5

HTTP-APIs

Ein I.T.P.-Fachbuch

Um Web-Seiten zu erstellen, statisch oder dynamisch, mussten Sie in der Vergangenheit auf eine andere Maschine zurückgreifen als die AS/400. Das bedeutete zusätzliche Kosten, mehr Arbeit und weitere Komplikationen für Ihre e-Business-Lösungen. Jetzt wo IBM diese Tools für die AS/400-Gemeinde unterstützt, um diese Aufgaben auf einer Maschine durchzuführen, mit der wir vertraut sind, ist das Spielfeld wieder eben. Wir müssen nicht länger auf Plattformen zurückgreifen, die die AS/400 nicht annähernd in Stabilität und Performance erreichen, um unsere e-Business-Lösungen anzubieten.

Mit dem Release V3R2 und V3R7 von OS/400 unterstützte IBM ein Set an Application Program Interfaces (APIs), die die Common-Gateway-Interface (CGI) -Programmierung von praktisch jeder auf der AS/400 laufenden Programmiersprache aus möglich machte. Diese APIs werden für die Ausführung von Funktionen wie das Schreiben an eine Standard-Ausgabe (an einen Browser), Lesen von einer Standard-Eingabe (von Formularfeldern eines Browsers) und die Rückgabe von Umgebungsvariablen verwendet.

Diese APIs sind womöglich neu auf der AS/400, aber sie sind nicht neu für andere CGI-Programmiersprachen wie z. B. Perl, eine Sprache, die ausgiebig für die CGI-Programmierung in der Nicht-AS/400-Welt verwendet wurde. Was der Rest der Welt über Jahre hinweg verwendete, kommt jetzt in unsere Welt und gerade recht für einen enormen Boom im e-Business-Markt.

Dieses Kapitel wiederholt die am meisten gebrauchten und wichtigsten APIs, die für uns zur Verfügung stehen und gibt Beispiele für deren Verwendung mit RPG. Diese APIs sind das Grundge-

rüst für die CGI-Programmierung auf der AS/400 und dürfen deshalb nicht ignoriert werden. Ohne sie würde die Herstellung von e-Business-Anwendungen auf der AS/400 schwierig, wenn nicht sogar unmöglich sein.

Wo die APIs zu finden sind und wie sie verwendet werden

Die in diesem Kapitel angesprochenen APIs sind ein perfektes Beispiel dafür, wie das Integrated Language Environment (ILE) benützt werden kann, um dem AS/400-Entwickler signifikant zu helfen, unabhängig von der Wahl der Programmiersprache. Ich sehe die CGI-Programmierung mit RPG nicht nur als einen Weg, die Services für die Benutzer der Midrange-Systeme zu vermehren, sondern als Schwelle zur Welt der ILE.

Die für die HTTP-Programmierung verwendeten APIs befinden sich in einem Serviceprogramm namens QTMHCGI in der Bibliothek QTCP. vServiceprogramme sind ein Teil von ILE, die das Erstellen von nichtspezifischen Anwendungen auf der AS/400 ermöglichen. Deshalb können Module, die in C geschrieben sind oder in irgendeiner anderen Sprache, durch RPG-Programmierer verwendet werden, ohne Kenntnisse über die tatsächliche Programmiersprache zu haben. Sogar wenn Sie Ihre Anwendungen in COBOL schreiben, können diese APIs so wie sie sind in den Beispielen verwendet werden, die in RPG im Laufe dieses Buches gegeben werden.

Wenn Sie sich das Serviceprogramm PTMHCGI mit dem Display-Service-Program-(DSPSRVPGM)Befehl anzeigen lassen, sehen Sie einen Bildschirm ähnlich dem, der in Abbildung 5.1 gezeigt wird. Dieser Bildschirm listet die Prozeduren auf, die von diesem Serviceprogramm für die CGI-Programmierung exportiert und genutzt werden können.

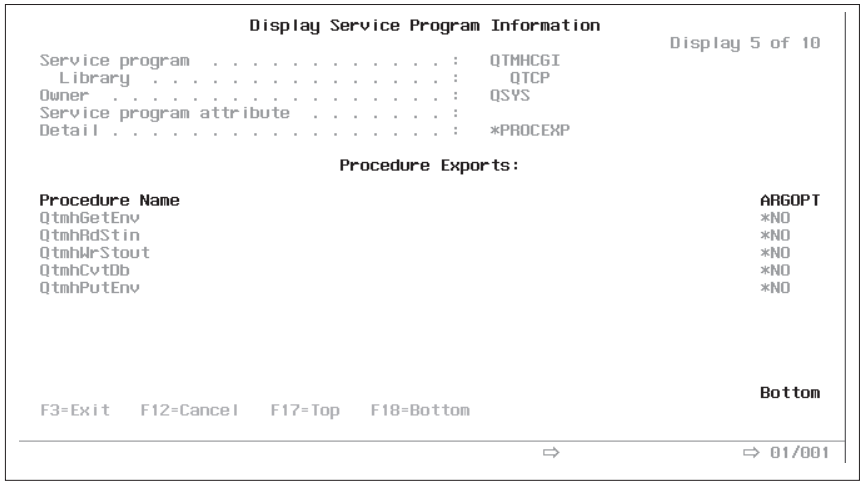

Abbildung 5.1: Die Prozeduren, die im QTMHCGI-Service-programm enthalten sind

Diese APIs werden verwendet, indem die Prozeduren von Ihrem RPG-Programm aufgerufen werden. Um sie aufzurufen verwenden Sie den CALL-Opcode, da es sich um verbundene Prozeduren handelt. Die aufgeführten Parameter, die für die Aufrufe gebraucht wird, sind die gleichen wie bei einem normalen Aufruf, mit dem die meisten von uns ja vertraut sind. Abbildung 5.2 enthält ein Beispiel von RPG-Code, um eines dieser APIs aufzurufen.

```
C                   CALLB     'QtmhWrStout'
C                   PARM                    WrtDta
C                   PARM                    WrtDtaLen
C                   PARM                    WPError
```

Abbildung 5.2: Ein Beispiel des Aufrufs eines HTTP-APIs mit CALLB

Wenn Sie ein RPG-Programm erstellen, das diese APIs verwendet, müssen Sie auf das QTMHCGI-Serviceprogramm in Ihrem Create-Bound-RPG(CRTBNDRPG)- oder Create-Program (CRTPGM)-Befehl Bezug nehmen. Sie können dies tun, indem Sie entweder das Serviceprogramm im CRTxxxPGM-Befehl spezifizieren oder ein Binding Directory festlegen, das das QTMHCGI-Serviceprogramm als Eintrag enthält. Die letztere Methode wird bevorzugt und deswegen später in diesem Kapitel erläutert.

