ Declare **class Rights_Temp** 801 ⟩ ≡
**class Rights_Temp** : **public CRecordset** {
**public**: **long** *m_rights_id*;
**CStringA** *m_dc_rights*;
⟨ Declare **Rights_Temp** functions 803 ⟩
};

This code is used in section 816.

**802. Functions.**

**803. Constructor.**

⟨ Declare **Rights_Temp** functions 803 ⟩ ≡
**Rights_Temp**(**CDatabase** ∗*pDatabase* = `NULL`);
`DECLARE_DYNAMIC`(**Rights_Temp**)

See also sections 805, 807, 809, 811, and 813.

This code is used in section 801.

**804.**

⟨ Define **Rights_Temp** functions 804 ⟩ ≡

```
IMPLEMENT_DYNAMIC(Rights_Temp, CRecordset)
Rights_Temp::Rights_Temp(CDatabase *pdb)
: CRecordset(pdb) {
  m_rights_id = 0;
  m_dc_rights = "";
  m_nFields = 2;
  m_nDefaultType = dynaset;
}
```

See also sections 806, 808, 810, 812, and 814.
This code is used in section 817.

**805. Get default connect.** [LDF 2006.06.22.]

⟨ Declare **Rights_Temp** functions 803 ⟩ +≡

```
virtual CString GetDefaultConnect( );
```

**806.**

⟨ Define **Rights_Temp** functions 804 ⟩ +≡

```
CString Rights_Temp::GetDefaultConnect( )
{
  return _T(DATABASE_CONNECTION_STRING);
}
```

**807. Standard-SQL for Recordset.** [LDF 2006.06.22.]

⟨ Declare **Rights_Temp** functions 803 ⟩ +≡

```
virtual CString GetDefaultSQL( );
```

**808.**

⟨ Define **Rights_Temp** functions 804 ⟩ +≡

```
CString Rights_Temp::GetDefaultSQL( )
{
  return _T("[dbo].[Rights_Temp]");
}
```

**809. Do Field Exchange.** RFX-Unterstützung. [LDF 2006.06.22.]

⟨ Declare **Rights_Temp** functions 803 ⟩ +≡

```
virtual void DoFieldExchange(CFieldExchange *pFX);
```

**810.**

⟨ Define **Rights_Temp** functions 804 ⟩ +≡

```
void Rights_Temp::DoFieldExchange(CFieldExchange *pFX)
{
   pFX->SetFieldType(CFieldExchange::outputColumn);
   RFX_Long(pFX, _T("[rights_id]"), m_rights_id);
   RFX_Text(pFX, _T("[dc_rights]"), m_dc_rights, 1024);
}
```

**811. Assert Valid.** [LDF 2006.06.22.]

⟨ Declare **Rights_Temp** functions 803 ⟩ +≡

```
#ifdef _DEBUG
   virtual void AssertValid() const;
#endif
```

**812.**

⟨ Define **Rights_Temp** functions 804 ⟩ +≡

```
#ifdef _DEBUG
   void Rights_Temp::AssertValid() const
   {
      CRecordset::AssertValid();
   }
#endif
```

**813. Dump.** [LDF 2006.06.22.]

⟨ Declare **Rights_Temp** functions 803 ⟩ +≡

```
#ifdef _DEBUG
   virtual void Dump(CDumpContext &dc) const;
#endif
```

**814.**

⟨ Define **Rights_Temp** functions 804 ⟩ +≡

```
#ifdef _DEBUG
   void Rights_Temp::Dump(CDumpContext &dc) const
   {
      CRecordset::Dump(dc);
   }
#endif
```

**815. Putting Rights_Temp together.**

**816.** This is what gets written to `rghtstmp.h`.

⟨ `rghtstmp.h` 816 ⟩ ≡
  ⟨ Preprocessor macro calls 17 ⟩
  ⟨ **using** declarations for namespaces 505 ⟩
  ⟨ Declare **class Rights_Temp** 801 ⟩

**817.** This is what gets compiled.

⟨ Include files 20 ⟩
⟨ Define **Rights_Temp** functions 804 ⟩

**818. Types_Temp (`typestmp.web`).**

---

**Log**

[LDF 2006.10.05.] Created this file. It contains code formerly in `Types_Temp.h` and `Types_Temp.cpp`.

---

⟨ `typestmp.web` 818 ⟩ ≡ /* Empty section for use in cross-references. */

This code is cited in sections 7 and 9.

This code is used in section 881.

