LOGIC IT-SERVICES
www.logic.ch
info@logic.ch

Die Vorteile von RPG Open Access und JSON um Metadaten in der Displayfile DDS zu speichern.

Während RPG Open Access nichts Neues ist, gibt es doch noch einige Verwirrung in der IBM i Community, was es ist und wie es verwendet wird. Zudem wurde in letzter Zeit viel über Open Access Metadaten diskutiert und wie sie genutzt werden sollten.

In diesem Artikel wird erklärt, um was es bei Rational Open Access RPG überhaupt geht, was Metadaten sind, was sie abdecken, in welcher Beziehung Metadaten zu Open Access sind, wie ProfoundUI die Metadaten speichert und welches die Vorteile sind.

Was ist Rational Open Access überhaupt?

Rational Open Access RPG (oder "RPG OA") ist ein Feature der ILE RPG Programmiersprache der IBM i ab Version 6.1 oder höher. Es wird von IBM seit 2010 bereitgestellt und ist mittlerweile kostenlos. RPG OA macht es möglich, einen "Handler" zu einer RPG F-Spezifikation anzuhängen, so dass, wenn RPG Operationen (EXFMT, READ, WRITE, UPDATE, etc) ausgeführt werden, die "Handler" deren Kontrolle übernehmen.

Dem Handler lassen sich jegliche Art von Dateien anhängen, hauptsächlich jedoch Dateidefinitionen für Bildschirme, Daten oder Drucker.

Wie in Beispiel 1 gezeigt, verwendet das Programm den RPG EXFMT Operationscode für eine Bildschirm Ausgabe. Ohne das HANDLER Schlüsselwort generiert der RPG-Compiler eine Routine für das Betriebssystem um den 5250-Bildschirm anzuzeigen. Da nun aber das Schlüsselwort HANDLER hinzugefügt wurde, wird die Kontrolle über die angegebene Routine aufgerufen (Eine Unterprozedur MYPROC, im Serviceprogramm MYHANDLER).

Diese Unterprozedur kann jede Technik des EXFMT Codes verarbeiten. Der Code im Handler kann eine Web-Seite ausgeben, einen Webdienst aufrufen oder auch ein mobiles Gerät steuern.

```
H DFTACTGRP(*NO)
FCUST001D  CF   E             WORKSTN HANDLER('MYLIB/MYHANDLER(MYPROC)')
   .
   .
   exfmt SCREEN1;
```

Abbildung 1: Einfaches Beispiel des Einsatzes von Open Access

Profound Logic hat einen Handler für das Erstellen von Web-oder Mobilen GUI Anzeigedateien entwickelt. Die RPG OA Handler-Schnittstellen mit dem JavaScript-Framework, stellt Rich-Display Files zur Verfügung, die in einem Web-Browser oder einer mobilen Anwendung angezeigt werden.

Das Beispiel zeigt den Handler als Unterprozedur HANDLER im Serviceprogramm PROFOUNDUI.

```
H DFTACTGRP(*NO)
FCUST001D  CF   E             WORKSTN HANDLER('PROFOUNDUI(HANDLER)')
    .
    .
   exfmt SCREEN1;
```

Abbildung 2: Mit dem ProfoundUI Handler

Bei vorhandenen RPG-Programmen ist die F-Spezifikation des Bildschirms die einzige Codezeile, die geändert werden muss. Sobald das HANDLER Schlüsselwort hinzugefügt wurde, werden alle Operationen der Bildschirmdatei durch den Handler gesteuert, der Rest des RPG Code funktioniert hingegen genau wie bisher, obwohl der PROFOUNDUI Handler jetzt eine grafischen Web-Maske für die Benutzer ausgibt.

Das ist alles was RPG OA macht: Mit dem Schlüsselwort Handler wird die herkömmliche I/O-Steuerung durch die Handler-Prozedur ersetzt. Aus der Sicht des Benutzers ist das extrem einfach. Es gibt nichts Neues, ausser dem neuen Schlüsselwort in den F-Bestimmungen der Bildschirmdatei.

Was sind Open Access Metadaten?