Es ist eine gute Idee, eine Kopie des QTMHCGI-Serviceprogramms zu machen und diese in einer Bibliothek zu platzieren, die Sie für die Verwendung in Ihrer CGI-Bibliothek vorsehen. Sie können vollständig sicher sein, dass wenn irgendwelche Veränderungen entweder durch eine Versionsaktualisierung oder die Anwendung von PTFs durchgeführt werden, Ihre CGI-Programme nicht betroffen werden, weil eine Kopie des Serviceprogrammes verwendet wird. Um eine Kopie des QTMHCGI-Serviceprogrammes zu erstellen, verwenden Sie den Create-Duplicate-Object-Befehl.

```
CRTDUPOBJ OBJ(QTCP/QTMHCGI) TYPE(*SRVPGM) TOLIB(Your_CGI_Library)
```

Dieses Thema wird später in diesem Kapitel behandelt.

HTTP-APIs unter der Lupe

Jetzt wo Sie wissen, wo sich die APIs befinden und wie auf sie zugegriffen wird, detailliert der folgende Abschnitt, wie sie zur Erzeugung von RPG-Programmen verwendet werden. Jedes dieser APIs bietet eine spezielle Funktion, die Sie in Reinform in Ihren CGI-Programmen wiederfinden werden, wie das Schreiben von Informationen oder das Lesen einer Eingabe von einem Browser. Sie finden eine Zusammenfassung dieser APIs im Anhang E.

QtmhWrStout-Standardausgaben schreiben

Das wichtigste und meist verwendete HTTP-API ist das QtmhWrStout-API. Dieses API wird verwendet für das Schreiben von Daten zu Standard-Ausgaben oder mit anderen Worten: HTML- oder JavaScript-Befehle in einen Browser zu schreiben. Es veranlasst den Browser, die erwarteten Informationen anzuzeigen, genau so wie der RPG-WRITE-Opcode Daten auf einen Grünbildschirm oder in eine Druckausgabe schreibt.

Wenn Sie mit der Verwendung von dynamischen Bildschirm-APIs vertraut sind, verursacht dieses API annähernd ein déjà-vu-Erlebnis. Wenn Sie es nicht sind, sorgen Sie sich nicht. Das Bellen der APIs ist schlimmer als das Beißen.

Die bis hierher gegebenen Beispiele waren alle statische HTML. Erinnern Sie sich, dass statische HTML erstellt wird, indem ein Texteditor oder Web-Seiten-Tool verwendet wird und dieses dann

als HTML-Datei gesichert wird. Die Datei wird dann an einen Web-Server übertragen und durch die Benutzer als HTML-Dokument angefordert. Wenn es abgerufen wird, übersetzt der Browser die Information und zeigt sie in einem für Menschen lesbaren Format an.

Durch die Verwendung des QtmhWrStout-APIs erstellen Sie HTML dynamisch oder im Handumdrehen. Das Konzept unterscheidet sich nicht von dem Erstellen eines Programmes, das Informationen auf einem 5250-Terminal anzeigt. Daten werden aus einer Datei gelesen, in Bildschirmfelder mit einem Datensatzformat platziert und der Bildschirm wird an ein Terminal geschrieben, entweder mit dem WRITE- oder dem EXFMT-Opcode.

Das QtmhWrStout-API ist von einem Standpunkt der Betrachtung tatsächlich einfacher zu verwenden. Wenn Sie dieses API aufrufen, werden die Daten, die Sie damit übergeben, in diesem Augenblick in den Browser geschrieben. Es gibt also keine Bildschirmdateien, um die Sie sich Sorgen machen brauchen, was bedeutet: keine Felder oder DDS zu erstellen und keine Sorgen um Level-Prüfungen. Jedes Mal, wenn Sie dieses API benutzen, ist es als würden Sie einen nagelneuen Bildschirm erstellen. Der Unterschied ist, dass Sie den Browser anstatt eines Terminals verwenden, um die Information an den Benutzer zu übertragen.

QtmhWrStout-Parameter

Das QtmhWrStout-API hat, wie die meisten APIs, einen Satz an erforderlichen Parametern. Es ist tatsächlich das am einfachsten zu verwendende HTTP-API, was auch gut ist, da es zudem das Meistverwendete ist. Tabelle 5.1 listet die nötigen Parameter für

dieses API auf. Ein Asterisk (*) wird für die Definition der variierenden Länge verwendet.

Tabelle 5.1: Erforderliche QtmhWrStout-Parameter

Parameter	Typ (Größe) – Verwendung	Beschreibung
Data	Zeichen(*) – Eingabe	Ein variables Zeichenfeld, das die Daten enthält, wie HTML-Code, der in den Browser geschrieben werden muss.
Length of Data (Datenlänge)	Binär(4) – Eingabe	Ein binäres Feld, das dem API die Länge der Daten im Data-Parameter mitteilt.
Error Parameter	Zeichen(*) – Ein/Ausgabe	Die Standard-API-Fehlerstruktur

Data (Daten)

Der erste erforderliche Parameter ist ein Parameter, der die Daten enthält, die Sie in den Browser schreiben wollen. Dieses Feld muss nicht alle Daten für eine gesamte Web-Seite enthalten. Es enthält üblicherweise die Daten für eine Zeile des HTML-Codes, gefolgt von einem Neue-Zeile-Zeichen. Dieses API kann mehrere Male aufgerufen werden, so dass Sie damit Ihre HTML-Seite Zeile für Zeile aufbauen können. Sorgen Sie sich nicht um die Geschwindigkeit; der Benutzer, der die Seite betrachtet, wird bei der Anzeige keinen Unterschied zwischen dieser und einer statischen Seite merken.