**819. Preprocessor macro calls.**

⟨ Preprocessor macro calls 17 ⟩ +≡
**#pragma** *once*

**820. using declarations for namespaces.**

⟨ **using** declarations for namespaces 505 ⟩ +≡
**using namespace std**;

**821. Include files.**

⟨ Include files 20 ⟩ +≡
**#include** `"stdafx.h"`

**822. Types_Temp class declaration.**

---

**Log**

[LDF 2006.06.22.] Added this declaration. It was generated by Visual Studio.

---

⟨ Declare **class Types_Temp** 822 ⟩ ≡

```
class Types_Temp : public CRecordset {
public: long m_type_id;
  CStringA m_dc_type;
  ⟨ Declare Types_Temp functions 824 ⟩
};
```

This code is used in section 837.

**823. Functions.**

**824. Constructor.**

⟨ Declare **Types_Temp** functions 824 ⟩ ≡

```
Types_Temp(CDatabase *pDatabase = NULL);
DECLARE_DYNAMIC(Types_Temp)
```

See also sections 826, 828, 830, 832, and 834.

This code is used in section 822.

**825.**

⟨ Define **Types_Temp** functions 825 ⟩ ≡

```
IMPLEMENT_DYNAMIC(Types_Temp, CRecordset)
Types_Temp::Types_Temp(CDatabase *pdb)
: CRecordset(pdb) {
  m_type_id = 0;
  m_dc_type = "";
  m_nFields = 2;
  m_nDefaultType = dynaset;
}
```

See also sections 827, 829, 831, 833, and 835.

This code is used in section 838.

**826. Get default connect.** [LDF 2006.06.22.]

⟨ Declare **Types_Temp** functions 824 ⟩ +≡

```
virtual CString GetDefaultConnect( );
```

**827.**

⟨ Define **Types_Temp** functions 825 ⟩ +≡

```
CString Types_Temp::GetDefaultConnect( )
{
  return _T(DATABASE_CONNECTION_STRING);
}
```

**828. Standard-SQL for Recordset.** [LDF 2006.06.22.]

⟨ Declare **Types_Temp** functions 824 ⟩ +≡

```
virtual CString GetDefaultSQL( );
```

**829.**

⟨ Define **Types_Temp** functions 825 ⟩ +≡

```
CString Types_Temp::GetDefaultSQL( )
{
  return _T("[dbo].[Types_Temp]");
}
```

**830. Do Field Exchange.** RFX-Unterstützung. [LDF 2006.06.22.]

⟨ Declare **Types_Temp** functions 824 ⟩ +≡

```
virtual void DoFieldExchange(CFieldExchange *pFX);
```

**831.**

⟨ Define **Types_Temp** functions 825 ⟩ +≡
```
void Types_Temp::DoFieldExchange(CFieldExchange *pFX)
{
   pFX->SetFieldType(CFieldExchange::outputColumn);
   RFX_Long(pFX, _T("[type_id]"), m_type_id);
   RFX_Text(pFX, _T("[dc_type]"), m_dc_type, 1023);
}
```

**832. Assert Valid.** [LDF 2006.06.22.]

⟨ Declare **Types_Temp** functions 824 ⟩ +≡
```
#ifdef _DEBUG
   virtual void AssertValid() const;
#endif
```

**833.**

⟨ Define **Types_Temp** functions 825 ⟩ +≡
```
#ifdef _DEBUG
   void Types_Temp::AssertValid() const
   {
      CRecordset::AssertValid();
   }
#endif
```

**834. Dump.** [LDF 2006.06.22.]

⟨ Declare **Types_Temp** functions 824 ⟩ +≡
```
#ifdef _DEBUG
   virtual void Dump(CDumpContext &dc) const;
#endif
```

**835.**

⟨ Define **Types_Temp** functions 825 ⟩ +≡
```
#ifdef _DEBUG
   void Types_Temp::Dump(CDumpContext &dc) const
   {
      CRecordset::Dump(dc);
   }
#endif
```

**836. Putting Types_Temp together.**

**837.** This is what gets written to `typestmp.h`.

⟨ `typestmp.h` 837 ⟩ ≡
   ⟨ Preprocessor macro calls 17 ⟩
   ⟨ **using** declarations for namespaces 505 ⟩
   ⟨ Declare **class Types_Temp** 822 ⟩

**838.** This is what gets compiled.

⟨ Include files 20 ⟩
⟨ Define **Types_Temp** functions 825 ⟩

**839. Temp_IDs (`tempids.web`).**

**Log**

[LDF 2006.10.05.] Created this file. It contains code formerly in `Temp_IDs.h` and `Temp_IDs.cpp`.

