



Dateien auf CD



Axel Romahn / Michael Groß

PDF-Viewer in Flash

Flash und PDF: Zwei Formate, die nichts miteinander zu tun haben? Keineswegs. Dieser Workshop zeigt Ihnen, wie Sie PDF-Dateien in Flash einlesen können, und reizt dabei Highend-Technologien bis ans Ende aus. Passend dazu finden Sie auf der CD die im Workshop besprochenen Beispiel-Dateien.

Die Entstehung des hier gezeigten PDF-Viewers hat eine lange Vorgeschichte: Im Rahmen eines Produkt-Management-Systems wurde eine browserbasierte Rich Internet Application benötigt, mit der man clientseitig PDF-Dokumente prüfen und mit Anmerkungen versehen kann. Es entstand ein Flash-Frontend, das über einen Java Applikation Server PDF-Dokumente visualisiert. Das hier vorliegende Tool „PDF-Viewer“ ist eine vereinfachte Variante dieser Flash-Anwendung. Die ursprüngliche Version entstand unter Flash 5, aktuelle Versionen laufen unter Flash 6.

System-Architektur. Das System ist einfach strukturiert und besteht aus einer Flash-Client-Applikation, dem PDF-Viewer und einer Server-Implementierung. Die Grafik auf der nächsten Seite stellt die System-Architektur schematisch im Überblick dar.

WebCode FL0573

Die entwickelte Server-Anwendung ist hauptsächlich für das Rastern der PDF-Dokumente in Bitmapgrafiken zuständig und verwaltet nebenbei die PDF-Dokumente. Der Server stellt seine Funktionen den Client-Anwendungen über eine definierte Schnittstelle zur Verfügung. Die Client-Anwendung kommuniziert über HTTP-Requests mit dem Server. Über die Schnittstelle holt sich die Flash-Applikation die Daten und stellt die Ergebnisse dar. Damit ist die Aufgabenstellung beider Komponenten klar voneinander abgegrenzt, was der gesamten System-Architektur zugute kommt. Durch die Aufteilung wird außerdem eine saubere Trennung zwischen Darstellungslogik (Client) und Geschäftslogik (Server) erreicht.

Dank dieser zuvor vereinbarten Schnittstelle zwischen Client und Server ließen sich von Beginn an beide Komponenten unabhängig entwickeln. Die Zusammenführung beider Komponenten erfolgte erst später, damit der PDF-Viewer mit „Live“-Daten weiterentwickelt und getestet werden

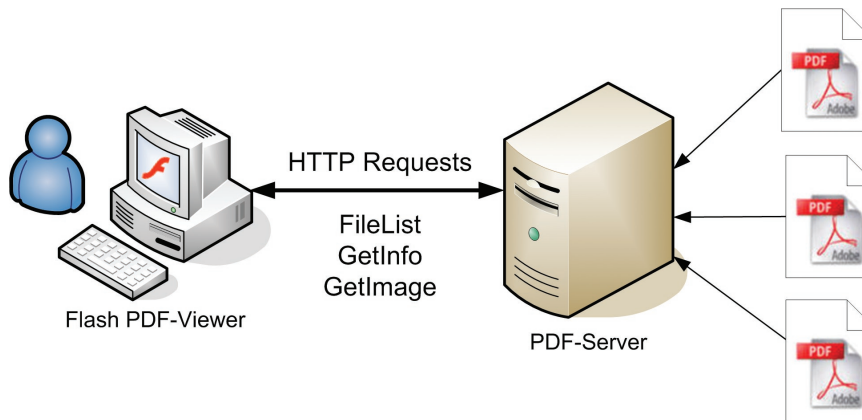
konnte. Die Erfahrung zeigt, dass eine gut durchdachte und möglichst genaue Schnittstelle-Spezifikation gleich zu Beginn gerade bei größeren und komplexeren Projekt extrem hilfreich ist und spätere umfangreiche Korrekturarbeiten vermeidet.