Ein Beispiel des möglichen Inhaltes des DATA-Parameters folgt:

```
<a href="www.mysite.com">Click Here!</a>\n
```

In diesem Beispiel sind die Daten HTML-Code, der ein Hyperlink auf einer Seite produziert. Der Hyperlink liest „Click Here!“ und wenn Sie dies tun, leitet er den Browser auf www.mysite.com um. Das \n steht für ein Neue-Zeile-Zeichen.

Sie sollten eine Sache bei der Verwendung dieses APIs beachten. In einigen Versionen von OS/400 gibt es ein Limit von 1024 Zeichen in einem Datenfeld und es kann zu unerwarteten Ergebnissen kommen, falls dieses Limit überschritten wird. Ich mache es mir zur Gewohnheit, die Größe der Felder in den meisten Fällen auf 1024 zu setzen. Damit weiß ich, dass, wenn ich die Anwendung auf eine Maschine übertrage und implementiere, die mit einer älteren Version des Betriebssystems läuft, es keine Probleme gibt.

Length of Data (Datenlänge)

Der zweite Parameter enthält ein binäres Feld, das die Größe der Daten enthält, die Sie in Ihren Browser schreiben. Dieses Feld ist nicht unbedingt die Größe des Feldes selbst, aber die Größe des Feldes minus irgendwelcher angehängter Leerzeichen.

Ich gewöhne es mir an, den Wert des Feldes noch vor dem Aufruf des QtmhWrStout-APIs zu berechnen. Ich tue dies auch, wenn ich den CHECKR-Opcode oder die %LEN-Built-In-Funktion (BIF) verwende. Da %LEN-BIF nicht auf allen Betriebssystemen verfügbar ist, wenn Sie e-RPG-Programme schreiben, die

auf eine andere AS/400 übertragen werden, verwende ich in der Regel CHECKR. Die meisten Beispiele in diesem Buch verwenden zur Vereinfachung die %LEN-Methode, aber ich zeige Ihnen, wie beide Methoden arbeiten. Abbildung 5.3 enthält Beispiel-Code für beide Methoden.

```
The CHECKR Method:
C      ' '          CHECKR    Data:1024    DataLen
The %len BIF Method:
C                   eval      DataLen = %len(%trim(Data))
```

Abbildung 5.3: Zwei Möglichkeiten, die Länge des Data-Parameters für das QtmhWrStout-API zu erhalten

Error-Parameter

Der Error-Parameter ist der letzte erforderliche Parameter und ist Standard in den meisten APIs. Da es nichts Einmaliges bei diesem Parameter gibt, was das QtmhWrStout-API betrifft, bespreche ich ihn auch nicht. Wenn Sie mit diesem Parameter nicht vertraut sind, arbeiten die Beispiele in diesem Buch auch ohne Modifikationen ganz gut für Sie.

QtmhRdStin – Read Standard Input (Lesen einer Standardeingabe)

Wenn wir Daten in einen Browser schreiben können, ist es logisch anzunehmen, dass wir diese Daten auch von einem Browser lesen können. Genauso wie wir von einem AS/400-Bildschirm lesen und schreiben können, können wir beides auch auf einer Web-Seite unter der Verwendung von APIs tun. Ohne die Möglichkeit, Eingaben von einem Browser zu lesen, würde es vermessen sein zu denken, dass der Browser für etwas anderes verwendet werden kann, als dem Benutzer Informationen zu senden.

Das Lesen von Eingaben eines Browsers wird durch das QtmhRdStin-API ermöglicht. Dieses API, weniger oft verwendet als das QtmhWrStout-API, ist etwas verzwickter. Dieses API liest nur Informationen, die ihm mit der POST-Operation bereitgestellt wurden. Die POST-Operation wird mit einer HTML-Form durchgeführt, die die ACTION- und METHOD-Schlüsselwörter verwendet. Zusätzlich wird ein SUBMIT-Knopf benutzt, um die Form abzusetzen. Der HTML-Code ähnelt dem Folgenden.

```
<FORM ACTION= "/cgi-bin/CGIPGM" METHOD="POST">
…form fields…
<INPUT TYPE="SUBMIT" VALUE="Click Here">
</FORM>
```

Die Information, die dem ACTION-Schlüsselwort folgt, sagt dem Browser, was zu tun ist, wenn die Form verarbeitet wird. In diesem Fall ruft sie ein Programm namens CGIPGM auf, das sich im /cgi-bin/ Verzeichnis befindet. Das METHOD-Schlüsselwort teilt dem Browser mit, welche Methode zu verwenden ist. In diesem Fall wird die POST-Methode verwendet. Wie bei dem aktu-

ellen Absetzen (submit) der Form, wird dies mit einem Knopf vom Typ SUBMIT erledigt. Wenn dieser Knopf gedrückt wird, wird die Information in der Form für das CGI-Programm verfügbar gemacht, um sie als Standard-Eingabe lesen zu können.

Das QtmhRdStin-API wird Ihnen vertrauter werden, sobald Sie verstehen, wann und wo es verwendet wird. Wenn Sie jemals auf eine Web-Seite stoßen, wo Sie Informationen in Textboxen eingeben, wie Ihren Namen, Telefonnummer oder E-Mail-Adresse, dann haben Sie zweifelsohne diese Lese-von-der-Standardeingabe-Funktion in der einen oder anderen Form gebraucht. Das ist eine dieser „Hab‘ ich mich schon immer gefragt, wie das funktioniert.“-Situation.