---

⟨ `tempids.web` 839 ⟩ ≡ /∗ Empty section for use in cross-references. ∗/

This code is cited in sections 7 and 9.

This code is used in section 881.

**840. Preprocessor macro calls.**

⟨ Preprocessor macro calls 17 ⟩ +≡
**#pragma** *once*

**841. using declarations for namespaces.**

⟨ **using** declarations for namespaces 505 ⟩ +≡
**using namespace std**;

**842. Include files.**

⟨ Include files 20 ⟩ +≡
**#include** "`stdafx.h`"

**843. Temp_IDs class declaration.**

**Log**

[LDF 2006.06.20.] Added this declaration. It was generated by Visual Studio.

---

⟨ Declare **class Temp_IDs** 843 ⟩ ≡

```
class Temp_IDs : public CRecordset {
public: long m_temp_id;
  ⟨ Declare Temp_IDs functions 845 ⟩
};
```

This code is used in section 858.

**844. Functions.**

**845. Constructor.**

⟨ Declare **Temp_IDs** functions 845 ⟩ ≡

```
Temp_IDs(CDatabase *pDatabase = NULL);
DECLARE_DYNAMIC(Temp_IDs)
```

See also sections 847, 849, 851, 853, and 855.

This code is used in section 843.

**846.**

⟨ Define **Temp_IDs** functions 846 ⟩ ≡
```
IMPLEMENT_DYNAMIC(Temp_IDs, CRecordset)
Temp_IDs::Temp_IDs(CDatabase *pdb)
: CRecordset(pdb) {
   m_temp_id = 0;
   m_nFields = 1;
   m_nDefaultType = dynaset;
}
```
See also sections 848, 850, 852, 854, and 856.

This code is used in section 859.

**847. Get default connect.** [LDF 2006.06.20.]

⟨ Declare **Temp_IDs** functions 845 ⟩ +≡
```
virtual CString GetDefaultConnect();
```

**848.**

⟨ Define **Temp_IDs** functions 846 ⟩ +≡
```
CString Temp_IDs::GetDefaultConnect()
{
   return _T(DATABASE_CONNECTION_STRING);
}
```

**849. Standard-SQL for Recordset.** [LDF 2006.06.20.]

⟨ Declare **Temp_IDs** functions 845 ⟩ +≡
```
virtual CString GetDefaultSQL();
```

**850.**

⟨ Define **Temp_IDs** functions 846 ⟩ +≡
```
CString Temp_IDs::GetDefaultSQL()
{
   return _T("[dbo].[Temp_IDs]");
}
```

**851. Do Field Exchange.** RFX-Unterstützung. [LDF 2006.06.20.]

⟨ Declare **Temp_IDs** functions 845 ⟩ +≡
```
virtual void DoFieldExchange(CFieldExchange *pFX);
```

**852.**

⟨ Define **Temp_IDs** functions 846 ⟩ +≡
```
void Temp_IDs::DoFieldExchange(CFieldExchange *pFX)
{
  pFX->SetFieldType(CFieldExchange::outputColumn);
  RFX_Long(pFX, _T("[temp_id]"), m_temp_id);
}
```

**853. Assert Valid.** [LDF 2006.06.20.]

⟨ Declare **Temp_IDs** functions 845 ⟩ +≡
```
#ifdef _DEBUG
  virtual void AssertValid() const;
#endif
```

**854.**

⟨ Define **Temp_IDs** functions 846 ⟩ +≡
```
#ifdef _DEBUG
  void Temp_IDs::AssertValid() const
  {
    CRecordset::AssertValid();
  }
#endif
```

**855. Dump.** [LDF 2006.06.20.]

⟨ Declare **Temp_IDs** functions 845 ⟩ +≡
```
#ifdef _DEBUG
  virtual void Dump(CDumpContext &dc) const;
#endif
```

**856.**

⟨ Define **Temp_IDs** functions 846 ⟩ +≡
```
#ifdef _DEBUG
  void Temp_IDs::Dump(CDumpContext &dc) const
  {
    CRecordset::Dump(dc);
  }
#endif
```

**857. Putting Temp_IDs together.**

**858.** This is what gets written to `tempids.h`.

⟨ `tempids.h` 858 ⟩ ≡
  ⟨ Preprocessor macro calls 17 ⟩
  ⟨ **using** declarations for namespaces 505 ⟩
  ⟨ Declare **class Temp_IDs** 843 ⟩

**859.** This is what gets compiled.

⟨ Include files 20 ⟩
⟨ Define **Temp_IDs** functions 846 ⟩

**860. Temp_IDs_1 (`tempids1.web`).**

**Log**

[LDF 2006.10.05.] Created this file. It contains code formerly in `Temp_IDs_1.h` and `Temp_IDs_1.cpp`.

