

Eclipse ist auch in der Welt der Design-Programmierer zu Hause

# Sonnenfinsternis bei Adobe und Macromedia

■ VON PHILIPP GIELEN UND KAI KÖNIG

Die Eclipse-Plattform ist längst über die Java-Welt hinausgewachsen und zu einer populären Entwicklungsplattform auch für andere Technologien geworden. Wir zeigen, wie sich Webentwickler Eclipse für ihre tägliche Arbeit mit den Technologien von Adobe (ehem. Macromedia) zunutze machen können. Auch nach der Fusion von Adobe und Macromedia liegt der Produktfokus auf Tools und Serverlösungen zur Webentwicklung.

Eines vorneweg: Um unschöne Sprachkonstrukte zu vermeiden, verweisen wir in diesem Artikel weiterhin auf Macromedia als Hersteller der entsprechenden Produkte – die aktuellen Versionen werden immerhin noch unter diesem Label vertrieben. Die Macromedia-Kernprodukte Dreamweaver und Flash orientieren sich auffällig an den Bedürfnissen von Designern oder generell an visuellen Arbeitsprozessen. Interessant ist in diesem Zusammenhang ein Schwenk in der Macromedia-Strategie besonders im Bereich der Entwicklung des Produktes ColdFusion. Nach Erscheinen von ColdFusion MX (6.0) verschwand die Entwicklungsumgebung ColdFusion Studio vom Markt – einige der CF-spezifischen Features wurden dabei in HomeSite+ 5.1 bzw. 5.5 überführt. HomeSite+ kann man allerdings nicht einzeln beziehen, sondern bekommt es nur als Dreingabe zu einem Kauf von Dreamweaver MX bzw. MX 2004 oder Dreamweaver 8. In HomeSite+ werden viele Features der neuen ColdFusion-Versionen jedoch nicht ordentlich unterstützt, so zum Beispiel das Browsen von ColdFusion Components, die Reporting Engine und vieles andere mehr. Die Idee dieser Vorgehensweise ist klar:

Macromedia wollte Dreamweaver als Entwicklungswerkzeug für CF-Entwicklung etablieren. Rückblickend hat dieser Ansatz nicht funktioniert. Dreamweaver ist sicherlich ein sehr gutes Werkzeug für Webdesign und -entwicklung mit nur geringer serverseitiger Anbindung, hat aber einige Probleme im Zusammenhang mit großen und sehr codelastigen ColdFusion-Projekten. Hinzu kommt, dass viele Coder die Möglichkeiten zur visuellen Entwicklung mit Dreamweaver überhaupt nicht nutzen bzw. gar nicht nutzen wollen. Eine ähnliche Situation findet man bei anderen serverseitigen Technologien, die von Dreamweaver unterstützt werden, seien es nun PHP, ASP, ASP.NET oder JSP. In der Regel weichen Coder und/oder Backend-Entwickler auf andere Entwicklungsumgebungen aus.

Ähnliches gilt für Flash, das auch in der aktuellen Version 8 seine Herkunft als Animationswerkzeug nicht verleugnen kann. Die Möglichkeiten für codezentriert arbeitende Entwickler sind gerade für professionelle Arbeiten deutlich eingeschränkt, sodass IDEs von Drittanbietern (zum Beispiel PrimalScript oder Sephy) heute eine sehr gute Position am Markt haben.

An dieser Stelle rückte Eclipse aufgrund der stetigen Verbesserung und Weiterentwicklung immer stärker ins Bewusstsein von Nicht-Java-Entwicklern – es war nur eine Frage der Zeit, bis sich sinnvolle Eclipse-Plug-ins für Macromedia-Technologien finden würden. Im weiteren Verlauf dieses Artikels werden wir auf einige der wichtigsten und interessantesten Plug-ins eingehen und zeigen, wie man sich als Macromedia-lastiger Entwickler eine sehr komfortable Entwicklungsumgebung zusammenstellen kann, die auf Eclipse und anderen frei verfügbaren Werkzeugen beruht.

## CFEclipse

Bei CFEclipse [1] handelt es sich um ein Open-Source Plug-in, das aus den Kreisen der ColdFusion-Entwickler-Community stammt, die sich aus den oben erwähnten Einschränkungen der von Macromedia bereitgestellten Entwicklungswerkzeuge eine eigene Alternative geschaffen hat. CFEclipse wird im Rahmen eines eigenen Artikels in der Plug-in-Parade in diesem Heft gesondert vorgestellt und sei daher hier nur kurz erwähnt (siehe Seite 55).