Nachdem Sie Ihre Information in ein Formular eingegeben und den Submit-Knopf gedrückt haben, werden diese Daten durch ein CGI-Programm eingelesen, das die gleichen Methoden verwendet wie das QtmhRdStin-API. Die Daten stehen dann dem CGI-Programm in Form einer Standardeingabe zur Verfügung, um Vergleiche, Änderungen oder Speichern in einer Datei vorzunehmen. QtmhRdStin ist ein sehr mächtiges API und wird regelmäßig in Web-Seiten, die Benutzereingaben erfordern, verwendet.

Die Parameter von QtmhRdStin

Die für QtmhRdStin erforderlichen Parameter werden in Tabelle 5.2 dargestellt.

Tabelle 5.2: Die erforderlichen Parameter für QtmhRdStin

Parameter	**Typ(Größe) – Verwendung**	**Beschreibung**
Receiver Variable (Empfängervariable)	Char(*) – Ausgabe	Der Variablenname, der die Daten enthält, die vom Read-Standard-Input-API erhalten werden.
Length of Receiver Variable (Länge der Empfängervariable)	Binary(4) – Eingabe	Ein binäres Feld, das die Länge der Receiver Variable enthält.
Length of Response Available (Länge der verfügbaren Antwort)	Binary(4)- Ausgabe	Die Länge der Daten, die vom Read-Standard-Input-API gelesen werden.
Error Parameter (Fehlerparameter)	Char(*) – Ein/Ausgabe	Die Standard-API-Fehlerstruktur

Receiver Variable

Der erste Parameter, die Receiver-Variable, enthält die Daten, die durch das QtmhRdStin-API eingelesen werden. Diese Daten liegen nahezu immer in der gleichen Form vor: sie enthalten die Namen der Felder, gefolgt von einem Istgleich-Zeichen und dem Wert dieser Variable. Jedes Variable/Wert-Kombination wird durch ein Ampersand (&) abgetrennt, wie es das folgende Beispiel zeigt.

```
FirstName=Brad&LastName=Stone
```

Die erste Variable ist FirstName, die den Wert „Brad“ enthält. Die zweite Variable ist LastName und enthält den Wert „Stone“. Das Ampersand (&) trennt die zwei Variablen/Wert-Kombinationen. Fangen Sie nicht jetzt schon an, sich Gedanken über das Herauslösen der Daten zu machen; es gibt einen einfachen Weg dafür und der wird später in diesem Kapitel behandelt.

Length of Receiver Variable

Der zweite Parameter für das QtmhRdStin-API enthält einen Wert, den Sie für das API spezifizieren. Der Parameter teilt dem API mit, wie viele Bytes, die von der Receiver-Variable eingelesen werden, Sie erwarten. Das ist nicht die Länge der Variable selbst, sondern die Länge der Receiver-Variable minus der angehängten Leerzeichen.

Dieser Parameter macht vielen AS/400-Web-Entwicklern Probleme. Wenn Sie nicht den korrekten Wert verwenden, wird Ihre letzte Variable möglicherweise nicht verarbeitet. Das API wird womöglich den letzten Parameter ignorieren, was Kopfschmerzen und Frustration während des Testens und Debuggens der CGI-Anwendung hervorruft.

Die Content Length Environment-Variable sollte für diesen Parameter verwendet werden. Das API QtmhGetEnv, das diesen Wert zurückholt, wird später in diesem Kapitel behandelt.

Manche Leute finden, dass die Verwendung einer Dummy-Variable in ihren Formalaren ihr Problem lösen kann, dass dieses API nicht die letzte Variable herauslöst. Eine Dummy-Variable trickst den Computer aus, indem sie ihn glauben macht, dass es

noch eine Variable im Formular gibt und die Dummy-Variable wird nicht genommen. Das ist kein Problem für den Entwickler, da alle wichtigen Variablen herausgelöst wurden. Ich befürworte diese Methode nicht. Verwenden Sie statt dessen die Content-Length-Environment-Variable. (Ich erwähne diese Methode, da Sie sich keine Sorge machen sollten, wenn das Problem des „missing last Parameter“ auftritt.) Manche Entwickler sagen Ihnen, dass Sie die Dummy-Variablen-Methode verwenden sollen. IBM dagegen stellt heraus, dass Sie inkorrekte Daten an das Convert-to-Data-API übergeben und dass Sie den Wert verwenden sollen, der von der Content-Length-Variable zurückgegeben wird. Der dritte Parameter teilt Ihnen die Länge der Receiver-Variable mit, die vom QtmhRdStin-API zurückgeholt wurde. Wenn keine Daten aus der Standardeingabe gelesen wurden, wird dieser Wert null sein.

Length of Response Available (Länge der verfügbaren Antwort)

Der nächste Parameter ist die Length of Response Available und wird vom QtmhRdStin-API zurückgegeben. Nach Aufruf des QtmhRdStin-API enthält dieser Parameter die Länge der Daten, die von der Standardeingabe gelesen werden. Wenn keine Daten aus der Standardeingabe gelesen wurden, ist dieser Wert null.

Error Parameter

Der letzte Parameter ist wiederum der Standard-Fehlerparameter, der von den meisten verwendeten APIs genutzt wird.

Ein QtmhRdStin-Beispiel

Um Ihnen ein Beispiel des QtmhRdStin-API zu geben, lassen Sie uns annehmen, dass Sie nur ein Formular brauchen, in das Sie Ihren Vornamen (Feldname fname) und Ihre Telefonnummer (Feldname phone) eingeben sollen. Wenn Sie sich die Daten nach Verwendung dieses APIs anschauen, werden sie sicher der folgenden Zeichenkette ähneln:

```
fname=Bradley&phone=15075551212
```

In diesem Beispiel repräsentieren fname und phone die Variablennamen und die Daten nach dem Istgleich-Zeichen sind die tatsächlichen Daten, die dieser Variable zugewiesen werden. Das Ampersand(&)-Zeichen trennt Felder und Daten. In diesem Fall sagen uns die Daten, dass der Vorname „Bradley“ und die Telefonnummer „15075551212“ ist.