Auf der Bühne. Die Flash-Applikation dient hauptsächlich der Aufgabe, gerasterte Ausschnitte eines PDF-Dokumentes darzustellen (siehe Ablauf-Grafik nächste Seite). Für die Navigation im Dokument bietet die Applikation dem Benutzer eine Reihe von Funktionen an, die man auch aus einem Grafikprogramm kennt. Das Tool bietet Möglichkeiten wie stufenweises Zoomen, das Drehen des Dokumentes in und gegen den Uhrzeigersinn sowie ein seitenweises Blättern im Dokument. Zusätzlich ist eine History-Funktion realisiert, mit der sich bereits angefragte Ausschnitte durchblättern lassen. Dabei wird der Ausschnitt aus dem Cache des Browsers geladen, was einen enormen Geschwindigkeitsgewinn darstellt.

Vom PDF zum Bitmap. Der PDF-Viewer erfragt vom Server einen Ausschnitt aus einem PDF-Dokument über die Server-Funktion *getImage*. Dieser Ausschnitt hat in unserem Beispiel eine feste Größe von 600x600 Pixel. Der tatsächlich angezeigte Bildausschnitt ist jedoch nur 600x470 Pixel groß. Dennoch wird ein quadratischer Ausschnitt geladen, da auf diese Weise der Ausschnitt einfach gedreht werden kann, ohne einen neuen anfordern zu müssen.

Zur Bestimmung, welchen Ausschnitt die Applikation anzeigen soll, benötigt man eine Ursprungsordinate. Der PDF-Viewer definiert dabei die Mitte des Dokuments als Nullpunkt. Ebenso legt die Applikation für den Ausschnitt als lokale Ursprungsordinate die Mitte fest. Dies ermöglicht eine einfachere Realisation der Funktionen zum Drehen und Navigieren. Da der Server die linke obere Ecke des gewünschten Ausschnitts erwartet, muss hier noch beim Aufruf der Server-Funktion eine Umrechnung erfolgen.

Zoom-Funktion. Die wichtigste und spannendste Funktion ist das Zoomen eines mit der Maus ausgewählten Bereiches. Zu dieser Funktion gehören



Client-Server-Kommunikation: Funktionsweise des PDF-Viewers.

eigentlich zwei ActionScript-Funktionen. Die Funktion *contentPress()* wird beim *onPress-Event* und die Funktion *contentRelease()* beim *onRelease-Event* aufgerufen.

Die Funktion *contentPress()* erfasst lediglich die Maus-Koordinaten beim *onPress-Event* und startet die Zoom-Animation.

Die eigentliche Arbeit beginnt in der Funktion *contentRelease()*. Das Zoom-Tool bietet insgesamt drei verschiedene Modi. Man kann durch Aufziehen eines Rahmens einen neuen Ausschnitt bestimmen, durch einfaches Klicken an der gewählten Position auf die nächst höhere Zoomstufe springen und durch

Drücken der STRG-Taste (Mac: Befehlstaste) zur Minus-Lupe wechseln und aus dem Bild heraus zoomen.

