

Taking ColdFusion Applications Off-line with Adobe Flex and AIR



Silver
Solution Partner
EMEA



AUTHORISED
Training Centre

Sven Ramuschkat





ColdFusion 9 Offline AIR Application Support

Adobe AIR® local/remote database synchronization **NEW**

Offer users applications with online/offline capability featuring reliable data management. Using SQLite in the client and ORM on the server, ColdFusion 9 manages conflict resolution and data synchronization between the client and server when your application comes back online.

A screenshot of a code editor window titled 'AIREmployee.cfc'. The code is written in a dark font on a light background. The code includes a comment and two lines of code: 'ArrayNew(1)' and 'ityLoad("employee")'. A blue play button icon is visible in the bottom right corner of the code editor window.

```
AIREmployee.cfc  
  
ab  
  
"CFIDE.AIR.ISyncManager">  
"access="remote" return  
ArrayNew(1)>  
ityLoad("employee")>
```

Adobe AIR 1.5 Capabilities

Cross-OS
Application

Integrated
Rendering

Integrated
DOMs and
Scripting



Flash

Flex

ActionScript™

XML

Audio

Video



HTML



PDF



HTML

HTML

JavaScript

XML

CSS



Flash



PDF

Adobe AIR
APIs

File System
Access

Network
Detection

Notifications

Application
Update

Drag and
Drop

**Local
Database**

Mac, Windows, Linux

SQLite Limits



- Max. number of Columns is 32.767 Bytes, max. number of Index Columns is 32.767 Bytes
- Maximum Length Of An SQL Statement is 1.073.741.824 Bytes
- The current SQLite implementation will only support a string or BLOB length up 2.147.483.647 Bytes
- SQLite does not support joins containing more than 64 tables
- Max. Number of Database Pages: 1.073.741.823
- Maximum Database Page Size: 32.768
- Performance
 - SQLite 2.7.6 is significantly faster (sometimes as much as 10 or 20 times faster) than the default PostgreSQL 7.1.3 installation on RedHat 7.2 for most common operations.
 - SQLite 2.7.6 is often faster (sometimes more than twice as fast) than MySQL 3.23.41 for most common operations.

ColdFusion Offline AIR Application Support

- nur für Flex Applikationen auf Basis von AIR
- Enthält ein ActionScript Persistence Framework (1), welches folgende Funktionalitäten bietet:
 - Creating and maintaining the SQLite database
 - Saving objects to SQLite without SQL statements
 - Handling relationships:
 - one-to-one
 - one-to-many
 - many-to-one
 - many-to-many
 - Syncing those records back to ColdFusion

(1) kommt aus der *cfair.swc* Lib in *C:/ColdFusion9/wwwroot/CFIDE/scripts/AIR/cfair.swc*

Die Backend ColdFusion Application

- Application.cfc

```
<cfcomponent>
  <cfset this.name = "OneTonOnefdExample">
  <cfset this.datasource="AIROffline">
  <cfset this.ormenabled="true">
  <cfset this.ormsettings={dialect = "MicrosoftSQLServer"}>
</cfcomponent>
```

Definieren von ColdFusion ORM-CFCs

Customer.cfc

```
<cfcomponent persistent="true">
    <cfproperty name="cid" fieldtype="id" >
    <cfproperty name="companyname" >
    <cfproperty name="address" fieldType='one-to-one'
        CFC="address" fkcolumn='aid' cascade='all' >
</cfcomponent>
```

Address.cfc

```
<cfcomponent persistent="true">
    <cfproperty name="aid" fieldtype="id" >
    <cfproperty name="street" >
    <cfproperty name="zip" >
    <cfproperty name="city" >
</cfcomponent>
```



Testen der ORMs

cusManager.cfc (1)

- Sync-Manager CFC which the AIR Offline app client interacts with.
- This Manger CFC will have to implement “CFIDE.AIR.ISyncManger”.
- The component has two functions:
 - A fetch function that the AIR applica ColdFusion.
 - A sync function that the AIR applica ColdFusion and AIR data sources

originalObjects: An array of objects, each item in this Array represents the corresponding original data from the server (if any), such as an existing employee record that a user is updating.

clientObjects: An array of objects, where each item in the array represents the client's current view of the data to be synchronized.