Von jetzt an können Sie Ihre Daten selbst herauslösen, aber es gibt einen viel leichteren Weg dies zu tun, indem Sie das Convert-To-DataBase-API verwenden. Da das Convert-To-DataBase-API auch zusammen mit anderen APIs verwendet wird, die Daten zurückgeben, wird es am Schluss behandelt.

QtmhGetEnv – Get Environment Variable (die Umgebungsvariable erhalten)

Das QtmhGetEnv-API wird verwendet, um Werte, die vom Server für bestimmte Umgebungsvariablen festgelegt wurden, zurück zu erhalten. Umgebungsvariablen sind von Benutzer zu Benutzer, der auf Dokumente auf Ihrem Web-Server zugreift, verschieden. Genauso wie manche Benutzer, die ein RPG-Programm laufen haben, bestimmte Werte in der Program-Status-Data-Struktur (SDS) haben, enthalten die Umgebungsvariablen spezifisch auf den Benutzer, der Ihre Web-Seite besucht, abgestimmte Informationen.

Zum Beispiel sind einige Variablen für eine bestimmte Umgebung auf Werte gesetzt, die für eine Person passen, die auf Dokumente zugreift, während eine andere Person wieder unterschiedliche Werte für bestimmte Umgebungsvariablen braucht. Zwei Benutzer, die ein RPG-Programm laufen lassen, können unterschiedliche Werte für die User-ID-Variable in SDS haben. Genauso können zwei Benutzer, die auf Ihren Web-Server zugreifen, für ihr spezielles Anliegen passende Informationen haben, wie die erwartete IP-Adresse. Ein anderes gutes Beispiel würde der Browsertyp sein. Ein Benutzer verwendet zum Beispiel Netscape und ein anderer den Internet Explorer. Beide Werte können als Umgebungsvariablen abgerufen werden.

Eine andere Verwendung von Umgebungsvariablen, die schon vorher erwähnt wurde, ist die Rückgabe der Länge des Inhaltes der Standardeingabe-Variable, die durch die POST-Operation bereitgestellt wird.

Umgebungsvariablen werden auch durch die GET-Operation ermöglicht. Anstatt die Daten mit Standard-Eingaben zu übergeben, kann man die Daten auch als Umgebungsvariable verfügbar machen. Dies ist bekannt als die *Query-String*-Umgebungsvariable. Das sind die Daten, die der Web-Adresse in URL folgen, wie im Beispiel gezeigt.

```
http://www.mypage.com/cgi-bin/mycgi.pgm?FirstName=Brad&LastName=Stone
```

Diese URL greift auf ein CGI-Programm namens MYCGI zu. Danach folgt ein Fragezeichen (?) und ein Satz an Query-String-Umgebungsdaten. Diese Daten sehen vielleicht vertraut aus, da das Format ähnlich dem der Daten ist, die von der Standardeingabe kommen. Die HTML-Quelle ist ähnlich der Quelle, die die POST-Operation ausführt, wie hier gezeigt:

```
<FORM ACTION= "/cgi-bin/CGIPGM" METHOD="GET">
…form fields…
<INPUT TYPE="SUBMIT" VALUE="Click Here">
</FORM>
```

Der einzige wirkliche Unterschied ist hier der Wert des METHOD-Schlüsselwortes im Formular. Anstatt die Daten von der Standardeingabe verfügbar zu machen, wenn der SUBMIT-Knopf gedrückt wird, werden die Daten aus der Query-String-Umgebungsvariable genommen und die sind genauso passend, wie die Daten die nach dem URL kommen. Aus diesem Grund sollte diese Methode nicht für sensible Daten verwendet werden.

QtmhGetEnv-Parameter

Tabelle 5.3 enthält die Parameter, die vom QtmhGetEnv-API verwendet werden.

Tabelle 5.3: Die von QtmhGetEnv benötigten Parameter

Parameter	**Typ(Größe) – Verwendung**	**Beschreibung**
Receiver Variable	Zeichen(*) – Ausgabe	Der Variablenname, der die Daten enthält, die vom Read-Standard-Input-API geholt werden.
Length of Receiver Variable	Binär(4) – Eingabe	Ein binäres Feld, das die Länge der Receiver Variable enthält.
Length of Response	Binär(4) – Ausgabe	Die Länge der Daten, die vom Read-Standard-Input-API eingelesen werden.
Request Variable	Zeichen(*) – Eingabe	Die Länge der Umgebungsvariable, für Antwort.
Length of Request Variable	Binär(4) – Eingabe	Die Länge der Request-Variable (Anwortvariable).
Error Parameter	Zeichen(*) – Ein/Ausgabe	Die Standard-API-Fehlerstruktur

Die Parameter des QtmhGetEnv-APIs sind ähnlich denen, die beim QtmhRdStin-API verwendet werden. Die Request-Variable enthält den Namen der Umgebungsvariablen und die Länge der Request-Variable enthält den Wert, der dem System die Länge der übergebenen Variable mitteilt. Das mag verwirrend erscheinen, aber es wird im folgenden Abschnitt beschrieben.

Receiver Variable

Der Receiver-Parameter enthält die Daten, die vom QtmhGetEnv-API zurückkommen. Er ist im Format den Daten ziemlich ähnlich, die vom QtmhRdStin-API kommen.

Length of Receiver Variable (Die Länge der Empfänger-Variable)

Der Length-of-Receiver-Parameter enthält einen Wert, in Ihrem Programm festgelegt, der dem API die Größe der Daten mitteilt, die von der Umgebungsvariable abgerufen werden. Dieses Feld kann auf die Größe der Receiver-Variable gesetzt werden.