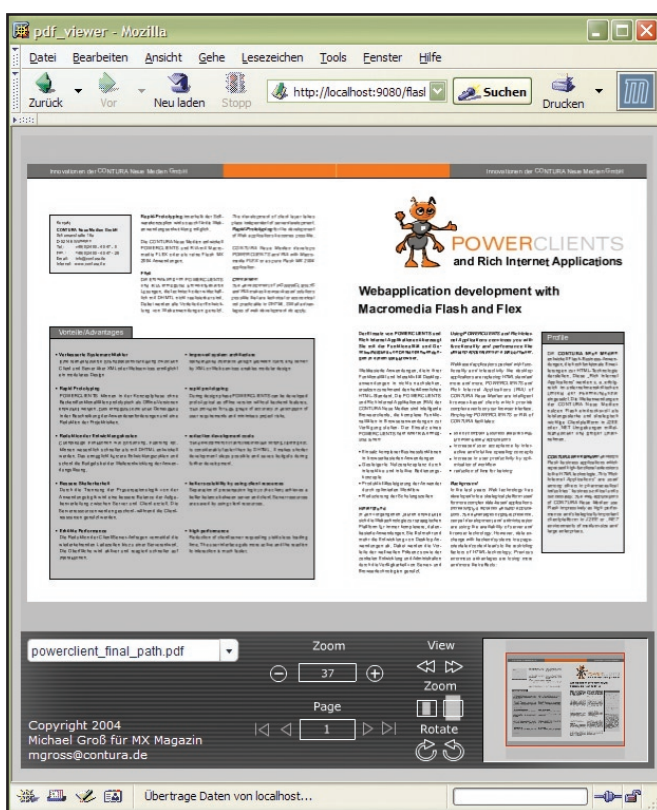
Zunächst muss man bestimmen, in welchem Modus sich das Zoom-Tool befindet. Bei einem einfachen Klick ermittelt die Hilfsfunktion *calcPos()* ein Koordinatenpaar, wobei die Funktion die derzeitige Rotation mit berücksichtigt. Daraus berechnet sich der neue Versatz, der Zoomfaktor wird auf die entsprechende Zoomstufe gesetzt.

```
var newCenter = _root.calcPos(
    _xmouse, _ymouse);
_root.xOffset += (newCenter.x
    - _root.centerX) * (100 /
    root.zoomfactor);
_root.yOffset += (newCenter.y
    - _root.centerY) * (100 /
    root.zoomfactor);
```

Wenn durch Aufziehen eines Rechtecks ein neuer Ausschnitt definiert wurde, bekommen wir richtig Arbeit: Zunächst braucht man das Delta zwischen der *onPress-Position* und der *onRelease-Position* – also wie viele Pixel in X- und Y-Richtung zwischen Klick und Release liegen. Addiert man jeweils die Hälfte davon zur linken oberen Ecke des aufgezogenen Bereichs, ergibt sich der neue Mittelpunkt des Ausschnitts.

Doch lassen Sie hier Vorsicht walten: Der Bereich kann auch von unten rechts nach oben links aufgezogen werden, also ist der Startpunkt nicht unbedingt links oben. Daher wird erst der linke obere Punkt ermittelt und diesem anschließende das halbe Delta zuaddiert.

```
var deltaX = Math.abs(
    root.zoomStartX - _xmouse);
```



PDF-Viewer in Aktion: Der Viewer zeigt vom Server in Grafiken gerenderte PDFs in Flash an.

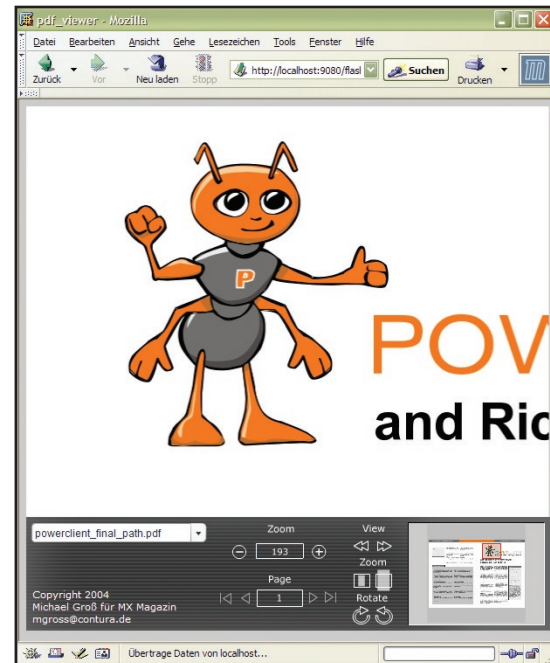
```

var deltaY = Math.abs(_
root.zoomStartY - _ymouse);
if (_xmouse < _root.zoomStartX)
{
var xpos = _xmouse + (deltaX /
2);

} else {
var xpos = _root.zoomStartX +
(deltaX / 2);
}
if (_ymouse < _root.zoomStartY)
{
var ypos = _ymouse + (deltaY /
2);
} else {
var ypos = _root.zoomStartY +
(deltaY / 2);
}