CFEclipse birgt eine sehr interessante Situation, denn Macromedia hat sich auf

der CFUnited-Konferenz im Juli 2005 sehr klar zur Unterstützung der Weiterentwicklung von CFEclipse als unabhängiges Projekt bekannt und angekündigt, dem Kernentwicklerteam Zugang zu verschiedenen Ressourcen zu gewähren. Es bleibt zu hoffen, dass dieser sehr positive Umgang mit dem Open-Source-Projekt nach der Übernahme durch Adobe weiter fortgeführt wird.

## SQLExplorer und Clay

Eine typische ColdFusion-Applikation greift in der Regel auf externe Datenquellen zurück, um die erstellte Webapplikation dynamisch mit Daten befüllen zu können. In der vorliegenden Version von CFEclipse (1.2) ist ein Browsen von genutzten Datenbanken oder Datenbank-Objekten noch nicht möglich, allerdings für die Zukunft geplant. Daher benötigt der ambitionierte Entwickler einer Webapplikation mit ColdFusion entweder ein eigenes Tool, um Datenbanken und/oder ihre Inhalte verwalten zu können – die meisten Datenbank-Anbieter liefern solche Tools bereits mit aus – oder muss auf der Suche nach einer Eclipse-Integration nach einem passenden Plug-in Ausschau halten.

Die perfekte Ergänzung zu CFEclipse ist das SQLExplorer-Plug-in. Es bietet für die meisten JDBC-fähigen Datenbanken Funktionen zum Ausführen von Quers und Darstellen von Tabellen. Mithilfe des SQLExplorer-Plug-ins wird Eclipse aus Sicht eines ColdFusion-Entwicklers endgültig zum vollwertigen Ersatz für Tools wie HomeSite oder Dreamweaver.

Das Plug-in kann unter [2] heruntergeladen werden. Die Konfiguration ist gewöhnungsbedürftig, aber wenn man es geschafft hat, eine Verbindung zum Datenbank-Server aufzubauen, ist alles sehr intuitiv handhabbar. Eine detaillierte Installationsanleitung würde allerdings den Rahmen dieses Artikels sprengen. Möchte man die Integration noch weitertreiben, bietet sich das Plug-in Clay [3] an. Mit dessen Hilfe lassen sich komplette Modellierungsprozesse für relationale Datenbanken in Eclipse durchführen.

## Flash-Alternativen

Neben ColdFusion und der mit ColdFusion in der Regel verbundenen Ar-

Abb. 1: Das CFEclipse-Plug-in für die Entwicklung mit ColdFusion in Aktion

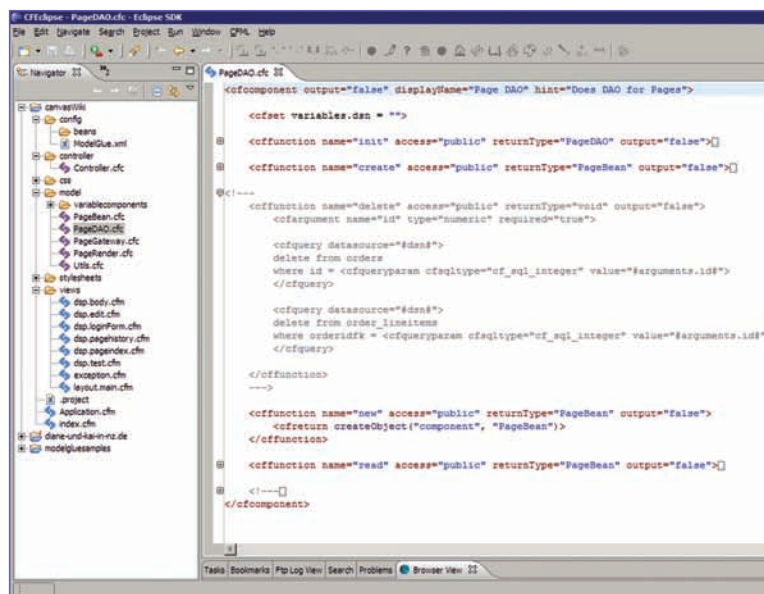
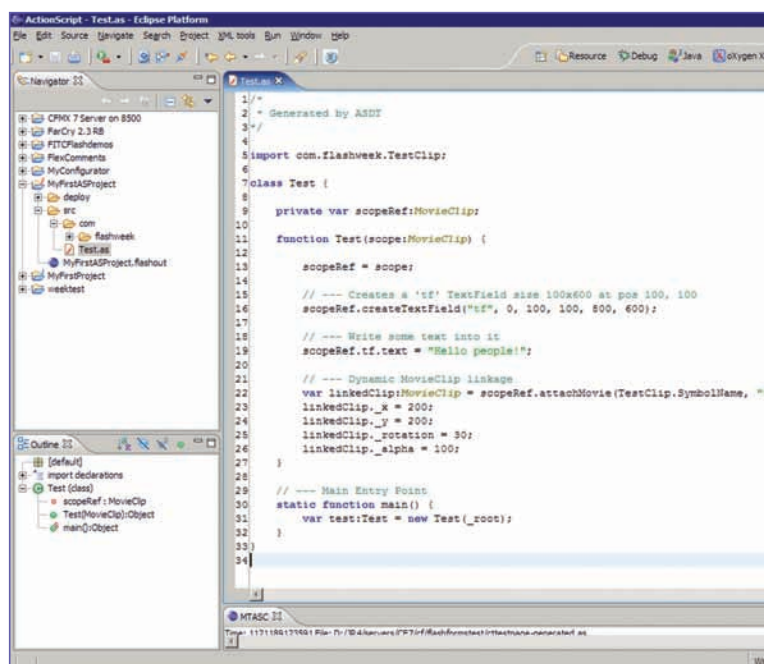