Length of Response (Die Länge der Antwort)

Der Wert, der in dem Length-of-Response-Parameter zurückgegeben wird, enthält die Länge der erhaltenen Eingabedaten. Wenn es nicht möglich ist, diesen Wert zu berechnen, wird null zurückgegeben. Wenn die Größe der Receiver-Variable zu klein ist, enthält dieser Parameter die tatsächlich erforderliche Größe, um die Rückgabe durchzuführen.

Request Variable (Antwortvariable)

Der Request-Variable-Parameter enthält den Namen einer Umgebungsvariablen, die Sie von einer Standardeingabe erhalten wollen. Zum Beispiel möchten Sie Query-String-Daten erhalten, dann würde dieser Parameter der Wert „QUERY_STRING“ enthalten.

Length of Request Variable (Die Länge der Anwortvariable)

Der Length-of-Request-Variable-Parameter enthält die Länge der Request-Variable, die im vorigen Parameter festgelegt wurde.

Error Parameter

Der Standard-Fehlerparameter, der von den meisten verwendeten APIs genutzt wird.

Umgebungsvariablen, die von QtmhGetEnv verwendet werden

Umgebungsvariablen werden für die Identifizierung des Datentyps verwendet, den Sie von Ihrem aktuellen Rückgabejob erhalten wollen. Da es eine große Anzahl an Umgebungsvariablen für Sie gibt, werde ich nur die wichtigsten besprechen.

Tabelle 5.4: Einige der Umgebungsvariablen, die von QtmhGetEnv verwendet werden.		
Umgebungsvariable	**Variablenname**	**Beschreibung**
Request Method (Rückgabemethode)	REQUEST_METHOD	Die Rückgabemethode, die ausgeführt wurde. Dies kann entweder GET oder POST sein.
Remote Host (Bezugsquelle)	REMOTE_HOST	Der Host-Name, der die Rückgabe durchführt.
Remote Address (Bezugsadresse)	REMOTE_ADDR	Die IP-Adresse des Host-Namens, der die Rückgabe vornimmt.
Content Length (enthaltene Länge)	CONTENT_LENGTH	Die Länge der Daten, die von einer POST-Methode verarbeitet werden sollen.
Query String (Abfragezeichenkette)	QUERY_STRING	Die Information, die in einem URL dem Fragezeichen folgt, welche die durch ein CGI-Programm zu verarbeitenden Daten repräsentiert.

–> Die Request-Method-Umgebungsvariable ist ziemlich einfach und enthält einen von zwei Werten, entweder GET oder POST, abhängig von der Request-Methode, die für den Zugriff auf das Dokument verwendet wurde.

–> Die Content-Length-Umgebungsvariable wird meistens für die Rückgabe eines Wertes verwendet, der die Länge der zu verarbeitenden Daten enthält, in den meisten Fällen durch das Read-Standard-Input-API.

–> Die Query-String-Umgebungsvariable wird sehr oft verwendet, wenn man Daten verarbeitet, die der URL folgen und beginnt mit einem Fragezeichen (?). Diese Daten werden bereitgestellt, wenn Sie eine GET-Requestmethode verwenden. Sie haben diesen Datentyp vielleicht schon einmal gesehen, wenn Sie auf die Positionierungsleisten Ihres Browsers acht geben, wenn Sie eine Suche nach einer Web-Seite durchführen. Die URL sieht normalerweise so ähnlich wie Folgende aus:

```
http://www.pagename.com/query?search=as400&perpage=10
```

Das teilt der URL mit, ein Programm namens query zu starten. Die auf das Fragezeichen folgende Information ist der zu verwendende Variablenname und der Wert dieser Variablen. Die Variablen und die Daten werden durch ein Ampersand (&) getrennt. In diesem Fall sagen uns die Daten, dass die Variable namens search den Wert „as400“ enthält und die Variable mit Namen perpage den Wert „10“. Wenn das QtmhGetEnv-API verwendet wird, um die Query-String-Umgebungsvariablen zurück zu erhalten, sind die erhaltenen Daten nur die, die auf das Fragezeichen folgen. Das ähnelt dem Folgenden:

```
search=as400&perpage=10
```

Wieder wie bei dem QtmhRdStin-API sind die Daten sehr ähnlich und Sie denken möglicherweise immer noch daran, wie Sie die Daten heraus lösen können. Aber das gleiche Convert-To-Data-API, das durchgeführt wird, um diesen Datentyp heraus zu lösen, kann für beide APIs verwendet werden, dieses und das QtmhRdStin-API.

QtmhCvtDB – Convert to DB (Konvertieren in eine Datenbank)

Die Umwandlung von Daten von einer Query-String-Variablen oder der Read-Standard-Input-Operation kann für Sie alleine langweilig sein. Glücklicherweise unterstützt IBM ein API, das diese Schmutzarbeit für Sie erledigt.

Das QtmhCvtDB-API wird für die Umwandlung von Datenzeichenketten verwendet, die von einem CGI-Programm in Felder zurückgegeben werden, die den gleichen Namen wie in einer unterstützten DDS-Dateistruktur haben. Obwohl die Daten in ein CGI-Programm im Zeichenformat eingelesen werden, werden die Daten trotzdem in den Typ umgewandelt, der in der DDS des unterstützten Dateinamens angegeben ist.

QtmhCvtDB-Parameter

Tabelle 5.5 zeigt die vom QtmhCvtDB-API verwendeten Parameter.

