

Jetzt auch Eclipse-basiert: Macromedias Flex 2.0

## Zweite Runde

**Dirk Eismann**

Für Überraschung sorgte Macromedia auf der diesjährigen Entwicklerkonferenz MAX in Anaheim, wo sie eine Public Alpha von Flex 2 der Öffentlichkeit zur freien Verfügung vorstellte. Denn bislang hatte der Hersteller stets geschlossene Alpha- und Betaprogramme durchgeführt. Grund genug, einen Blick auf die Neuerungen zu werfen, die die Version 2 der J2EE-Anwendung für die Entwicklung von Rich Clients bringt.

**G**leichzeitig mit der Ankündigung der Flex 2 Alpha hat Macromedia den Launch ihrer neuen Website Macromedia Labs ([labs.macromedia.com](http://labs.macromedia.com)) bekannt gegeben, auf der die Flex 2 Alpha als kostenloser Download zur Verfügung steht. In den Labs will der Hersteller künftig neue Produktversionen als Previews anbieten. Darüber hinaus finden Entwickler dort ein Wiki, über das sie eigene Artikel und Codebeispiele veröffentlichen können (siehe auch S. 24).

Die neue Flex-2-Produktlinie besteht aus der auf Eclipse basierenden

IDE Flex Builder 2, dem Framework, den Charting Components und den Enterprise Data Services, die allerdings nicht Bestandteil der Alpha sind. Im Gegensatz zur Vorgängerversion ist es mit Flex 2 möglich, Flex-Anwendungen ohne einen serverseitigen Compiler zu erstellen. Der Compiler ist Teil der IDE geworden und kompiliert Anwendungen lokal, ein Flex-Server ist für das Development nicht mehr notwendig. Macromedia erhofft sich durch diesen Schritt, Flex 2 für eine größere Nutzerschicht interessant zu machen.

Sowohl das Flex Framework als auch die API hat der Hersteller komplett überarbeitet und nach Actionscript 3 portiert, einer auf dem ECMAScript 4 Netscape Proposal basierenden objektorientierten Programmiersprache, deren Syntax jetzt stark an Java und C# erinnert. Macromedia strebt eine 100%ige Kompatibilität mit der Spezifikation an und versucht, durch Mitgliedschaft im ECMAScript-Komitee weiterhin die Entwicklung der Sprache zu steuern. Actionscript 3 bietet dem Entwickler umfangreiche Neuerungen, darunter echte Packages, benutzerdefinierte Namespaces, innere statische Klassen und Closures. Das Event-System basiert nun auf dem DOM-Eventmodel des W3C, unterstützt die Capture- und Bubbles-Phasen und ist im Gegensatz zum Vorgänger nativ in die Sprache integriert und nicht in Actionscript implementiert. Weitere wichtige neue Sprachfeatures sind E4X (ECMAScript for XML [1]), Regular Expressions sowie diverse neue Datentypen, beispielsweise *ByteArray* zur Verarbeitung von Binärdaten.

## Neuer Flash-Player als Laufzeitumgebung

Als Laufzeitumgebung für Flex-2-Anwendungen hat Macromedia eigens einen neuen Flash Player entwickelt, der bessere Performance und Stabilität bietet. Zwar hat man sich damit begnügt, dem neuen Player die Versionsnummer 8.5 zu geben, jedoch handelt es sich um eine komplette Neuentwicklung. Unter der Haube werkeln zwei voneinander getrennte Virtual Machines – in Anlehnung an Java Actionscript Virtual Machine (AVM) genannt. AVM1 ist mit dem aktuellen Flash Player 8 kompatibel und dient dazu, „alte“ Flash-basierte Anwendungen auszuführen. Die neue AVM2 hingegen ist ausschließlich zum Ausführen von Flex-2-Anwendungen beziehungsweise in Zukunft von mit der nächsten Version der Flash-Autoren-umgebung erstellten Anwendungen und Animationen gedacht.

Durch diese Trennung der beiden VMs brauchen sich die Entwickler nicht um eine Abwärtskompatibilität zu bemühen – die gibt es schlicht nicht mehr. Wichtigste Neuerung der AVM2 dürfte der JIT (Just in Time) Compiler sein, der enorme Performance-Steigerungen verspricht. Macromedia setzt hier auf eine Eigenentwicklung namens

MIR (Macromedia Intermediate Runtime), um Actionscript-3-Bytecode in plattformspezifischen Maschinencode zu überführen. Weiteres Schmankehl ist ein Bytecode Verifier, der Abstürze durch Stack Overflows verhindern soll. Endlich gibt es auch eine Fehlerbehandlung zur Laufzeit: Hat man den Debug Player installiert, zeigt dieser Laufzeitfehler in einem Dialogfenster an.

