ColdFusion次バージョン（コードネーム：Splendor）の最新機能とデモンストレーション

ColdFusion Day 2013
2013.12.10

このセッションについて

- お約束
  - 当セッションは、現在開発中の次期ColdFusion（コードネーム：Splendor）についての紹介です。
  - 現時点で紹介可能な情報の中から厳選して紹介しています。
  - 今回紹介する機能は、今後の開発の進捗によっては変更や廃止となる場合があります。
  - リリースの時期やエディションの違いによる機能の有無等はお答えできません。
  - 本資料の内容は予告なく内容の変更や削除等が行われます。予めご了承願います。

ColdFusion Splendorとは？

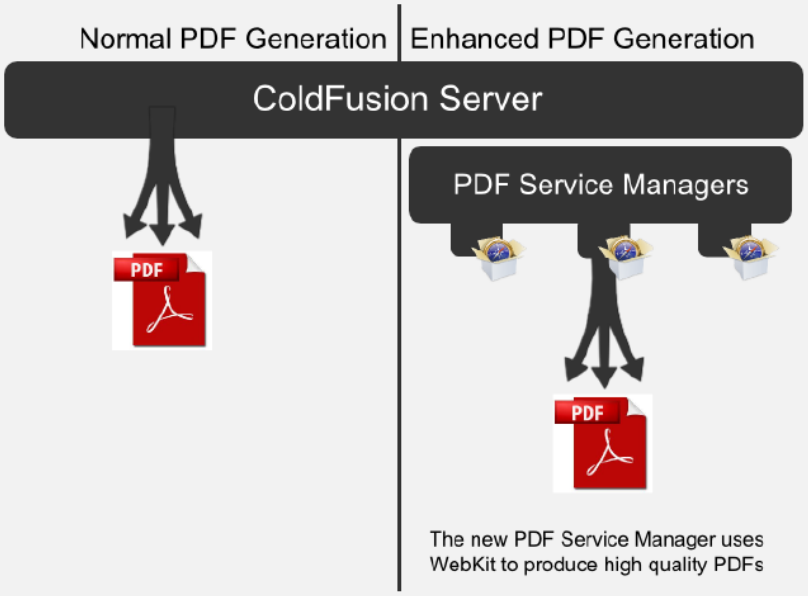
- 米国時間 2012年5月15日にリリースされた ColdFusion 10 に続くメジャーバージョンアップ

- ColdFusion Splendorのポイント
  - PDF生成＆操作
  - モバイルアプリケーション機能
  - JSON機能強化
  - プログラミング強化
  - 他

PDF生成＆操作

ColdFusion Splendor
新しいPDF生成エンジン

- 従来（CF MX7～）
  - IceBrowser & iText
    - <CFDocument>
    - <CFDocumentitem>
    - <CFDocumentsection>
- 新しいPDF生成エンジン
  - Webkit エンジンを利用した高クオリティのHTML→PDF変換
    - <CFHtmlToPDF>
    - <CFHtmlToPDFItem>



ColdFusion Splendor
PDFファイルの操作

- PDFアーカイブ
  - PDFファイルをPDF/A-1bに変換
    - Preserve PDF for long time as a self contained document.
    - <CFPDF action="archive" source=".." destination="..">

- 電子署名
  - PDFに電子署名を付与等の処理
    - <cfpdf action="sign" source="…" destination=".." keystore="cert.jks" keystorepassword="password" signaturefieldname="signField"/>
    - 追加されたアクション
      - SIGN, UNSIGN, VALIDATESIGNATURE, READSIGNATUREFIELDS