operations: An array of operations to perform INSERT, UPDATE, or DELETE.

```
<cfcomponent implements="CFIDE.AIR.ISyncManager">

<!---Fetch method--->
<cffunction name="fetch" returnType="Array" access="remote">
  <cfset cus = ArrayNew(1)>
  <!---This ORM Method will load all data from Server Customer table and send it to AIR client--->
  <cfset cus = EntityLoad("customer")>
  <cfreturn cus>
</cffunction>

<!---SYNC method, Sync Client data with Server and also handles Conflicts--->
<cffunction name="sync" returnType="any">
  <cfargument name="operations" type="array" required="true">
  <cfargument name="clientobjects" type="array" required="true">
  <cfargument name="originalobjects" type="array" required="true">
  ...
</cffunction>
</cfcomponent>
```

CustomerManager.cfc (2)

```

<cffunction name="sync" returntype="any">
  <cfargument name="operations" type="array" required="true">
  <cfargument name="clientobjects" type="array" required="true">
  <cfargument name="originalobjects" type="array" required="false">

  <cfset conflicts = ArrayNew(1)>
  <cfset conflictcount = 1>

  <cfloop index="i" from="1" to="#ArrayLen(operations)#">

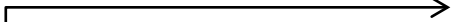
    <cfset operation = operations[i]>
    <cfset clientobject = clientobjects[i]>
    <cfset originalobject = originalobjects[i]>

    <cfif operation eq "INSERT">
      <cfset obj = ORMGetSession().merge(clientobject)>
      <cfset EntitySave(obj)>

    <cfelseif listfindnocase("UPDATE,DELETE",operation) neq 0>
      <cfset serverobject = EntityLoadByPK("customer",originalobject.getcid())>
      <cfif not isdefined('serverobject') >
        <cfset conflict = CreateObject("component","CFIDE.AIR.conflict")>
        <cfset conflict.clientobject = clientobject>
        <cfset conflict.originalobject = originalobject>
        <cfset conflict.operation = operation>
        <cfset conflicts[conflictcount++] = conflict>
        <cfcontinue> <!--- Returns processing to the top of a loop --->
      </cfif>

    <cfset isNotConflict = ObjectEquals(originalobject, serverobject)>

```



```

    <cfif isNotConflict>
      <cfif operation eq "UPDATE">
        <cfset obj = ORMGetSession().merge(clientobject)>
        <cfset EntitySave(obj)>
      <cfelseif operation eq "DELETE">
        <cfset obj = ORMGetSession().merge(originalobject)>
        <cfset EntityDelete(obj)>
      </cfif>
    <cfelse> <!---Conflict--->
      <cflog text = "is a conflict">
      <cfset conflict =
        CreateObject("component","CFIDE.AIR.conflict")>
      <cfset conflict.serverobject = serverobject>
      <cfset conflict.clientobject = clientobject>
      <cfset conflict.originalobject = originalobject>
      <cfset conflict.operation = operation>
      <cfset conflicts[conflictcount++] = conflict>
      <cfcontinue> <!--- Returns processing to the top of a loop --->
    </cfif>
  </cfif>
</cfloop>

<cfif conflictcount gt 1>
  <cfreturn conflicts>
</cfif>

</cffunction>

```

Die vollständige ColdFusion Application

- Application.cfc
- Customer.cfc und Address.cfc als Value Objects mit ORM
- cusManager.cfc für die Implementierung der SYNC und FETCH Methoden



Testen des CustomerManagers

Die Flex AIR Application

Neues Flex-Projekt

Flex-Projekt erstellen

Wählen Sie einen Namen und einen Pfad für Ihr Projekt. Konfigurieren Sie dann

Projektname: SOTR-Offline-AIR

Projektpfad

Standardpfad verwenden

Ordner: C:\ColdFusion9\wwwroot\SOTR-Offline\SOTR-Offline-AIR

Anwendungstyp

Web (wird in Adobe Flash Player ausgeführt)

Desktop (wird in Adobe AIR ausgeführt)

Flex SDK-Version

Standard SDK verwenden (derzeit Flex 4.0*)

Bestimmtes SDK verwenden Flex 3.4

Flex 3.4 requires Adobe AIR 1.5.