Das Flex Framework selbst setzt auf dem ebenfalls neu strukturierten Klassenmodell auf. Macromedia hat sich bemüht, die Framework-API konsistenter und feingranulierter zu gestalten. Auf den ersten Blick mag die Anzahl neu eingeführter Klassen unübersichtlich erscheinen, dies zahlt sich aber spätestens beim verbesserten Laufzeitverhalten aus. Während Flex 1.x noch zahlreiche Hacks und Workarounds seitens Macromedia verlangte, damit sich die Unzulänglichkeiten des Flash Player 7 und dessen Klassenmodells umschiffen ließen, steht den Anwendern jetzt eine verständliche und einfach zu erweiternde API zur Verfügung.

Kritische und im Projektalltag häufig wiederkehrende Workflows wie die

Entwicklung eigener Komponenten, die zur Flex-API kompatibel sind, wurden extrem vereinfacht. Macromedia hat sich außerdem der Themen Cell Renderer (benutzerdefinierte Anzeige von Zeilen beziehungsweise Zellen in Listendarstellungen) und Cell Editors (dito für editierbare Zellen/Zeilen) angenommen und die entsprechenden Schnittstellen vereinfacht. Neu hinzugekommen sind außerdem so genannte Inline Cell Renderer, über die sich Cell Renderer direkt als Child-Tag einer listbasierten Klasse, etwa einem *DataGrid*, definieren lassen. An der MXML-Syntax hat Macromedia zahlreiche Vereinfachungen vorgenommen, die dem Entwickler Schreibarbeit abnehmen. So kann er jetzt beispielsweise die einzelnen *DataGridColumn*s eines *DataGrid* direkt in der Eigenschaft *columns* deklarieren und muss nicht wie in Flex 1.x auf ein Array zurückgreifen. Über die *[DefaultProperty]*-Metadaten können Entwickler ihre eigenen Komponenten ebenfalls von unnötigen Verschachtelungen befreien.

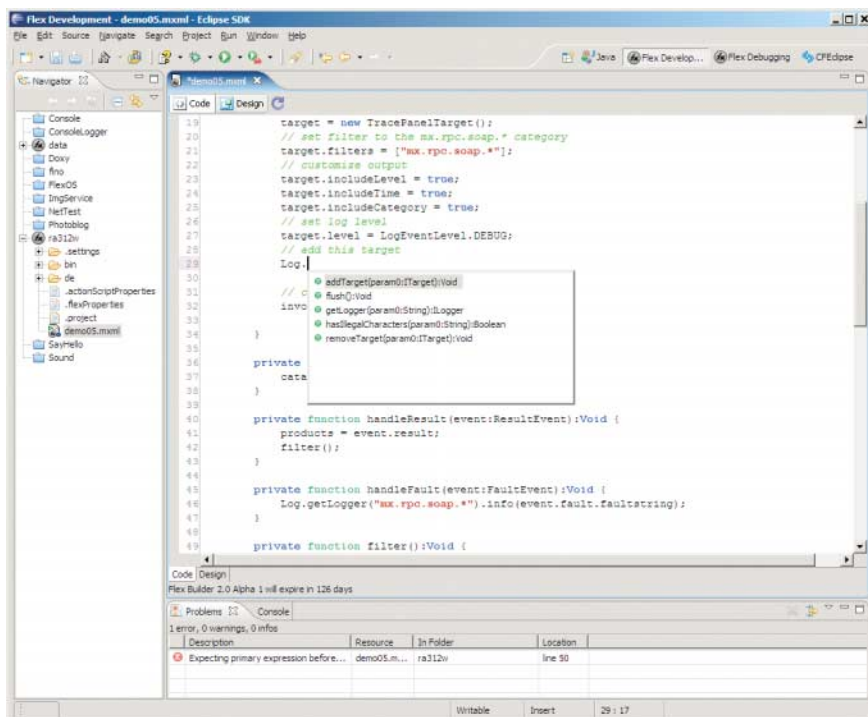
Neben dem aus Flex 1.x bekannten Container Based Layout bietet Flex 2

jetzt Constraint Based Layout. Komponenten, die sich innerhalb einer *Canvas* befinden, lassen sich über so genannte *EdgeConstraints* an den Seiten der *Canvas* verankern. Die Anzahl verschachtelter *HBox*- und *VBox*-Container lässt sich dadurch massiv verringern und das Laufzeitverhalten enorm verbessern.