```

Kleine Ameise ganz groß:
Zoom mit fast 200%



Damit hat man den eigentlichen Klickpunkt aus dem oberen Fall und kann genauso die neuen Offset-Werte berechnen. Zusätzlich muss man noch einen neuen Zoomfaktor berechnen. Wichtig ist hier das Verhältnis zwischen aufgezogenem Zoomrechteck und Ausschnittgröße. Was heißt das? Unser Ausschnitt hat – wie bereits erwähnt – eine feste Breite von 600 Pixel. Wenn man nun ein Rechteck von 500 Pixel Breite aufzieht, so ist das Verhältnis 600 zu 500, also 1,2. Bei einem Zoomfaktor von 100% würde man, um den gewünschten Ausschnitt anzuzeigen, einen neuen Zoomfaktor von $100\% \cdot 1,2 = 120\%$, benötigen.

Beachten Sie, dass der Benutzer nicht zwingend ein Rechteck im Verhältnis der Anzeigegröße aufzieht. Er kann sogar einen Ausschnitt über die gesamte Breite des Dokuments aufziehen,

der nur ein paar Pixel hoch ist. Trotzdem soll man alles, was sich in dem Rechteck befindet, später sehen. Die nachfolgenden Zeilen berechnen den Zoomfaktor:

```

var factorX = _root.displayWidth
/ deltaX;
var factorY = _
root.displayHeight / deltaY;
if ( (deltaX / deltaY) >
(_root.displayWidth / _
root.displayHeight)) {
_root.zoomfactor = Math.round(_
root.zoomfactor * factorX);
} else {
_root.zoomfactor = Math.round
(_root.zoomfactor * factorY);
}

```

Hat man Versatz und Zoomfaktor erst mal berechnet, kann man einen neuen Ausschnitt anfordern. Da der Server nur Integer-Werte zulässt, entstehen durch Berechnungsrundungen so manchmal leichte Ungenauigkeiten, die jedoch nicht ins Gewicht fallen. Der PDF-Viewer lädt den neuen Ausschnitt mit der Funktion *GetImage* vom Server. Dabei übergibt der Parameter *crop* den Ausschnitt und der Parameter *zoom* den Zoomfaktor.

<http://localhost:9080/GetImage?file=MeinPDF.pdf&page=1&zoom=200&crop=100,100,300,300>

Hinter der Bühne. Als Server-Anwendung zum Verwalten und Rastern der PDF-Dokumente dient ein einfacher

Schnittstellen zum Server		
Funktion	Format	Beschreibung
FileList	XML	Gibt eine Liste aller PDF-Dokumente zurück, die auf dem Server abgelegt sind.
GetInfo	XML	Liefert Informationen über das PDF-Dokument und die ausgewählte Seite. Enthalten sind zum Beispiel Anzahl der Seiten, Auflösung in DPI und Seitengröße (Weite und Höhe).
GetImage	Binär	Rastert einen Ausschnitt einer Seite des PDF-Dokumentes in der angegebenen Zoomstufe. Als Resultat wird ein JPEG-Bild zurückgegeben.

Weitere Server-Funktionen	
Funktion	Beschreibung
Upload	Über die Upload-Funktion lassen sich neue PDF-Dokumente auf den Server einspielen.
Render	Mit dieser Seite können Sie den Server testen. Über die Liste wählen Sie eine Datei aus, um sie vom Server rastern zu lassen (Funktion <i>GetImage</i>). Über den Link „Info“ erhält man die Dokumenten-Informationen (Funktion <i>GetInfo</i>)



Java-Server. Der Java-Server basiert auf der Open-Source-Servlet-Engine Jetty, die sich in diesem Fall besonders durch die einfache Möglichkeit zur Integration in die eigene Anwendung anbot. So erledigt man das Initialisieren, Starten der Servlet-Engine und Registrieren der Server-Funktionen in weniger als 20 Zeilen Sourcecode. Dadurch lässt sich die Server-Anwendung direkt von der Kommandozeile aus starten, ohne dass eine Servlet-Engine lokal installiert werden muss.