Tabelle 5.5: Die von QtmhCvtDB benötigten Parameter

Parameter	**Typ(Größe) – Verwendung**	**Beschreibung**
Database Name (Name der Datenbank)	Zeichen(20) – Eingabe	Eine Variable, die den qualifizierten Datenbanknamen enthält, der die Feldbeschreibung der Umwandlung enthält. Die ersten 10 Zeichen sind der Dateiname und die letzten 10 Zeichen sind der Bibliotheksname.
Input String (Eingabezeichenkette)	Zeichen(*) – Eingabe	Der Variablenname, der die Zeichenkette enthält, die vom Read-Standard-Input-API oder der Query-String-Umgebungsvariablen zurückgegeben wird.
Length of Input String	Binär(4) – Eingabe	Ein binäres Feld, das die Länge der herauszulösenden Eingabezeichenkette enthält.
Response Variable (Antwortvariable)	Zeichen(20) – Eingabe	Die Variable, die die Struktur enthält, die mit der entsprechenden Datenbankdatei zusammenhängt, beschreibt die Eingabeparameter, die vom CGI-Programm vorausgeahnt werden.
Length of Response Available (Länge der verfügbaren Antwort)	Binär(4) – Eingabe	Die gesamte Länge des Puffers in den der CGI-Parameter hineingestellt wird.

Length of Response Variable (Länge der verfügbaren Variable)	Binär(4) – Ausgabe	Die Länge der Antwort. Wenn die Response-Variable zu klein ist, um die gesamte Antwort zu beinhalten, wird dieser Parameter auf die Größe gesetzt, die für den Inhalt der gesamten Antwort nötig ist.
Response Code (zurückgegebener Code)	Binär(4) – Ausgabe	Ein Code, der den Status der Operation enthält.
Error Parameter	Zeichen(*) – Ein/Ausgabe	Die Standard-API-Fehlerstruktur

Lassen Sie sich nicht von der Anzahl der Parameter dieses APIs beeindrucken. “APIs, die bellen, beißen nicht“. Aus den ganzen aufgelisteten Parametern sind es nur 4, die einer Erklärung bedürfen. Der Rest kann vor der Ausführung des APIs herausgesucht werden.

Database Name

Dieser Parameter ist der Name der Datenbankdatei, die für die Berechnung von Feldtyp und Feldgröße verwendet wird. Die ersten 10 Zeichen enthalten den Feldnamen und die zweiten 10 Zeichen die Bibliothek, wo die Datei sich befindet. Die hier angegebenen Datei- und Bibliotheksnamen sollten in Großbuchstaben sein.

Input String

Der zweite Parameter, den wir bringen müssen, ist der Input String. Sie haben diese Information bereits von einem der anderen APIs, die Sie verwendet haben, um die Informationen von einem

Browser zu lesen. Dies ist eine Zeichenfolge, die zurückgegeben wird unter der Verwendung des QtmhRdStin-API oder der Query-String-Umgebungsvariable, was bewirkt, dass das QtmhGetEnv-API QUERY_STRING als Umgebungsvariable spezifiziert.

Length of Input String

Der nächste erforderliche Parameter ist die Länge der Eingabe-Zeichenkette. Dieser Wert ist die Länge der Eingabezeichenkette minus irgendwelcher anhängender Leerzeichen.

Response Variable

Die Response-Variable wird normalerweise in Datenstrukturen verwendet, die extern definiert sind und die gleichen Dateispezifikationen benutzen, die im Database-Name-Parameter dieses APIs angegeben sind. Nach einem erfolgreichen Aufruf dieses APIs enthält die Datenstruktur, die für diesen Parameter angegeben ist, die Werte von der Eingabe-Zeichenkette. Feldnamen in der Eingabe-Zeichenkette werden mit Feldnamen der Datenstruktur verbunden. Mit anderen Worten: ein Teil der Eingabe-Zeichenkette enthält den Wert:

```
FNAME=Brad
```

Dann enthält das Feld in der Datenstruktur mit Namen FNAME den Wert „Brad“. Das Äußere ist nicht die Frage bei dieser Variable. Der Variablenname in der Eingabezeichenkette kann in Groß- oder Kleinbuchstaben sein und wird in das richtige Feld der Datenstruktur platziert.

Length of Response Available

Der nächste Parameter ist die Länge der Antwortvaribale. Dieses Feld ist leicht berechnet durch die Angabe der Größe der Antwortvariable.

Length of Resoponse Variable

Der nächste Parameter enthält die tatsächliche Länge der zurückgegebenen Antwort. Wenn die verwendete Antwort-Variable zu klein ist, um alle Antwortdaten zu beinhalten, enthält dieses Feld den Wert der Größe der benötigten Antwortvariable.

Response Code

Der Response-Code-Parameter wird zurückgegeben und teilt Ihnen mit, ob der Aufruf dieses APIs erfolgreich war. Tabelle 5.6 enthält eine Liste der Response-Codes und deren Bedeutung.

Tabelle 5.6: Response-Codes, die vom QtmhCvtDB-API zurückgegeben werden

Response-Code	Bedeutung
0	Alle Felder wurden entsprechend der Datenbankdatei erfolgreich übersetzt.
-1	Die Datenbankdatei enthält ein oder mehr Felder, die in dieser Eingabe nicht spezifiziert wurden.
-2	Die Eingabe enthält ein oder mehrere Felder, die in der Datenbankdatei nicht spezifiziert sind.
-3	Eine Kombination der Response-Codes -1 und -2.
-4	Ein Fehler ist während der Umwandlung aufgetreten. Die Daten könnten möglicherweise für Ihr Programm unbrauchbar sein.
-5	Dieses API wurde durch ein Programm aufgerufen, das nicht auf dem HTTP-Server läuft. Die Anfrage wird ignoriert.
-6	Die Anfrage wurde gemacht, während des Arbeitens im Binär-Modus (%%BINARY%%). Die Anfrage wird ignoriert.

Wenn Sie den Response-Code 0 oder -1 erhalten, sind die Daten überwiegend ohne Probleme konvertiert worden. Ich tendiere dazu, eine Datenbankdatei zu definieren, die alle gültigen Ein-

gabeparameter in einer bestimmten Anwendung enthält. Da nicht alle Felder in allen Aufrufen für dieses API verwendet werden können, erhalte ich einen Response-Code von -1. Das bedeutet, dass die Daten, die ich konvertiert haben möchte, erfolgreich umgewandelt wurden, aber es gibt Felder in meiner Datenbankdatei, die nicht in meiner gelieferten Eingabevariable existieren. Allen anderen Response-Codes sollte nachgegangen werden.