モバイルアプリケーション機能

Mobile & ColdFusion

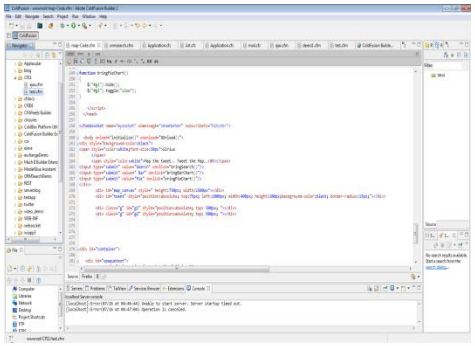
使いやすいモバイルアプリケーション開発
プラットフォーム

End-to-end モバイルアプリケーションワークフロー
ビルド、テスト、デバッグ、デプロイ

ColdFusion & Mobile
End-to-end Solution

1. 作成

<CFML> とJS を使用



2. ビルド & デプロイ

phoneGap Build を利用してiOSやAndroidアプリを作成

3. デバッグ

On Device Step デバッグ

4. テスト

Inspect & Debug across devices

ColdFusion & Mobile
CFML+JavaScript

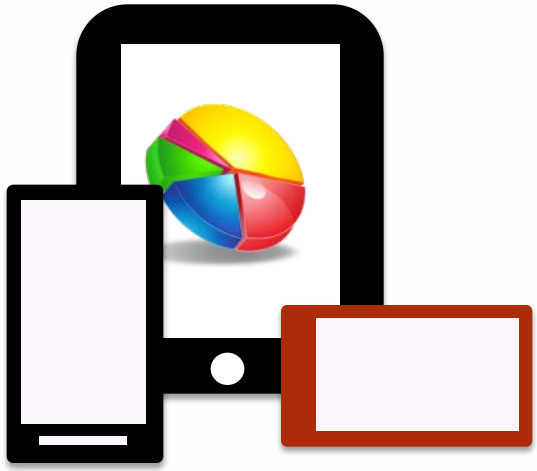
- <cfclient>: CFML → Client Side
  - CFMLはコンパイル時にJavaScriptに変換
  - CFSET, CFOUTPUT, CFSCRIPT
  - CFInclude - CFM, JS & CSS
  - CFC(サーバーサイド、クライアントサイド)
  - CFML カスタムタグの記述形式に対応
- JavaScriptとの相互運用性
- お好みのJSフレームワークを使用可能

ColdFusion & Mobile
注意点

- 注意点（一例）
  - 関数と変数は大文字と小文字が区別される
  - 変数の解析はJSエンジンによる
    - 変数はアクセス時に適切なスコープで定義されていること
  - データ型の自動変換は行われない
    - ブール型や数値型は、ブール·数値のまま
  - <cfinclude>の動的なテンプレートの指定はできない
  - ColdFusionの例外は cfclient 内では動かない

ColdFusion & Mobile
デバイス機能

- <cfclientsettings>でデバイス機能のON/OFFを指定
  - デバイスのプロパティや特性を識別可能
  - 画面サイズ、ブラウザの種類やバージョン、メディアサポート、CSS、HTML、JavaScriptのサポートレベルなど
  - PhoneGapに用意されているデバイス機能に対応する処理を用意



モバイルアプリケーション
To be Continued …

- 引き続きモバイル機能については、情報を取得してアップデートしていきます。

JSON機能強化

JSON機能強化
構造体のキーの大文字小文字の維持

```
<cfscript>
  stTest=StructNew();
  stTest.aaa="aaa";
  stTest["bbb"]="bbb";
</cfscript>
```

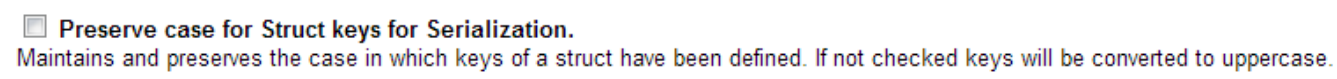
- ColdFusion 10以前では、ドット表記のキー名は大文字に変換される： {"AAA":"aaa", "bbb":"bbb"}
- ColdFusion 11では、ドット表記のキーの大文字小文字の維持が可能： {"aaa":"aaa", "bbb":"bbb"}

JSON機能強化
構造体のキーの大文字小文字の維持（続き）

- 構造体のキーの大文字小文字の維持の有効・無効
  - アプリケーションレベル
    - Application.cfc

```
<cfscript>
  this.name="cfdemo_cf11";
  this.serialization.preserveCaseForStructKey=true;
</cfscript>
```

  - サーバーレベル
    - ColdFusion Administrator の[設定]



JSON機能強化
serializeJSON関数の機能強化

- serializeJSONの queryFormatの指定が強化
  - struct

    serializeJSON(objToSerialize [, queryFormat] [,boolean secure])
    - QueryFormat : " true ¦ false " または " column ¦ row ¦ struct"

  - サンプル

query

	COLOR	ID
1	red	1
2	green	2
3	blue	3

```
//Serialized Output as row
{"COLUMNS":["ID","COLOR"],"DATA":[["1","red"],["2","green"],["3","blue"]]}

//Serialized Output as column
{"ROWCOUNT":3,"COLUMNS":["ID","COLOR"],"DATA":{"ID":["1","2","3"],"COLOR":["red","green","blue"]}}

//Serialized Output as struct
[{"ID":"1","COLOR":"red"},{"ID":"2","COLOR":"green"},{"ID":"3","COLOR":"blue"}]
```

JSON機能強化
serializeJSON関数の機能強化（続き）

- queryFormatをアプリケーションレベルで指定
  - Application.cfc

```
component
{
    this.name="implicitMetaDataDemo";
    this.serialization.serializeQueryAs = "struct | column | row";
}
```