Abb. 2: ActionScript-Entwicklung mit dem frei verfügbaren ASDT



beit mit Datenbanken lässt sich Eclipse inzwischen auch zur Entwicklung von ActionScript-, Flash- und Flex-Applikationen nutzen. Gerade in diesem Umfeld hat sich im Laufe dieses Jahres enorm viel getan, sodass es sich lohnt, einen Blick auf entsprechende Plug-ins zu werfen.

Für Flash-Nutzer bietet sich seit einiger Zeit die Installation des AS2-Plug-ins ASDT [4] von Peter Schreiber an. Entstanden ist dieses Open-Source-Projekt am Rande eines großen Integrationsprojektes mit Flash und Java, als deutlich wurde, dass die Möglichkeiten der Flash-

Authoring-Umgebung für professionelle Enterprise-Projekte bei weitem nicht zufriedenstellend sind. Beschränkten sich die ersten Versionen des Plug-ins noch auf Code-Highlighting, sind im Laufe der Zeit immer mehr Wizards und Anbindungsmöglichkeiten an externe Projekte hinzugekommen. So bietet das ASDT seit einigen Versionen die Möglichkeit, den externen – und ebenfalls frei verfügbaren – AS2-Kommandozeilen-Compiler MTASC [5] anzusteuern. Die Integration zwischen beiden Projekten ist sehr einfach und gelingt reibungslos.