---

⟨ `tempids1.web` 860 ⟩ ≡ /* Empty section for use in cross-references. */

This code is cited in sections 7 and 9.

This code is used in section 881.

**861. Preprocessor macro calls.**

⟨ Preprocessor macro calls 17 ⟩ +≡
**#pragma** *once*

**862. using declarations for namespaces.**

⟨ **using** declarations for namespaces 505 ⟩ +≡
**using namespace std**;

**863. Include files.**

⟨ Include files 20 ⟩ +≡
**#include** `"stdafx.h"`

**864. Temp_IDs_1 class declaration.**

**Log**

[LDF 2006.06.30.] Added this declaration. It was generated by Visual Studio.

---

⟨ Declare **class Temp_IDs_1** 864 ⟩ ≡
**class Temp_IDs_1** : **public CRecordset** {
**public**: **long** *m_temp_id*;
⟨ Declare **Temp_IDs_1** functions 866 ⟩
};

This code is used in section 879.

**865. Functions.**

**866. Constructor.**

⟨ Declare **Temp_IDs_1** functions 866 ⟩ ≡
**Temp_IDs_1**(**CDatabase** **pDatabase* = `NULL`);
`DECLARE_DYNAMIC`(**Temp_IDs_1**)

See also sections 868, 870, 872, 874, and 876.

This code is used in section 864.

**867.**

⟨ Define **Temp_IDs_1** functions 867 ⟩ ≡
```
IMPLEMENT_DYNAMIC(Temp_IDs_1, CRecordset)
Temp_IDs_1::Temp_IDs_1(CDatabase *pdb)
: CRecordset(pdb) {
  m_temp_id = 0;
  m_nFields = 1;
  m_nDefaultType = dynaset;
}
```
See also sections 869, 871, 873, 875, and 877.

This code is used in section 880.

**868. Get default connect.** [LDF 2006.06.30.]

⟨ Declare **Temp_IDs_1** functions 866 ⟩ +≡
```
virtual CString GetDefaultConnect( );
```

**869.**

⟨ Define **Temp_IDs_1** functions 867 ⟩ +≡
```
CString Temp_IDs_1::GetDefaultConnect( )
{
  return _T(DATABASE_CONNECTION_STRING);
}
```

**870. Standard-SQL for Recordset.** [LDF 2006.06.30.]

⟨ Declare **Temp_IDs_1** functions 866 ⟩ +≡
```
virtual CString GetDefaultSQL( );
```

**871.**

⟨ Define **Temp_IDs_1** functions 867 ⟩ +≡
```
CString Temp_IDs_1::GetDefaultSQL( )
{
  return _T("[dbo].[Temp_IDs_1]");
}
```

**872. Do Field Exchange.** RFX-Unterstützung. [LDF 2006.06.30.]

⟨ Declare **Temp_IDs_1** functions 866 ⟩ +≡
```
virtual void DoFieldExchange(CFieldExchange *pFX);
```

**873.**

⟨ Define **Temp_IDs_1** functions 867 ⟩ +≡

```
void Temp_IDs_1::DoFieldExchange(CFieldExchange *pFX)
{
  pFX->SetFieldType(CFieldExchange::outputColumn);
  RFX_Long(pFX, _T("[temp_id]"), m_temp_id);
}
```

**874. Assert Valid.** [LDF 2006.06.30.]

⟨ Declare **Temp_IDs_1** functions 866 ⟩ +≡

```
#ifdef _DEBUG
  virtual void AssertValid() const;
#endif
```

**875.**

⟨ Define **Temp_IDs_1** functions 867 ⟩ +≡

```
#ifdef _DEBUG
  void Temp_IDs_1::AssertValid() const
  {
    CRecordset::AssertValid();
  }
#endif
```

**876. Dump.** [LDF 2006.06.30.]

⟨ Declare **Temp_IDs_1** functions 866 ⟩ +≡

```
#ifdef _DEBUG
  virtual void Dump(CDumpContext &dc) const;
#endif
```

**877.**

⟨ Define **Temp_IDs_1** functions 867 ⟩ +≡

```
#ifdef _DEBUG
  void Temp_IDs_1::Dump(CDumpContext &dc) const
  {
    CRecordset::Dump(dc);
  }
#endif
```

**878. Putting Temp_IDs_1 together.**

**879.** This is what gets written to `tempids1.h`.

⟨ `tempids1.h` 879 ⟩ ≡
  ⟨ Preprocessor macro calls 17 ⟩
  ⟨ **using** declarations for namespaces 505 ⟩
  ⟨ Declare **class Temp_IDs_1** 864 ⟩

**880.** This is what gets compiled.

  ⟨ Include files 20 ⟩
  ⟨ Define **Temp_IDs_1** functions 867 ⟩

**881.** This section just prevents CWEAVE from issuing warning messages, which would happen if these empty sections weren't included somewhere. [LDF 2006.09.27.]