## Neue Instanzen ohne Lernwege

Das dynamische Erstellen von Komponenten und Containern gestaltet sich dank der neuen Display-List-API ebenfalls wesentlich einfacher als in Flex 1.x: Neue Instanzen kann der Entwickler ohne Umwege über *createClassObject()* direkt per *new* erzeugen und per *addChild()* einer Komponente zuweisen.

So genannte View States dienen dazu, den Übergang zu beschreiben, wenn eine visuelle Komponente von einer bestimmten Darstellung in eine andere wechselt. So kann man einer einzelnen UI-Komponente verschiede-



**Im Code View kann der Anwender auf klassische Features wie Code Hinting und Auto Completion zählen (Abb. 1).**

ne (An-)Sichten zuweisen. Eine einfache Login-Maske beispielsweise kann zwei View States besitzen. Neben dem Standard View State, der lediglich ein Formular zur Eingabe von Benutzernamen und Passwort enthält, stellt ein weiterer ein zusätzliches Feld zur Eingabe der E-Mail-Adresse bereit, etwa um ein vergessenes Passwort anzufordern. Alle nötigen Änderungen, um vom Standard View State in den „abgeleiteten“ zu gelangen, beschreibt der Entwickler über die View-State-API. Als Eyecatcher kann er den Übergang zwischen den View States animieren (Resize, Zoom et cetera).

Zu den bekannten UI Controls von Flex 1.x gesellen sich in der Version 2 einige neue, darunter *PopUpButton*, *ApplicationControlBar* und *RichTextEditor*. Die Flex Charting Components wurden ebenfalls überarbeitet und bieten einige neue Diagrammtypen und -features. Die Komponenten werden in der finalen Version nicht Teil des Flex Frameworks sein. Sie muss man separat erwerben.

Um eine Anwendung optisch an beispielsweise das Corporate Design anpassen zu können, hat Macromedia das so genannte Styling und Skinning enorm vereinfacht. Musste man in Flex 1.x noch umständlich die Grafikressource in Flash bearbeiten und als so genanntes Theme in Flex importie-

ren, lassen sich nun alle visuellen Elemente der UI-Komponenten per CSS steuern – entweder über eine externe CSS-Datei, die zur Kompilierzeit eingebunden wird, oder über das Style-Element. Durch Einsatz der aus Flash 8 übernommenen Scale9-Technik skalieren benutzerspezifische Skins nun ebenfalls korrekt.

Weitere Sorgenkinder wie Validatoren und die Effect-API haben ebenfalls eine Überarbeitung erfahren. Validatoren lassen sich dynamisch per Actionscript instanzieren und sind wiederverwendbar. Hilfreich ist auch eine Konfiguration per *RegExp*, um bestimmte Eingabemuster herauszufiltern. Das Handling der Effekte wurde ebenfalls verbessert. Neben den aus Flash 8 bekannten Bitmap-Effekten wie Blur und Glow lassen sich komplexe kombinierte Sequenzen einfach und effizient per MXML oder Actionscript umsetzen.

## Eclipse statt Dreamweaver

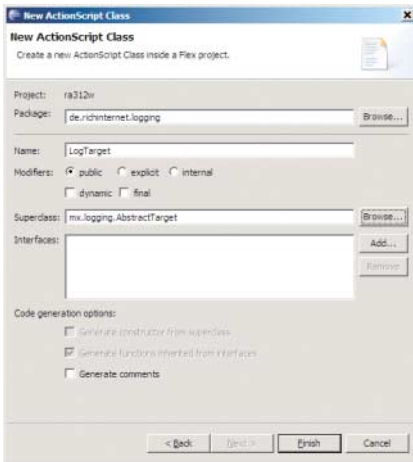
Basierte Flex Builder 1.5 noch auf Dreamweaver MX 2004, so hat Macromedia erfreulicherweise den Schritt gewagt, Flex Builder 2 auf Basis von Eclipse 3.1 zu entwickeln. In der aktuellen Alpha ist die IDE als Standalone oder als Plug-in verfügbar. Ers-

tere soll Nutzern ohne Eclipse-Erfahrung den Einstieg erleichtern. Letzteres ist an J2EE-Entwickler gerichtet, die auch mit Flex arbeiten möchten.