Für den Flash-Client übernimmt der Server drei wichtige Aufgaben:

- ◆ eine Liste von PDF-Dokumenten zurückgeben
- ◆ Informationen über ein bestimmtes PDF-Dokument liefern
- ◆ einen Ausschnitt einer Seite eines PDF-Dokuments rastern

In den ersten beiden Fällen erfolgt die Verarbeitung eines XML-Dokuments. Auf die Verwendung von SOAP/HTTP als Webservice-Schnittstelle wurde verzichtet. Die XML-Dokumente sind einfach strukturiert und lassen sich gut mit Flash verarbeiten, so dass die Verwendung von SOAP oder Flash Remoting nicht notwendig erschien.

Für den Ausschnitt wird ein JPEG-Bild auf dem Server im Speicher erzeugt und direkt zurückgegeben. Das JPEG-Bild ist entweder ein Ausschnitt aus einer Seite im PDF-Dokument oder die gesamte Seite, wenn kein Ausschnitt angegeben wurde. In beiden Fällen ist es möglich, eine Zoomstufe anzugeben. So fordert der schon oben erwähnte Aufruf

```
http://localhost:9080/GetImage?file=MeinPDF.pdf&page=1&zoom=200&crop=100,100,300,300
```

einen Ausschnitt von Seite 1 des PDF-Dokuments *MeinPDF.pdf* an. Dabei wird auf 200% gezoomt und ein Rechteck ab Position (100,100) mit einer Kantenlänge von 300x300 Pixel im PDF-Dokument ausgeschnitten. Das resultierende JPEG-Bild hat wegen der gewählten Zoomstufe nachher die Größe 600x600 Pixel.

PDF-Dokumente rastern. Zum Rastern der PDF-Dokumente findet in diesem Beispiel die Open-Source-Bibliothek Multivalent Verwendung.

Mit Multivalent lassen sich eine Vielzahl von Dokumententypen anzeigen. Die Bibliothek enthält auch einen eigenständigen Browser zur

Darstellung der Dokumente. Im Falle des Java-Servers wird jedoch nur die Raster-Komponente für PDF-Dokumente verwendet.

Die Methode *renderPDFPage()* in der Klasse *GetImage* im Paket *de.contura.pdf-render.servlet* erzeugt für einen Ausschnitt die gerasterte Pixelgrafik. Für diese Funktionalität sind die nachfolgenden Zeilen im Java-Programm relevant:

More Info

Server starten

Minimale Voraussetzung für den Server ist ein installiertes Java 2 Runtime Environment. Wenn Sie die Sourcen auch neu übersetzen möchten, benötigen Sie allerdings das J2SE SDK. Beides – das Runtime Environment bzw. das SDK – kann man über die Java-Webseiten von SUN runterladen (<http://java.sun.com/j2se/1.4.2/download.html>).