⟨ Dummy sections 15 ⟩ +≡
⟨ Special character formatting 11 ⟩
⟨ `atest.web` 50 ⟩
⟨ `atestdoc.web` 75 ⟩
⟨ `atstview.web` 96 ⟩
⟨ `cntrbtmp.web` 608 ⟩
⟨ `cntrbtrs.web` 587 ⟩
⟨ `creators.web` 545 ⟩
⟨ `crtrstmp.web` 566 ⟩
⟨ `dialog1a.web` 130 ⟩
⟨ `dialog2a.web` 208 ⟩
⟨ `dscrptmp.web` 671 ⟩
⟨ `glblfncs.web` 294 ⟩
⟨ `identtmp.web` 734 ⟩
⟨ `langtmp.web` 755 ⟩
⟨ `mainfrm.web` 28 ⟩
⟨ `mtdtsrc.web` 312 ⟩
⟨ `pblshtmp.web` 776 ⟩
⟨ `rcrdstmp.web` 524 ⟩
⟨ `records.web` 503 ⟩
⟨ `resource.web` 12 ⟩
⟨ `rghtstmp.web` 797 ⟩
⟨ `sbjcttmp.web` 713 ⟩
⟨ `selector.web` 379 ⟩
⟨ `spchrtbl.web` 10 ⟩
⟨ `stdafx.web` 16 ⟩
⟨ `subjects.web` 692 ⟩
⟨ `tempids.web` 839 ⟩
⟨ `tempids1.web` 860 ⟩
⟨ `titles.web` 629 ⟩
⟨ `ttlstmp.web` 650 ⟩
⟨ `typestmp.web` 818 ⟩

**882. GNU General Public License.** [LDF 2006.10.23.]

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

## GNU GENERAL PUBLIC LICENSE
## TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will

automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our

decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

> ⟨one line to give the program's name and a brief idea of what it does.⟩
> Copyright (C) ⟨year⟩ ⟨name of author⟩
>
> This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.
>
> This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.
>
> You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

> Gnomovision version 69, Copyright © year name of author
> Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type 'show w'.
> This is free software, and you are welcome to redistribute it under certain conditions; type 'show c' for details.

The hypothetical commands 'show w' and 'show c' should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than 'show w' and 'show c'; they could even be mouse-clicks or menu items–whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program 'Gnomovision' (which makes passes at compilers) written by James Hacker.

⟨signature of Ty Coon⟩, 1 April 1989 Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.

⟨ GNU General Public License 882 ⟩ ≡ /* This section contains no C++ code. */

This code is cited in section 2.

This code is used in section 884.

**883. GNU Free Documentation License.** [LDF 2006.10.23.]

GNU Free Documentation License
Version 1.2, November 2002



0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

## 2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

## 3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

## 4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission. B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement. C. State on the Title page the name of the publisher of the Modified Version, as the publisher. D. Preserve all the copyright notices of the Document. E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices. F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below. G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice. H. Include an unaltered copy of this License. I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence. J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission. K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein. L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles. M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version. N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section. O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties–for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

## 5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

## 6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

## 7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

## 8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

## 9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

## 10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See `http://www.gnu.org/copyleft/`.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

> Copyright © YEAR YOUR NAME. Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the "with...Texts." line with this:

> with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

⟨ GNU Free Documentation License 883 ⟩ ≡ /* This section contains no C++ code. */

This code is cited in section 2.

This code is used in section 884.

**884.** Include sections without C++ code. These sections contain no code, or only commented-out code. However, they must be included somewhere, or CWEAVE will issue warnings. [LDF 2006.09.27.]

⟨ Dummy sections 15 ⟩
⟨ GNU General Public License 882 ⟩
⟨ GNU Free Documentation License 883 ⟩

**885. Index.**

# IWF Metadata Harvester I
# ATest: The Program
# Version 1.0
by Laurence D. Finston

October 2006