Error Parameter

Auch für dieses API wird der Error-Parameter wieder benötigt.

Die APIs zum Gebrauch herrichten

Nachdem Sie jetzt die Informationen haben, was diese APIs tun, müssen Sie sie einrichten, so dass Sie sie in Ihren Anwendungen verwenden können. Hier gehen wir mehr ins Detail, was Sie tun müssen, um dieses APIs für Ihre CGI-Programme verfügbar zu machen.

Kopieren des QTMHCGI-Serviceprogramms

Das erste Wichtige ist, sich zu entscheiden, wie der Name der Bibliothek lauten soll, die Ihre CGI-Programme enthalten soll. Ich bevorzuge für meine AS400CGI. Das lässt keinen Zweifel daran, was diese Bibliothek tut.

Als nächstes sollten Sie eine Kopie des QTMHCGI-Serviceprogramms in diese Bibliothek stellen. Wie vorher bereits erwähnt, ist dies ein gutes Verfahren, da, wenn OS aktualisiert wird oder PTFs irgendetwas in diesen Serviceprogrammen ändern, Ihre CGI-Anwendungen davon unberührt bleiben. Um dieses Serviceprogramm in Ihre CGI-Bibliothek zu kopieren, verwenden Sie folgenden Befehl:

```
CRTDUPOBJ OBJ(QTCP/QTMHCGI) TYPE(*SRVPGM) TOLIB(AS400CGI)
```

Es ist auch eine gute Idee, dieses Serviceprogramm zurück zu kopieren, wenn Sie Ihre OS-Version aktualisieren oder von Zeit zu Zeit. Wenn Sie meinen, dass Dinge gemütlicher arbeiten, ist es möglicherweise nicht von erster Priorität. Aber wenn Sie der

Überzeugung sind, dass Sie Probleme nach der Anwendung von PTFs oder einer neuen Version von OS haben, sollten Sie dieses Serviceprogramm in Ihr Verzeichnis zurück kopieren, um sicher zu gehen, dass Sie die letzte Version erhalten. Denken Sie auch daran, dass Sie den HTTP-Server anhalten und neu starten müssen, so dass die laufenden HTTP-Jobs die neue Version verwenden werden.

Ein Binding Directory erstellen

Bevor Sie daran gehen, Ihre CGI-Anwendungen zu erstellen, würde es eine gute Idee sein, ein verbundenes Verzeichnis für die allgemeinen Belange zu erstellen. Das verbundene Verzeichnis enthält eine Liste der Module und/oder Serviceprogramme. Wenn Sie Ihr RPG-Programm kompilieren und ein verbundenes Verzeichnis im CRTBNDRPG-Befehl angeben, sucht es in dieser Liste der Module/Serviceprogramme in dem verbundenen Verzeichnis nach verwendeten Unterprozeduren.

Ich bevorzuge, dieses allgemeinbetreffende CGI-Binding-Directory CGIBNDDIR zu nennen. Platzieren Sie das verbundene Verzeichnis in der CGI-Bibliothek, die Sie im vorigen Abschnitt erstellt haben. Verwenden Sie dazu folgenden Befehl, um ein verbundenes Verzeichnis zu erstellen:

```
CRTBNDDIR BNDDIR(AS400CGI/CGIBNDDIR)
```

QTMHCGI zu Ihrem verbundenen Verzeichnis hinzufügen

Der letzte Einrichtungsschritt ist das Hinzufügen des QTMHCGI-Serviceprogrammes zu Ihrem verbundenen Verzeichnis. Dieses Serviceprogramm wird von den meisten, wenn nicht von allen, Ihren CGI-Programmen, verwendet. Deshalb bezeichne ich dieses verbundene Verzeichnis als allgemeinbetreffendes verbundenes Verzeichnis.

Um das QTMHCGI-Serviceprogramm Ihrem verbundenen Verzeichnis hinzuzufügen, verwenden Sie folgenden Befehl:

```
ADDBNDDIRE BNDDIR(AS400CGI/CGIBNDDIR) BNDDIRE(AS400CGI/QTMHCGI)
```

Wenn dieses Hinzufügen zu Ihrem verbundenen Verzeichnis einmal gemacht wurde, können Sie das verbundene Verzeichnis im CRTxxxPGM-Befehl spezifizieren, anstatt jedes Serviceprogramm oder Modul individuell im Befehl angeben zu müssen. Es können mehr Module und Serviceprogramme in dieses Verzeichnis hinzugefügt werden, um den Prozess Ihrer Programmerstellung einfacher zu machen.

Wagen – nach Osten!

So, wir haben es geschafft, uns durch alle erforderlichen Informationen für den Beginn des CGI-Programm-Schreibens durch zu arbeiten. Es war ein langer Ritt und ab jetzt wird es ein Spaß sein. Nach dem Lernen aller Komponenten, die man für das Schreiben von CGI-Programmen auf meiner AS/400 braucht, finde ich, dass der Rest der Prozedur ziemlich einfach ist. Da meine CGI-Programme mit RPG geschrieben werden und da ich eine Menge an APIs vorher schon verwendet hatte, wusste ich, dass es nicht sehr schwer werden würde, diese CGI-Programme zu schreiben.

Die Anwendung der bis dahin beschriebenen Techniken und APIs mag wie eine schwere Arbeit aussehen, aber wenn Sie einmal damit begonnen haben, werden Sie entdecken, dass alle diese Dinge auf natürliche Weise zusammenfinden. Es gibt eine logische Struktur, dass sich alles zum Guten wendet und CGI-Programme sind da keine Ausnahme.