Servertechnologie

Anwendungsservertyp ColdFusion

Remote Object Access Service verwenden

LiveCycle-Datendienste ES

BlazeDS

ColdFusion Flash Remoting

Neues Flex-Projekt

Flex-Projekt erstellen

⚠ Der Bezeichner sollte ein eindeutiger paketähnlicher Name sein; beispielsweise „com.example.app“

Quellpfad Bibliothekspfad

Component set: MX + Spark MX only [\[Learn more\]](#)

Framework-Verbindung: Standard-SDK verwenden (in Code zusammengeführt)

Erstellungspfad-Bibliotheken:

- Flex 3.4 - C:\Program Files (x86)\Adobe\Adobe Flash Builder Plug-in Beta 2\sdk\3.4.1
- libs
- cfair.swc - C:\ColdFusion9\wwwroot\CFIDE\scripts\AIR
 - Quellenanhang: (Keine)
 - Verbindungstyp: In Code zusammengeführt
 - RSL-URL: (nicht zutreffend)
 - SWF automatisch extrahieren: (nicht zutreffend)

RSL-Digests überprüfen (für Produktion empfohlen)

Definieren der Customer ActionScript Klasse als VO

Customer.as

```
package vo
{
    [Bindable]
    [RemoteClass(alias="AIRIntegration.customer")]
    [Entity]
    public class Customer
    {
        [Id]
        public var cid:int;
        public var name: String;
        [OneToOne(cascadeType='ALL',fetchType="EAGER")]
        [JoinColumn(name="add_id",referencedColumnName="aid")]
        public var address:Address;
    }
}
```

RemoteClass mapping zu einer entsprechenden ColdFusion CFC

Definieren der Adress ActionScript Klasse als VO

Address.as

```
package vo
{
    [Bindable]
    [RemoteClass(alias=" AIRIntegration.address" )]
    [Entity]
    public class Address
    {
        [Id]
        public var aid:int;
        public var street: String;
    }
}
```

RemoteClass mapping zu einer entsprechenden ColdFusion CFC

Metadaten des ActionScript Persistence Framework (1)

Allgemein

MetaData Tags	Purpose
[Entity]	Specifies that instances of this class can be persisted to the local SQLite database.
[Table(name = "tableName")]	The name of the SQLite table in which the object is to be stored. Defaults to the name of the class.
[Id]	Precedes a field definition. Indicates that the field is a primary key in the table. For composite keys, use Id tags on all the primary key fields.
[Column(name="name", columnDefinition="TEXT INTEGER FLOAT BOOLEAN VARCHACHAR", nullable = true false, unique = true false)]),	<p>Specifies the SQLite database column that contains data for this field.</p> <p>Name: Column name. If not specified, defaults to the property name.</p> <p>columnDefinition: The SQL Datatype to use for the column.</p> <p>Nullable: Specifies whether a field can have a null value.</p> <p>Unique: Specifies whether the value for each field must be unique within the column.</p>

Metadaten des ActionScript Persistence Framework (2)

Relationships

```
[Id]
var id:uint;
[OneToOne(targetEntity="Department" | fetchType="EAGER(default) | LAZY")]
[JoinColumn(name="DEPTID", referencedColumnName="DEPT_ID")]
var dept:Department;
```

```
[Id]
var id:uint;
[OneToMany(targetEntity="Department", mappedBy="department", fetchType="EAGER | LAZY(default)")]
var depts:ArrayCollection;
```

```
[Id]
var id:uint;
[ManyToOne(targetEntity="Department", fetchType="EAGER(default) | LAZY")]
[JoinColumn(name="deptId", referencedColumnName="DEPT_ID")]
var dept:Department;
```

```
[Id]
[Column(name="ID")]
var id:uint;
[ManyToOne(targetEntity="Department", fetchType="EAGER | LAZY(default)")]
[JoinTable(name="EMP_DEPT")]
[JoinColumn(name="EMPID", referencedColumnName="ID")]
[InverseJoinColumn(name="DEPID", referencedColumnName="DEPTID")]
var depts:ArrayCollection;
```

Metadaten des ActionScript Persistence Framework (3)

Lazy loading and fetch type

- Beim Laden von Objekten können in Beziehung stehende Objekte wahlweise sofort mit geladen werden (**eager** loading) oder erst dann, wenn sie wirklich benötigt werden (**lazy** loading).