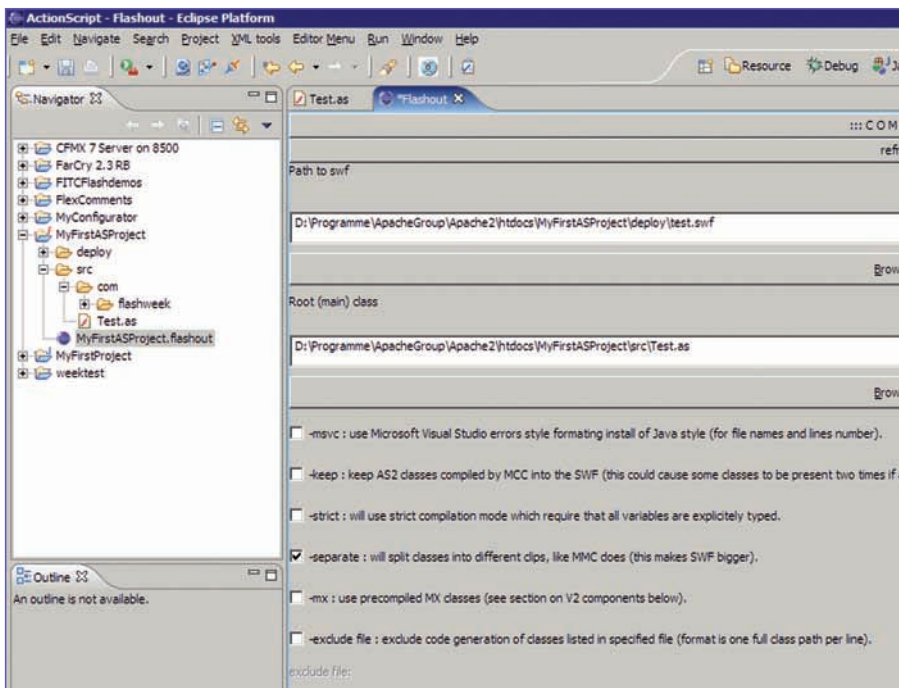


Abb. 3: Flashout integriert sich als Runtime-Umgebung für .swf in Eclipse

Die von MTASC erzeugten Kompilate entsprechen dem, was Macromedia mithilfe der Flash-Authoring-Umgebung als so genannte SWF-Dateien erzeugt. swfmill [6] ist ein Werkzeug, das es ermöglicht, mit den SWF-Dateien zu arbeiten und XML-Daten in SWF-Dateien zu wandeln und umgekehrt. In einem Entwicklungs-Workflow hat swfmill in der Regel die Aufgabe, die typischen Ressourcen eines Flash-Projekts zusammenzufügen: die Ergänzung des kompilierten Codes durch mit swfmill definierte visuelle Bibliotheken.

In diesem Kontext ist ebenfalls Flashout [7] zu nennen. Mit Flashout wird es abschließend sogar möglich, die generierte und kompilierte SWF-Applikation in Eclipse auszuführen und somit das Ergebnis der Entwicklungsarbeit zu sehen. Flashout ist zwar frei verfügbar, aber nicht quelloffen – doch trotzdem einen Blick wert.

Ein sehr interessantes Werkzeug – leider nicht als Plug-in für Eclipse verfügbar – ist ArgoUML [8] und die Erweiterung ArgoUMLASGenerator [9]. Anhand von ArgoUML lassen sich die üblichen Modellierungsaufgaben im Rahmen der Architekturerstellung eines Softwareproduktes durchführen und mit der Er-

weiterung Code-/Klassenfragmente für ActionScript 2 erzeugen.

### Zukunft von Flash und Open Source

Langfristig wird das Thema Flash Open Source sicher immer interessanter werden. Von Open-Source-Evangelisten als Manko angesehen wird die Tatsache, dass MTASC und Flashout nach wie vor auf die mit der Flash-Authoring-Umgebung mitgelieferten Klassenbibliotheken zurückgreifen müssen. Letztlich und in einem formalen bzw. lizenzrechtlichen Sinn ist dieser Ansatz daher wohl nicht in der Gesamtheit als offen zu bezeichnen. Entwickler, die auf der Suche nach einer zukunftssträchtigen und codezentrierten Entwicklungsumgebung für Flash sind, werden mit dem hier beschriebenen Ansatz jedoch bestimmt viel Spaß haben.

### Kommerzielles Flash: FDT

Die in Aachen ansässige Firma Powerflasher hat im Spätsommer ein kommerzielles ActionScript-Plug-in namens FDT veröffentlicht [10] [11]. Das Plug-in erinnert auf den ersten Blick an die Möglichkeiten des frei verfügbaren ASDT, wurde von den Powerflashern jedoch um viele Wizards und Möglichkeiten zur integrierten Projektabwicklung ergänzt. Die Re-

sonanz der Flash Community auf FDT ist vergleichsweise gut, auch wenn der Preis von 199 Euro für eine Einzelplatzlizenz im Flash-Umfeld erstaunlich hoch ausgefallen ist.

### Flex 1.5 und Eclipse

Zur Entwicklung von Flex-Applikationen in der vorliegenden Version 1.5 des Serverproduktes setzt Macromedia auf ein eigenes Produkt namens Flex Builder. Als Alternative zum Flex Builder bieten sich bereits heute verschiedene kommerzielle oder freie Entwicklungsumgebungen an. Ein sehr gutes Feature von Flex Builder ist der Design-Modus, der es erlaubt, die Oberfläche der Flex-Anwendung aus Komponenten zusammenzustellen und den dazu benötigten Code im Hintergrund zu erzeugen. Benötigt man diese Möglichkeit nicht, leidet Flex Builder generell unter den gleichen Mängeln wie Dreamweaver, da es auf der gleichen internen Engine beruht.