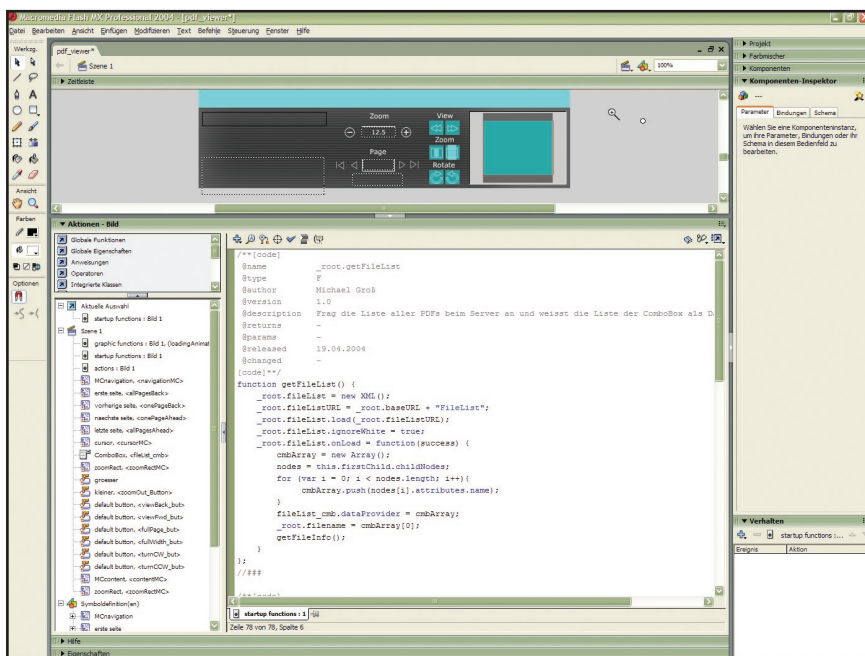
Grafikbibliothek

Die Bibliothek Multivalent benötigt eine grafische Umgebung (Windows, X-Windows, MacOS X usw.), um die Bilder rastern zu können. Dies ist eine Abhängigkeit, die auf der internen Verwendung von Java-GUI-Klassen beruht und leider zurzeit nicht umgangen werden kann. Lösungen oder Vorschläge dazu nehmen wir gerne entgegen.

CD-Dateien

Die MX Magazin-CD enthält alle notwendigen Dateien zum Starten des Servers. Im Projektverzeichnis befindet sich die Batch-Datei *startServer.bat* (für Windows) bzw. *startServer.sh* (über UNIX-Shells). Mit dem Aufruf *startserver.bat* wird der Server mit den Vorgabewerten gestartet und ist sofort einsatzbereit. Im Webbrowser öffnen Sie mit <http://localhost:9080> die Startseite. Sie können dort gleichzeitig überprüfen, ob der Server korrekt installiert wurde.

Möchten Sie eigene PDF-Dokumente rastern lassen, sollten Sie das gesamte Projektverzeichnis von der CD auf die Festplatte kopieren und den Server von dort aus starten. Die eigenen PDF-Dokumente kopieren Sie entweder direkt in das Projektverzeichnis oder Sie nutzen die Upload-Funktionen des Servers.



IDE Flash Professional MX:
Panel und Zoom-Funktion.

```

StyleSheet styleSheet =
doc.getStyleSheet();
Context ctx =
styleSheet.getContext(g, Toolkit
.getDefaultToolkit(), null);
top.bbox.setLocation(-rect.x, -
rect.y);
rect.setLocation(0,0);
top.paintBeforeAfter(rect, ctx);

```

Die dritte und vierte Zeile sind notwendig, weil das Koordinaten-System im PDF-Dokument nicht an der oberen linken Ecke beginnt, sondern unten links. Erst mit der letzten Zeile wird der Ausschnitt in das vorher erzeugte Bildobjekt gezeichnet.

Schnittstellen zum Server. Der PDF-Viewer nutzt die Funktionen in der Tabelle 1, um die PDF-Dokumente anzeigen zu können. Der URL für die einzelnen Funktionen setzt sich wie folgt zusammen: `http://<host>:<port>/<Funktion>`.

Die genaue Beschreibung der notwendigen und optionalen Parameter können Sie der Sourcecode-Dokumentation des Servers entnehmen. Alle Funktionen sind als Ableitungen

der Klasse `javax.servlet.http.HttpServlet` implementiert und im Paket `de.contura.pdfrender.servlet` zu finden.

Beispiel Schnittstelle. Informationen über das PDF-Dokument bzw. über die einzelne PDF-Seite erhält der PDF-Viewer mit dem Aufruf der Funktion `GetInfo`. Der Aufruf erfolgt über den URL `http://localhost:9080/GetInfo`.