Wie sein Vorgänger bietet Flex Builder 2 einen Code View und einen Design View (siehe Aufmacher sowie Abbildung 1) und erweitert Eclipse um die Projekttypen „Flex Project“. Neben der Flex Development Perspective, die diese beiden Views enthält, gibt es die Flex Debugging Perspective, die einen umfangreichen Debugger zur Verfügung stellt. Für die Erstellung neuer MXML-Komponenten gibt es verschiedene Wizards (Abbildung 2). Der zur Erstellung von neuen Actionscript-Klassen erinnert an den Class Wizard der Eclipse JDT und erlaubt neben der Auswahl der Superklasse die Angabe zu implementierender Interfaces. Der Code-Editor verfügt über klassische Features wie Code Hinting und Auto Completion. Das Code Hinting arbeitet gut und löst Objekte auch in MXML Bindings auf. Ein Outline View für Actionscript- und MXML-Klassen fehlt in der Alpha noch, soll aber genauso wie Code Collapsing in der finalen Version verfügbar sein. Insgesamt bietet der Code-Editor viele Features, die man von einer IDE gewohnt ist.

Der Design View bietet einen WYSIWYG-Editor für die visuelle Erstellung des Layouts einer Anwendung. In der aktuellen Alpha ist dieser Editor zwar noch speicherhungrig, bietet insgesamt aber einen guten Workflow. Über Drag & Drop lassen sich Komponenten und Controls in der Anwendung platzieren, ein Property Inspector erlaubt das einfache Editieren der einzelnen Attribute. Der Inspector bietet eine kontextsensitive Sicht auf die Eigenschaften und lässt sich auf eine umfangreiche Listendarstellung umschalten, in der er weitere Parameter bereitstellt. Die oben besprochenen *EdgeConstraints* kann der Anwender ebenso wie View States bequem im Design View anlegen. Ihm sieht man anders als dem Code View den Alpha-Status an – es gibt noch viele kleine Bugs und Unschönheiten.

Ebenfalls im Rahmen der MAX hat Macromedia die Flex 2 Enterprise Data Services ([labs.macromedia.com/wiki/index.php/Flex\\_Enterprise\\_Services:Data\\_Services](http://labs.macromedia.com/wiki/index.php/Flex_Enterprise_Services:Data_Services)) präsentiert, die allerdings nicht Teil der Alpha sind und erst zu einem späteren Zeitpunkt verfügbar sein werden. Die Data Services bieten eine datenorientierte Alternative zu den aus Flex 1.x bekannten serviceorien-



### Wizards helfen bei der Erstellung neuer MXML-Komponenten (Abb. 2).

tierten RPC-Diensten (*WebService*, *RemoteObject*, *HTTPService*). Sie erlauben Datensynchronisation, Kollaboration in Echtzeit und Konfliktbehandlung. Die Integration in eine bestehende Infrastruktur soll über verschiedene Adapter (JDBC, Hibernate) vonstatten gehen. Eine Unterstützung von .Net in Form eigener Data Services oder die

Anbindung an .Net Assemblies wird es in naher Zukunft leider nicht geben. Die nach wie vor einzige Möglichkeit, mit einem .Net-Backend zu kommunizieren, ist über SOAP.

Endgültige Preise und Termine für die Final Release von Flex 2 sind noch nicht bekannt. Offiziellen Aussagen Macromedias zufolge wird Flex Builder 2 zusammen mit dem Flex 2 Framework weniger als 1000 US-\$ kosten. Die Preise für die Flex 2 Enterprise Data Services werden sicherlich wie die für Flex 1.x im fünfstelligen Bereich angesiedelt sein. Laut Macromedia soll Flex 2 im ersten Halbjahr 2006 als Final Release verfügbar sein.

### Fazit

Der erste Eindruck von Flex Builder 2 entschädigt für die vielen verzweifelten Stunden, die man vor Flex Builder 1.5 verbracht hat. Neben der Wahl von Eclipse als Grundlage für die neue IDE dürfte das neue Preismodell Flex für mehr Entwickler attraktiv machen.

Mit Flex 2 hat Macromedia einen großen Schritt in Richtung einer soli-

den, performanten Plattform für die Entwicklung browserbasierter Rich Clients getan. Angesichts des deutlich zu vernehmenden Störfeuers aus Redmond namens Microsoft Expression auf die gemeinsamen Claims von Macromedia und Adobe ([www.microsoft.com/products/expression/en/interactive\\_designer/default.aspx](http://www.microsoft.com/products/expression/en/interactive_designer/default.aspx)) dürfte die Entwicklung in diesem Marktsegment interessant verlaufen. Es bleibt abzuwarten, wie viel von der neuen, offenen Informationspolitik unter dem Namen Adobe Bestand haben wird. (ka)

### DIRK EISMANN

arbeitet als Software Engineer bei der Herrlich & Ramuschkat GmbH in Hannover. Sein Hauptaufgabengebiet ist die Entwicklung von Rich Clients mit Flex.

### Literatur

- [1] Stefan Mintert; Ein E für ein X; E4X: Ecmascript/Javascript for XML; iX 11/04, S. 60