JSON機能強化
新しいシリアライゼーション関数

- ４つのシリアライゼーション関数
  - XML シリアライゼーション関数
    - *serializeXML(objToSerialize, useCustomSerializer)*
    - *deserializeXML(stringToDeserialize, useCustomSerializer)*
  - 一般的なシリアライゼーション関数
    - *serialize(objToSerialize,　type, useCustomSerializer)*
    - *deserialize(strToDeserialize,　type, useCustomSerializer)*

JSON機能強化
カスタム・シリアライザ/デシリアライザ

- 複雑型要素（Complex Type）を扱うためのカスタム･シリアライザ/デシリアライザを指定することが可能
  - Application.cfcでカスタム･シリアライザ/デシリアライザのパスを指定

    this.cutomSerializer = {pathToSerializerCFC};

  - シリアライザCFCには、下記のメソッドの指定が必要:

    boolean canSerialize(type);
    boolean canDeSerialize(type);
    string serializeData(objToBeSerialized, type);
    Object deserializeData(strToBeDeserialized, type);

- すべてのシリアライゼーション関数には、新たな引数が追加され、カスタム・シリアライゼーションの enable / disable を指定できる

プログラミング強化

ColdFusion Splendor
スクリプト記述の拡張

- 一般的なスクリプトのシンタックスに合わせた記述に対応

```
<cfscript>
  親（ベース）タグ 属性1=値1 属性2=値2 …
  {
     サブタグ 属性1=値1 属性2=値2 …
     {
          サブタグ …
     }
  }
</cfscript>
```

- <CFHTTP>,<CFHTTPPARAM>

```
<cfscript>
  http method="post" url="http://www.samuraiz.co.jp/" charset="MS932"
  {
     httpparam name= "username" type="FormField" value= "aaa";
     httpparam name= "password" type="FormField" value= "bbb";
  }
</cfscript>
```

ColdFusion Splendor
スクリプト記述の拡張 <例>

- データベースからデータを取得しグリッドで表示

```
<cfscript>
  query name="qEmp" datasource="cfdocexamples" sql="SELECT *
  FROM Employee";
  form name="Form1"
  {
     grid format="html" name="emp_grid2" query="qEmp"
     {
                gridcolumn name="Emp_ID";
                gridcolumn name="LastName";
                gridcolumn name="Dept_ID";
     }
  }
</cfscript>
```

ColdFusion Splendor
スクリプト記述の拡張 <例>

- タグの属性を括弧内に記述する方法の例

```
<cfscript>
  query(name="qEmp" datasource="cfdocexamples" sql="SELECT *
  FROM Employee");
  form(name="form1")
  {
     gridAttr={format:"html", name:"emp_grid3", query:"qEmp"};
     grid(attributeCollection="#gridAttr#")
     {
                gridcolumn(name="Emp_ID" header="従業員ID");
                gridcolumn(name="LastName" header="名前");
                gridcolumn(name="Dept_ID" header="部署コード");
     }
  }
</cfscript>
```

ColdFusion Splendor
メンバー関数のサポート

- ネイティブデータ構造/オブジェクトに対するオブジェクト指向型メソッド呼び出し。
  - ArrayLen(arr)に相当：arr.len()
- Array
- Struct
- String
- List
- Date
- Query
- Image
- Spreadsheet
- XML

ColdFusion Splendor
メンバー関数のサポート（続き）

- これまでは関数としては、ColdFusionで以前から定められてる方法だけだった
  - ArrayAppend(arrayObj, objToAppend)
- メンバー関数の導入により、オブジェクト指向モデルと整合するコーディング記述が行える
  - arrayObj.append(objToAppend);
- シンタックス
  - obj.MemberFunction([argument(s)])

ColdFusion Splendor
メンバー関数のサポート＜例＞

- 記述例

```
<cfscript>
  aFruit=ArrayNew(1);
  ArrayAppend(aFruit,"リンゴ");
  ArrayAppend(aFruit,"みかん");
  ArrayAppend(aFruit,"メロン");
  writeOutput(ArrayLen(aFruit));

  //新しい記述方法
  aFruit.add("パパイヤ"); // Java API
  aFruit.append("キウイ"); // CF API
  writeOutput(aFruit.Len());
</cfscript>
```

ColdFusion Splendor
メンバー関数のサポート（補足）

- メンバ関数について覚えておくべき重要なこと：

  - 基礎となるデータ構造の参照はColdFusionのクラス（coldfusion.runtime.Array）またはJavaクラス（java.util.Listの）のいずれか

  - デベロッパーは、もしColdFusionクラスがJavaクラスを実装/拡張する場合、Javaインタフェースを指すオブジェクトに直接ColdFusion関数を呼び出すことができるようになります