Die Funktion `GetInfo` liefert dabei die Daten als einfaches XML-Dokument zurück. Beispielhaft sieht ein solches XML-Dokument wie folgt aus:

```

<?xml version="1.0"
encoding="UTF-8"?>
<docinfo>
  <filename>MeinPDF.pdf</filename>
  <dpi>96</dpi>
  <metric>px</metric>
  <page-count>2</page-count>
  <current-page>1</current-page>
  <page-width>794</page-width>
  <page-height>1059</page-
height>
</docinfo>

```

Das XML-Info-Dokument enthält den Dateinamen des PDF-Dokuments (`filename`) und die Anzahl der Seiten (`page-count`). Außerdem gibt es Auskunft über die aktuelle Seite (`current-page`). Für die aktuelle Seite werden die Weite (`page-width`) und die Höhe (`page-height`) zurückgegeben. Die Maßeinheit bestimmt das Tag `metric`. Die Auflösung des gerasterten Bildes gibt die DPI-Angabe (`dpi`) an.

Weitere Funktionen. Zusätzlich sind einige weitere Funktionen verfügbar. Hauptsächlich dienen diese Funktionen zum Testen des Servers, ohne dass die Flash-Anwendung aufgerufen werden muss. Alle weiteren Funktionen sind in der Tabelle 2 aufgeführt. Sie rufen alle Funktionen über `http://localhost:9080<Funktion>` auf.

Zum Speichern von neuen PDF-Dokumenten auf dem Server verwenden Sie die Server-Funktion `Upload`.

Fazit. Die vorliegende Anwendung vermittelt einen Eindruck, was ein Flash-Client in Verbindung mit einer Server-Applikation leisten kann. Der Flash-Code ist im vorliegenden Fall kompatibel mit Version 6. Durch

More Info**Server selbst kompilieren**

Wenn Sie die Sourcen selbst übersetzen möchten, benötigen Sie das J2SE SDK. Dies können Sie von der Java-Webseite von SUN kostenlos herunterladen (<http://java.sun.com/j2se/1.4.2/download.html>).

Für das Kompilieren der Sourcen verwenden Sie am besten das Build-Tool Ant. Die aktuelle Version können Sie über die Webseite (<http://ant.apache.org>) beziehen. Anschließend müssen Sie die Archiv-Datei auspacken und das bin-Verzeichnis im Suchpfad mit aufnehmen, damit die entsprechenden Shell-Skripte gefunden werden können.

Für das J2SE SDK sollte man schließlich noch eine Environment-Variable `JAVA_HOME` setzen, die auf das Installationsverzeichnis vom J2SE SDK zeigt. Ant nutzt diese Variable, um die Java-Compiler aufzurufen.

Installation erfolgreich?

Ob die Ant-Installation erfolgreich war, können Sie mit dem folgenden Aufruf testen:

```
> ant -version
Apache Ant version 1.6.1 compiled on February 12 2004
```

Es sollte die aktuelle Versionsnummer von Ant ausgegeben werden. Hat dies geklappt, steht dem Kompilieren nichts mehr im Wege. Mit

```
> ant clean compile
```

löschen Sie alle vorher erzeugten Dateien und übersetzen das Projekt neu. Den Server starten Sie anschließend mit dem folgenden Kommando:

```
> ant run
```

Viel Spaß beim Ausprobieren!

die geschickte Verknüpfung von Servertechnologien mit einem Flash-Client lassen sich Anwendungsfälle entwickeln, die auf den ersten Blick mit Flash nicht umzusetzen sind. Dies eröffnet neue Potenziale in der Web-Applikations-Entwicklung: Klassische Client-Server-Funktionalitäten sind über einen Flash-Client realisierbar.

Autor Michael Groß

Michael Groß ist als Software-Entwickler bei der CONTURA Neue Medien GmbH verantwortlich für die Realisation komplexer Business-Anwendungen auf Basis von Flash. Diese Anwendungen werden unter anderem im unternehmenskritischen Umfeld in der Pharmaindustrie eingesetzt.



Kontakt: mgross@contura.de

Autor Axel Romahn

Axel Romahn ist Software-Architekt und Java-Professional bei der CONTURA Neue Medien GmbH. Seit den Anfängen von Java dabei, hat er sich im Laufe der Jahre auf die Entwicklung und Konzeption von Java Enterprise Systemen spezialisiert. Innerhalb von Projekten ist er damit meist für die Schnittstelle zur Flash-Anwendung und die Daten-Anbindung verantwortlich.



Kontakt: aromahn@contura.de