Im Falle von Flex zeigte sich Macromedia jedoch deutlich offener und hat sich dazu entschieden, die deklarative Sprache MXML auf Basis eines XML Schema zu definieren. Diese Schemabeschreibung wird mit dem Flex Presentation Server 1.5 mitgeliefert und lässt sich generell in jede XML Schema-fähige Entwicklungsumgebung importieren. Mit erfolgreichem Import stehen Entwicklern in der Regel Syntax-Highlighting und Code Completion zur Verfügung.

Eclipse als solches bietet keine direkte XML-Unterstützung. Als sehr gutes Plug-in für die Flex-Entwicklung hat sich Oxygen XML [12] erwiesen. Bei Oxygen handelt es sich um ein kommerzielles Produkt, das preislich mit ca. 100 bis 150 US-Dollar (je nach Lizenztyp) absolut angemessen für seine Möglichkeiten ist. Letztendlich unterstützt Oxygen XML nicht nur XML Schema und somit die Flex-Entwicklung, sondern ist ein Plug-in, das mit nahezu jedem Typ von XML-Dokument zurechtkommt und sich nahtlos in die Arbeit an Webapplikationen einfügt.

Eine sehr gute und kostenfrei erhältliche Alternative bieten das Eclipse-eigene Web Standard Tools-Projekt [13]. Neben Unterstützung für Technologien und Sprachen wie (X)HTML, JavaScript und

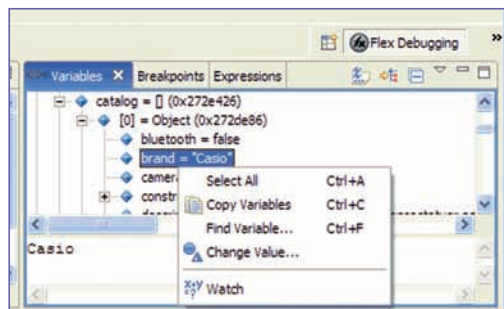


Abb. 4: Debug View in Flex 2.0



Abb. 5: Flex Builder 2.0 unterstützt Code Completion

viele weitere Standards lässt sich nach einigen wenigen Konfigurationsschritten auch Flex-Code mit der WST-Umgebung entwickeln. Eine sehr ausführliche Anleitung zur Konfiguration findet man im Blog von Darron Schall [14].

Es bleibt festzustellen, dass aus Sicht eines Flex-Entwicklers die WST-Umgebung den Möglichkeiten von OxygenXML in nichts Wesentlichem nachsteht, sodass gerade unter Gesichtspunkten der Weiterentwicklung und Zukunftssicherheit die Umstellung auf WST eine gute Empfehlung darstellen sollte.

### Flex 2.0 – Flex Builder wird Eclipse

Das zurzeit in einer frei verfügbaren Beta-Version [15] vorliegende Flex 2.0 geht einen anderen Weg. Macromedia hat auf die Anregungen seiner Entwicklergemeinde gehört und setzt mit der Version 2.0 direkt auf die Eclipse-Technologie. Der Flex Builder 2.0 zeigt sich in der momentan verfügbaren Version als Plug-in für eine bestehende Eclipse 3.1-Installation oder als Komplett-Download im Paket für diejenigen User, die bislang noch nicht mit Eclipse gearbeitet haben.

Das Plug-in erweitert Eclipse um einen eigenen Flex-Projekttyp und bietet neben der klassischen Code View mit Features wie Code Hinting und Auto-Completion auch eine Design View. In dieser lassen sich nach schöner WYSIWYG-Manier Applikationslayouts visuell zusammensetzen, der Flex Builder generiert im Hintergrund den Code.

Des Weiteren gibt es eine Flex-Debug-Perspektive, die umfangreiche Möglichkeiten zum Tracen und Debuggen von Applikationen bereitstellt. Dazu greift sie genau wie die Development-Perspektive auf die vom Flex 2-Compiler angebotenen Dienstleistungen zurück, sodass die

Arbeit in der IDE nahtlos in den Kodierungsprozess eingebunden ist.

### LogViewer für ColdFusion

Zum Schluss noch ein weiteres Schmankehl für die Fans von ColdFusion. Mike Nimer, ein Mitglied des ColdFusion-Entwicklungsteams, hat ein Eclipse-Plug-in zum Anzeigen der ColdFusion-Logfiles [16] entwickelt und auf seiner Webseite zum Download bereitgestellt. Der Quellcode ist dort ebenfalls erhältlich und lädt dazu ein, das kleine Plug-in eigenständig zu erweitern und auch auf andere Log-Formate umzustellen.

