

dbinsert

Übersicht dbinsert.c

dbinsert.c ist ein Makro zum Einfügen von Ditabis ImagePlate-Bildern in die Datenbank. Alle markierten Bilder im Bildspeicher werden in die Datenbank eingefügt. Dabei werden gleichzeitig Metadaten aus dem Feld Bildinformation ausgelesen. Diese werden in entsprechende Felder der Datenbank eingefügt. Das Makro erspart das Umtragen der Daten aus den Bildinformationen.

Da sich die gesuchten Werte und die Felder in der Datenbank bei jedem Nutzer unterscheiden, muss leider Hand an den Quelltext gelegt werden:

Anpassen des Makros

Das Programm hat drei Bereiche, die wahrscheinlich angepasst werden müssen:

Variablendeklarationen:

Änderungen müssen hier nur vorgenommen werden, wenn neue Variablen eingeführt werden. Zeilen 2 – 13:

```
1
2  char zaehler[80];
3  char ditabisheader[2048];
4  char gridbox[10];
5  char exposure[5];
6  char specimen[255];
7  char gridschnitt[20];
8  char probe[20];
9  char plate[20];
10 char datensatzname[100];
11
12 char abfragewert[100];
13 char abfragelabel[100];
14
15
16
17 WORD wImg; //WORD = unsigned short
18
19 char searchstring[2048];
20 int searchstringlen;
21 int charindex = 0;
22
```

Wertsuche:

Der Quelltext enthält mehrere sich wiederholende Abschnitte die die Werte aus der Bilddatei suchen (z.B. Zeilen 85 – 100).

Jeder dieser Abschnitte beginnt mit: `resultstring = ""`; und endet mit den Zeilen:
`variable = resultstring;`

```

}
83
84
85 resultstring = "";
86 searchstring = "EXPOSURE: ";
87 if (startpointer = strstr(ditabisheader, searchstring)) {
88     ....
89     searchstringlen = strlen(searchstring);
90     startpointer = startpointer + searchstringlen;
91     ....
92     while ((*startpointer != '\r') && (charindex <= 2048)) {
93         sprintf (zwischenstring, "%c", *startpointer);
94         strcat (resultstring, zwischenstring);
95         charindex++;
96         *startpointer++;
97     }
98     //dlgShowMessage(resultstring, 3, IDL_USA, "dlgShowMessage Test",
99     //MB_YESNOCANCEL | MB_ICONINFORMATION, 3);
100     exposure = resultstring;
101     ..
102 ..

```

Für jeden gesuchten Wert muss ein solcher Abschnitt vorhanden sein. Mit der Zeile:

```
searchstring = "EXPOSURE: ";
```

wird festgelegt, nach welchem Begriff gesucht wird. In diesem Fall wird nach EXPOSURE: gesucht. Anschließend werden die Zeichen die auf den Suchbegriff folgen in eine Ergebnisvariable eingefügt, bis ein definiertes Stopzeichen erreicht wird.

Die Zeile

```
while ((*startpointer != '\r') && (charindex <= 2048)) {
```

bestimmt dieses Stopzeichen:

'\r' steht für das Zeilenende

' ' steht für ein Leerzeichen

Am Ende der Routine wird Das Ergebnis der zuständigen Variablen übergeben:

```
exposure = resultstring;
}
```

Es können beliebig viele dieser Abschnitte im Programm vorhanden sein. Wichtig ist:

den **Suchbegriff definieren** in: `searchstring = "";`

Das **Stopzeichen definieren** in: `while ((*startpointer != '\r') && (charindex <= 2048)) {`

Die **Variable für das Ergebnis** bestimmen: `variable = resultstring;` Die Variable muss natürlich deklariert sein (siehe oben) und ihr Wert in die Datenbank eingefügt werden (siehe unten).

Abschnitte, die nicht gebraucht werden können mit `/*` und `*/` eingeschlossen werden, so dass sie auskommentiert sind.