Metadaten sind Spezifikationen, die der Code innerhalb des Handlers braucht, um seine Arbeit zu verrichten. Um dies zu verdeutlichen, betrachten wir den DDS Code einer herkömmlichen Bildschirmdatei.

```
A            PPRICE        11Y 2B 10 22
A  30                                 DSPATR(PR)
A N30                                 DSPATR(UL)
A                                     EDTCDE(J)
```

Abbildung 3: DDS Snippet von einer traditionellen Bildschirmdatei

In Abbildung 3 enthält der Datensatz der Bildschirmdatei die Felder *IN30 (als Indikator 30) und PPRICE. Wird dieser Datensatz ausgeführt, zeigt der Bildschirm diese Felder mit den entsprechenden Eigenschaften an. Wie erkennt nun der Bildschirm, ob die numerische Tastatur benutzen soll? Oder mit welcher Maske das Feld PPRICE dargestellt werden soll? Und wie soll er wissen, wie das PR- oder UL-Display-Attribut angewendet werden soll? Er kennt das von den DDS Schlüsselworten natürlich!

In einer traditionellen grünen Bildschirmanzeige sind diese DDS Schlüsselworte die "Metadaten". Also die Spezifikationen, die 5250 Ausgabe-Routinen im Betriebssystem verwenden, um zu verstehen wie die Informationen im Ausgabesatz zu interpretieren sind, um die Anzeige zu erzeugen.

In einer modernen, grafischen Oberfläche , können nicht einfach so die DDS Schlüsselwörter wie DSPATR, EDTCDE etc oder Feldarten verwenden werden, ansonsten die grafischen Oberflächen immer noch auf die gleichen Funktionen wie die Legacy 5250 Displays limitiert wären. Ergo braucht es eine leistungsfähigere Methode zur Speicherung der Metadaten und eine Methode, die erweitert kann, um neue GUI Funktionen hinzuzufügen, ohne auf IBM für neue DDS-Schlüsselworte warten zu müssen.

Wie speichert ProfoundUI RUI Metadaten?

Bei der Entwicklung von ProfoundUI, verfolgte das Entwicklerteam verschiedene Ansätze zum Speichern der Metadaten:

1. Verwenden der vorhandenen DDS Schlüsselworte
2. Speichern der Metadaten in einer separaten physischen Datei
3. Speichern der Metadaten in einem separaten Streamfile
4. Einen Weg finden, um die eigenen Metadaten in der Bildschirmdatei selbst einzubetten

Die erste Option wurde schnell verworfen. Wenn Green-Screen-Schlüsselworte verwendet werden, kann zwar eine nette grafische Oberfläche angezeigt werden, aber die Features die eine moderne Bildschirmoberfläche ausmachen, könnten nicht implementiert werden und eine weitere Entwicklung verunmöglichen.

Die 2. und 3. Option würde funktionieren, aber auch einige grosse Probleme schaffen! RPG Dialogprogramme benötigen für die Ausgabe eine Bildschirmdatei in den F-Spezifikationen. Wenn die Metadaten zusätzlich in einer separaten Datei (ausserhalb der Bildschirmdatei) gespeichert würden, müsste ein Weg gefunden werden, um die beiden Dateien zu korrelieren.

Zum Beispiel, jedes Mal, wenn Kopierfunktionen wie MOVOBJ oder CRTDUPOBJ für die Bildschirmdatei verwenden werden, muss auch die externe Metadaten-Datei zur gleichen Zeit berücksichtigt werden. Das Gleiche gilt bei Rückspeicherungen und Sicherungen. Wird eine Hochverfügbarkeits-Software verwendet, muss man diese Objekte synchronisieren, als wären sie ein Objekt. Bei einer Change Management-Software braucht man zu verstehen in welcher Beziehung die Dateien stehen, wie sie eingesetzt werden und so weiter. Erhebliche Anpassungen oder Setups des Change-Management-Paketes können notwendig werden und damit auch ein erhöhter Wartungsbedarf. Ausserdem performt das Öffnen und Lesen zweier Dateien (eine physische oder IFS-Stream Datei plus das benötigte Displayfile) weniger, als nur bei einer Bildschirmdatei.

Diese Ueberlegungen führten zur vierten Option: Speichern der eigenen Metadaten in der Bildschirmdatei selber. Dem zu Grunde liegen zwei wichtige Technologien:

1. **IBM 's HTML DDS Schlüsselwort.** Dies ist ein Schlüsselwort, das IBM der DDS seit langer Zeit hinzugefügt hat, um Software zu unterstützen, die Web-Daten innerhalb der Bildschirmdatei selbst enthält. Trotz des Namens des Schlüsselwortes "HTML", können beliebige Zeichenfolgen von Daten gespeichert werden. Und so können auch unbegrenzte Mengen von Metadaten zur Anzeige in der Bildschirmdatei selbst abgelegt werden.