  - ColdFusionとのJava APIが同じ機能のために異なる場合、両方がサポートされる。array.append（）とarray.add（）新しいオブジェクトを追加するために使用することができます

ColdFusion Splendor
メンバー関数のサポート（補足）

- リスト関数はStringオブジェクトとして利用できる
- 関数は、両方のデータ型で存在する

List	String
listFind	find
listFindNoCase	findNoCase
listLen	len

- 文字列関数は、リテラル文字列は呼び出すことはできない
  - "xyz".len()　//は動作しない

ColdFusion Splendor
QueryExecute関数

- QueryExecute
  - QueryExecute(クエリ文 [,クエリパラメーター] [,クエリオプション])

```
<cfscript>
  qEmp1=QueryExecute("SELECT * FROM
  Employee","",{datasource="cfdocexamples"});
  qEmp2=QueryExecute("SELECT * FROM Employee where
  DEPT_ID=1","",{datasource="cfdocexamples"});
  qEmp3=QueryExecute("SELECT * FROM Employee where
  DEPT_ID=?",[1],{datasource="cfdocexamples"});
  qEmp4=QueryExecute("SELECT * FROM Employee where
  DEPT_ID=? AND EMP_ID=?",[1,1],{datasource="cfdocexamples"});
  qEmp5=QueryExecute("SELECT * FROM Employee where
  DEPT_ID=:deptid AND EMP_ID=:empid",
  {deptid:{type:"integer",value:1}, empid:7},
  {datasource="cfdocexamples"});
</cfscript>
```

ColdFusion Splendor
QueryGetRow関数

- QueryExecute
  - 特定のクエリ行を取得
    - QueryGetRow(queryObj, rowIndex)
  - クエリオブジェクトに対しては下記の方法でも同様に対応
    - row = queryObj.getRow(rowIndex)

```
<cfscript>
  qEmp=QueryExecute("SELECT * FROM
  Employee","",{datasource="cfdocexamples"});
  stEmpData=QueryGetRow(qEmp,1);
  writeDump(stEmpData);
</cfscript>
```

struct	
CONTRACT	Y
CONTRACT_FILE	/opt/coldfusionmx/wwwroot/vw_files/contracts/frueh.txt
DEPT_ID	1
EMP_ID	1
FIRSTNAME	Benerwrwerrw
LASTNAME	Frueh
PROJECT_DOCS	/opt/coldfusionmx/wwwroot/vw_files/frueh
SALARY	100000
STARTDATE	{ts '1987-01-19 00:00:00'}

ColdFusion Splendor
ZIPファイルのパスワードサポート

- CFZIPタグで、パスワード付きのファイルをサポート
  - 追加された属性
    - Password
    - encryptionAlgorithm
  - 暗号化アルゴリズム
    - Standard (zip2.0)
    - AES-128
    - AES-256 (default)
  - <cfzip action="zip" encryptionAlgorithm="AES-128¦AES-256¦Standard" password = "passwd" … >
  - <cfzip action="unzip" password="passwd" …>

ColdFusion Splendor
CFMAILの暗号化

- CFMAILタグを使用したメールの暗号化をサポート
  - <cfmail
    …
    encrypt="true"
    encryptionAlogirthmn = "algorithm"
    recipientcert="certPath"
    >
    Mail content
    </cfmail>

アルゴリズム-
DES_EDE3_CBC,RC2_CBC,AES128_CBC,AES192_CBC,AES256_CBC

ColdFusion Splendor
CFLOGIN, セキュリティ関連

- <cflogin>
  - MX～9 同じユーザー名での複数同時ログイン可
  - 10 同じユーザー名での複数同時ログイン不可
  - 11 allowconcurrent="true ¦ false" 設定可（Administrator設定有）
- ColdFusion Administrator でセキュアプロファイルの有効/無効切り替え
  - 10 インストール時に有効/無効を切り替え
- Adminコンポーネントに対するIP アドレス制限の追加
  - MX～9 IPアドレス制限なし
  - 10 Administrator に対するIPアドレス制限を追加
  - IPアドレス制限に adminapi, componentexplorer を追加

お問い合わせ先

株式会社サムライズ

アドビソフトウェア事業部 ColdFusion ビジネスユニット

E-mail: adobe_software@samuraiz.co.jp

http://www.samuraiz.co.jp/

※サムライズのホームページでColdFusion情報を公開中

http://www.samuraiz.co.jp/adobeproduct/coldfusion/index.html

(ColdFusion カフェテリア) http://forum.samuraiz.co.jp

(ColdFusion Associate) http://cfassociates.samuraiz.co.jp