### Fazit

Während Macromedias Entwicklungstools Dreamweaver und Flash eher auf die Zielgruppe der Designer-Programmierer fokussiert sind, spricht Eclipse gezielt professionelle Entwickler an, die ihren Code eher von Hand schreiben als Komponenten zusammensetzen.

Die Möglichkeiten für Macromedia-nahe Entwickler mit Eclipse komfortabel zu arbeiten, haben sich in der letzten Zeit deutlich verbessert. Plug-ins wie CFEclipse, die aus der reinen Community herausgewachsen sind und nun offiziell von Macromedia unterstützt werden, zeigen, dass der Hersteller unserer Entwicklungsplattformen einmal mehr auf Forderungen und Anregungen der Community reagiert und die Zeichen der Zeit erkannt hat. Dennoch hat auch Eclipse noch einige Schwachstellen, die der Ausbesserung bedürfen, wie z.B. nur unzureichend umgesetzte FTP-Plug-ins. Wenn es also auch in absehbarer Zukunft nicht das ultimative ColdFusion-Kodiertool aus dem Hause Macromedia geben wird, verschaffen die jüngsten Entwicklungen doch wieder Grund zur Hoffnung auf Besserung.

Eclipse genießt als offene und flexibel erweiterbare Entwicklungsplattform weltweit in der Entwickler-Community einen sehr guten Ruf. Ein Grund mehr, die Stellung der Macromedia-Client- und Servertechnologien durch gut gelungene Plug-ins und Erweiterungen zu unterstreichen. Letztlich ist ein professionelles Eclipse-Plug-in für eine Programmiersprache eine Visitenkarte, mit der man gut hausieren gehen kann.



**Kai König** ist Macromedia Certified Professional und Master Instructor, lebt in Wellington, Neuseeland, und arbeitet dort für ZeroOne als Digital Solutions Architect. Kai bloggt über sein Leben mit Macromedia-Produkten als AgentK im Blog in Black ([kai@bloginblack.de](mailto:kai@bloginblack.de), [www.bloginblack.de](http://www.bloginblack.de), [www.zeroone.co.nz](http://www.zeroone.co.nz)).



**Philipp Cielen** ist Geschäftsführer der Agentur cielen.com, die sich auf Anwendungsentwicklung im Bereich Macromedia-Technologien spezialisiert. Philipp ist Co-Autor des deutschsprachigen Buchs „ColdFusion MX“, Macromedia-zertifizierter Entwickler und leitet die CFUG Frankfurt. ([philipp@cielen.com](mailto:philipp@cielen.com), [www.cielen.com/blog](http://www.cielen.com/blog), [www.coldflash.de](http://www.coldflash.de))

### Links & Literatur

- [1] CFEclipse: [www.cfeclipse.org](http://www.cfeclipse.org)
- [2] SQL Explorer: [sourceforge.net/projects/eclipsesql](http://sourceforge.net/projects/eclipsesql)
- [3] Clay: [www.azzurri.jp/en/software/clay/](http://www.azzurri.jp/en/software/clay/)
- [4] ASDT: [sourceforge.net/projects/aseclipsePlug-in/](http://sourceforge.net/projects/aseclipsePlug-in/)
- [5] MTASC: [www.mtasc.org](http://www.mtasc.org)
- [6] Swfmill: [www.swfmill.org](http://www.swfmill.org)
- [7] Flashout: [www.potapenko.com/flashout/](http://www.potapenko.com/flashout/)
- [8] ArgoUML: [argouml.tigris.org](http://argouml.tigris.org)
- [9] ArgoUMLASGenerator: [www.codealloy.com/argoumlforactionsript.htm](http://www.codealloy.com/argoumlforactionsript.htm)
- [10] FDT: [fdt.powerflasher.com](http://fdt.powerflasher.com)
- [11] Nico Zimmermann: FDT – Development Tool for Flash, in *Eclipse Magazin* Vol. 5
- [12] OxygenXML: [www.oxygenxml.com](http://www.oxygenxml.com)
- [13] WST: [download.eclipse.org/webtools/downloads/](http://download.eclipse.org/webtools/downloads/)
- [14] Blog von Darron Schall: [www.darronschall.com/weblog/archives/000182.cfm](http://www.darronschall.com/weblog/archives/000182.cfm)
- [15] Adobe Labs: [labs.macromedia.com](http://labs.macromedia.com)
- [16] CF Logfile Viewer: [www.mikenimer.com/eclipse/logviewer/](http://www.mikenimer.com/eclipse/logviewer/)