Metadaten des ActionScript Persistence Framework (4)

Cascading options

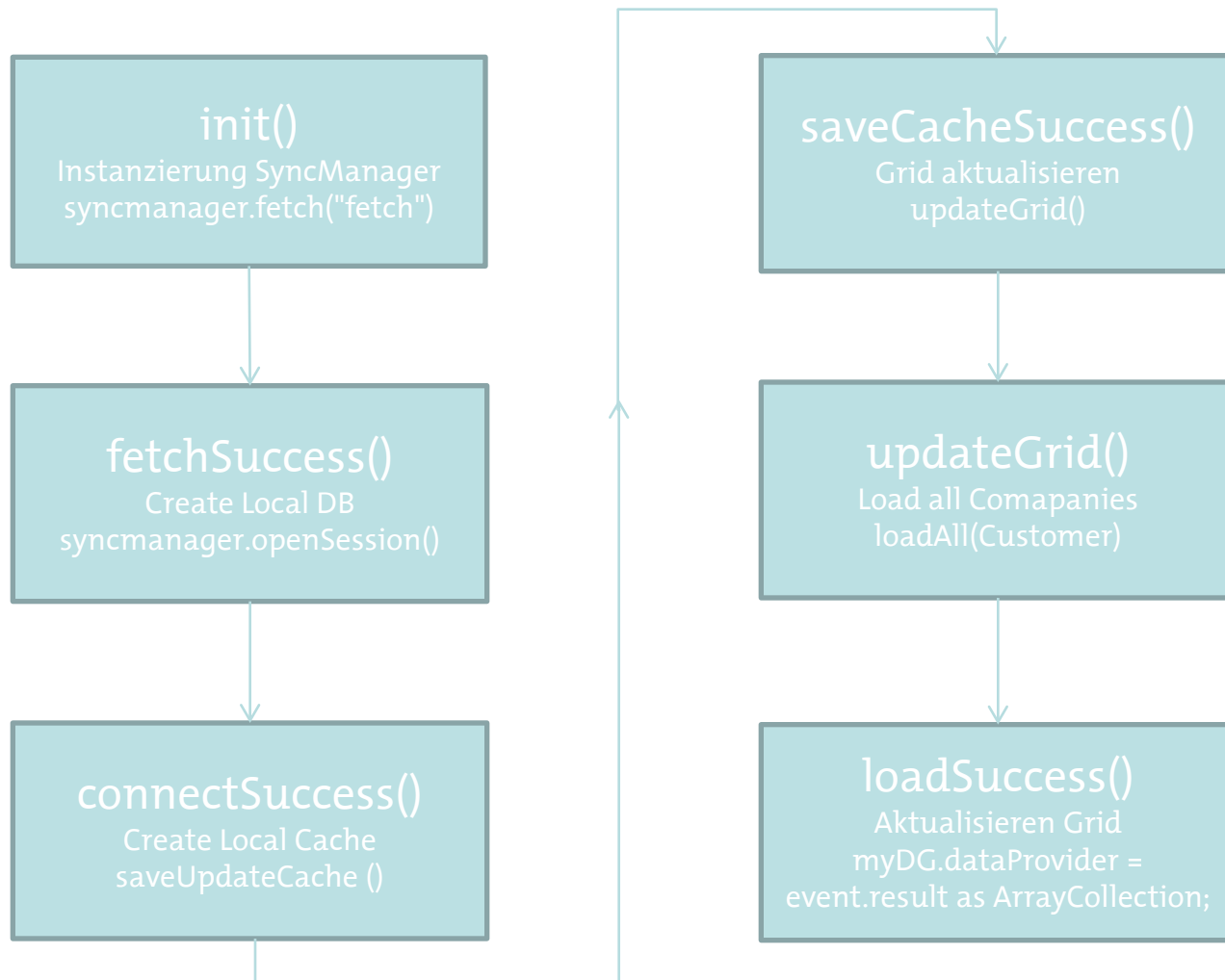
- Cascading lets you specify the operations to be cascaded from the parent object to the associated object. The supported operations are INSERT, UPDATE, and DELETE.
- The cascadeType attribute lets you set any of the following values.
 - ALL: If The source entity is inserted, updated, or deleted, the target entity is also inserted, updated, or deleted.
 - PERSIST: If The source entity is inserted or updated, the target entity is also inserted or updated.
 - REMOVE: If The source entity is deleted, the target entity is also deleted.
- The one-to-one, one-to-many, many-to-one, and many-to-many relationships are all supported by cascading.

```
ManyToMany( cascadeType="ALL or PERSIST or REMOVE" )
```