Einfügen der Ergebnisse in die Datenbank

Die in den Variablen gespeicherten Ergebnisse werden in die Spalten der Datenbank zugeordnet. Dies geschieht in den Zeilen 256 – 265.

```

253
254 // fill empty user fields of the inserted record:
255
256 dbRet = dbSetFields( hRecordset, FALSE,
257     .           .           "Gridbox", gridbox,
258     .           .           "Exposure", exposure,
259     .           .           "Artname", specimen,
260     .           .           "Grid / Schnitt", gridschnitt,
261     .           .           "Probe", probe,
262     .           .           "Platte", plate,
263     .           .           //"Datensatzname", datensatzname,
264     .           .           //"Abfragefeld", abfragewert,
265     .           .           NULL );
266     .           .           .....
267     .           .           .....

```

Jede Zuordnung geschieht über eine Zeile:

„Spaltenname“, Variable,

Dabei werden die in den Variablen gespeicherten Werte der Datenbankspalte mit dem entsprechenden Namen zugeordnet. Alle hier angegebenen Spaltennamen müssen in der Datenbank tatsächlich vorhanden sein, sonst bricht das Makro mit einem Fehler ab. Einzelne nicht gebrauchte Zeilen können mit // auskommentiert werden.

Zusätzliche Möglichkeiten:

Ergebnisse zusammenfügen:

Die Ergebnisse mehrerer Suchen können in einer Variablen zusammengefügt werden (Zeilen 206 – 216).

```

205
206 //Datensatzname aus Gridbox und Grid / Schnitt zusammenstellen
207     datensatzname = "";
208
209 ☐ if (gridbox) {
210     strcat (datensatzname, gridbox);
211     strcat (datensatzname, " ");
212 }
213
214 ☐ if (gridschnitt) {
215     strcat (datensatzname, gridschnitt);
216 }
217

```

Mit dem Befehl `strcat (variable1, variable2)` wird an den in `variable1` enthaltenen Text der Inhalt von `variable2` angefügt.

Werte Abfragen

Beim Einfügen in die Datenbank können die Werte für einzelne Spalten auch vom Nutzer abgefragt werden (Zeile 61 – 66).

```

59
60 //Abfrage eines Wertes der in Datenbank eingetragen werden soll
61 {
62     abfragewert = "";
63     abfragelabel = "Bitte Wert fuer \"Abfragefeld\" eingeben.";
64     if (!dlgInput(&abfragewert, sizeof(abfragewert), abfragelabel, "Eingabe")) == IDOK)..
65         abfragewert = "";
66 }
67

```

Der Code öffnet beim Ausführen des Makros einen Abfragedialog in den ein Text eingegeben werden kann. Dieser Text wird in die Datenbankspalte geschrieben, die der Variablen `abfragewert` zugeordnet ist (siehe oben). Der im Abfragedialog angezeigte Text "Bitte Wert fuer \"Abfragefeld\" eingeben." kann natürlich ebenso geändert werden, wie der Variablenname.

Bemerkungen

Hinweise zum Einrichten von Makros gibt es auf der Downloadseite von `dbinsert`.

`dbinsert` minimiert alle in Analysis angezeigten Fenster. Das muss so sein, da Analysis die Datenbank beschädigen kann, wenn gleichzeitig mit dem Einfügen in ein anderes Fenster geklickt wird.

Analysis zeigt eingefügte Bilder oft erst an nachdem man einmal auf den Button mit dem durchgestrichenen Fernglas gedrückt hat.

`dbinsert` funktioniert nicht, wenn die in der Zuordnung angegebenen Spaltennamen nicht exakt mit den in der Datenbanktabelle vorhandenen Spaltennamen übereinstimmt.

Ich empfehle dringend, **dbinsert erst einmal mit einer Testdatenbank auszuprobieren**, um zu überprüfen, ob alle Anpassungen wie gewünscht funktionieren und das keine unerwünschten Nebeneffekte auftreten.

Viel Erfolg beim Coden

Björn Quast

bquast@zoosyst-berlin.de