2. **JavaScript Object Notation (JSON).** JavaScript ist die native Programmiersprache der Web-Browser und die am weitesten verbreitete Programmiersprache der Welt. JSON ist JavaScripts natives Format für die Speicherung von Objekten, Datenstrukturen und Arrays. Es hat ähnliche Fähigkeiten wie XML, ist aber kompakter und wird nativ von den Web-Browsern verstanden.

Wird nun ein Bildschirm mit dem ProfoundUI Rich Designer entworfen, werden alle Bildschirmdaten im JSON-Format gespeichert. Das ist kein Wunder, denn der Designer ist in JavaScript geschrieben, so dass es auch der "natürlichste" Weg ist, die Daten aufzubereiten. Beim Speichern einer entworfenen Anzeige werden diese JSON-Strings via HTML-Keyword in die DDS Bildschirmdatei eingebettet.

Mit diesem Design sind alle Bildschirmdaten - einschließlich des Bildschirm Satzformates welches von RPG und Open Access benötigt wird und die erforderlichen Metadaten der GUI-Anzeige - **an einem Ort gespeichert**.

```
A                                       INDARA
A          R MYREC
A  30                                   CA01
A                                  1  2HTML('QPUIREC0    ')
A                                  1  3HTML('{"screen":{"record format nam-
A                                      e":"MYREC"},"items":[{"id":"web_pag-
A                                      e_id","field type":"textbox","css c-
A                                      lass":"input","value":{"fieldName":-
A                                      "PPRICE","dataLength":"11","decPos"-
A                                      :"2","numSep":"false","zeroBalance"-
A                                      :"false","numBlankFill":"false","ze-
A                                      roFill":"false","curSym":"","dataTy-
A                                      pe":"zoned","formatting":"Number","-
A                                      negNum":"-999.00","units":"","desig-
A                                      nValue":"[PPRICE]"},"left":"16px","-
A                                      top":"16px","read only":{"fieldName-
A                                      ":"30","dataType":"expression","for-
A                                      matting":"Indicator","indFormat":"t-
A                                      rue / false"},"text decoration":{"f-
A                                      ieldName":"N30","customTrue":"under-
A                                      line","customFalse":"none","dataTyp-
A                                      e":"expression","formatting":"Indic-
A                                      ator","indFormat":"Custom Values"}}-
A                                      ]}')
A            PPRICE        11S 2H
```

Abbildung 4: Ein einfaches Beispiel eines ProfoundUI Display Files mit Metadaten

Das Beispiel zeigt die Metadaten aus Abbildung 3 wie der Indikator 30 das Feld schützt bzw. unterstreicht oder das Feld PPRICE zu speichern ist. Nur sind die Bildschirmformate nicht mehr ausschliesslich auf diese Merkmale beschränkt. Informationen über GUI Attribute, die normalerweise nicht in DDS verwendet werden können, lassen sich hier mitgeben, wie zum Beispiel Drop-Down-Boxes, Checkboxes, Bilder, Buttons usw.

Natürlich lässt der kompliziert aussehende Code eine Bearbeitung mit SEU oder RDp in dieser Form nur schwer zu. Anstelle dessen oder SDA wird für die grafische Entwicklung auch ein grafischer WYSIWYG-Screendesigner verwendet, der die ganze Umsetzung vornimmt.

Zusammenfassung

Da die Metadaten in der Bildschirmdatei gespeichert werden, gibt es keine Probleme mit Kopieren, Verschieben oder Verbreitung des Objekts. Alles ist in einem Objekt selbst enthalten, entsprechend der traditionellen GreenScreen Methode. Daher funktioniert es auch perfekt mit einer Change-Management-Software.

ProfoundUI ist das einzige Werkzeug für grafische Bildschirme mit dieser Technik und Doppelspurigkeiten, wie das doppelte Speichern von Metadaten, zum vornhinein eliminiert. Somit lassen sich ProfoundUI Metadaten auch einfacher verwalten und effizienter verarbeiten.

Mehr Informationen:

Logic IT-Services
Kontakt: Karl Fritz, Tel. +41 79 440 24 87, Mail: k.fritz@logic.ch,
Web: www.logic.ch

Quelle: Scott Klement, Profound Logic