Demo AIR Anwendung Laden der Daten

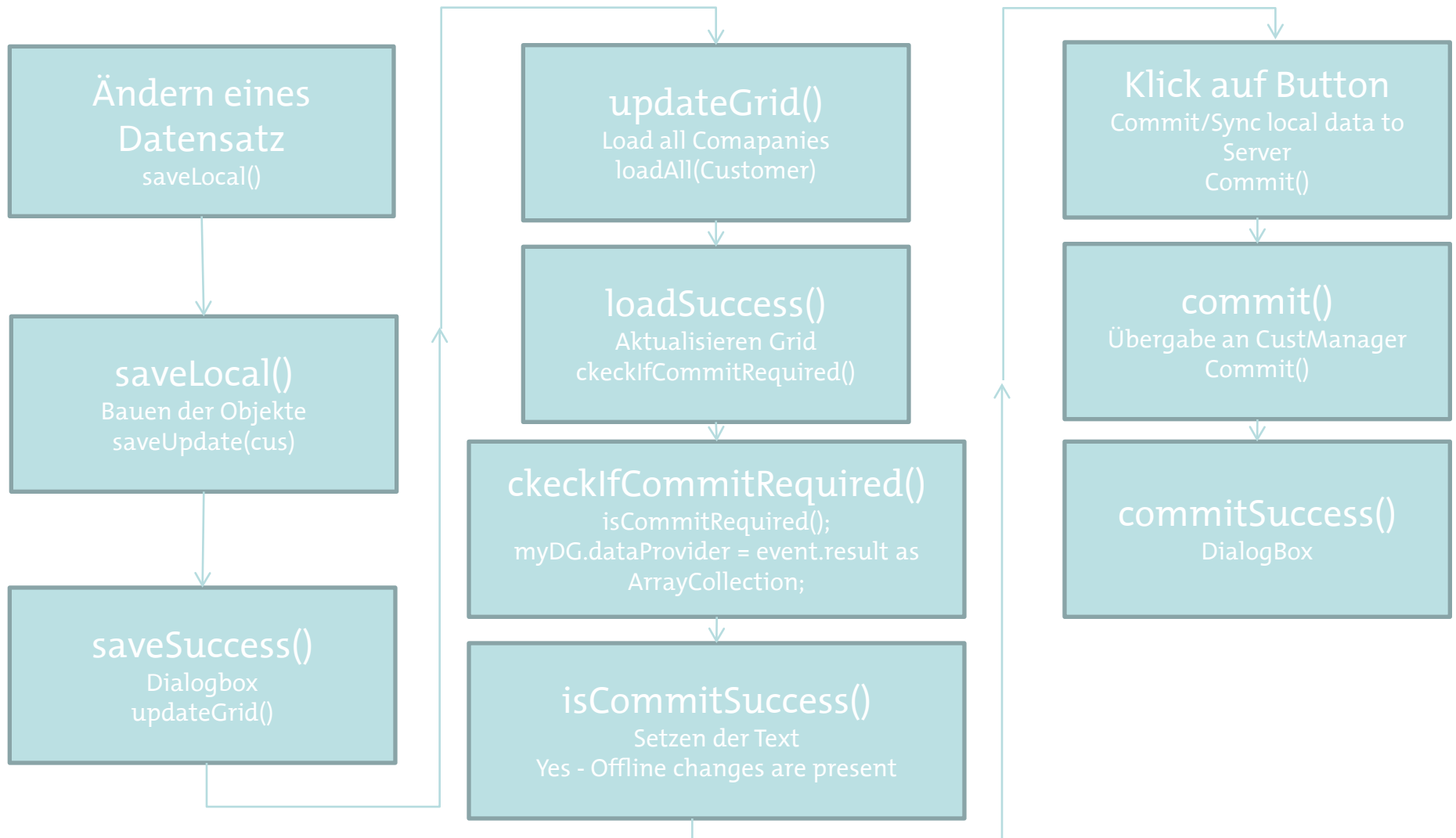
Flow AIR Application (1) – Laden & Anzeige der Daten





Demo AIR Anwendung Updaten von Daten

Flow AIR Application (2) – Update



Vielen Dank für Ihre Aufmerksamkeit

<http://www.richinternet.de>

<http://www.die-flexperten.de>

<http://www.flashmediaserver-blog.de>

<http://www.flexforum.de>

