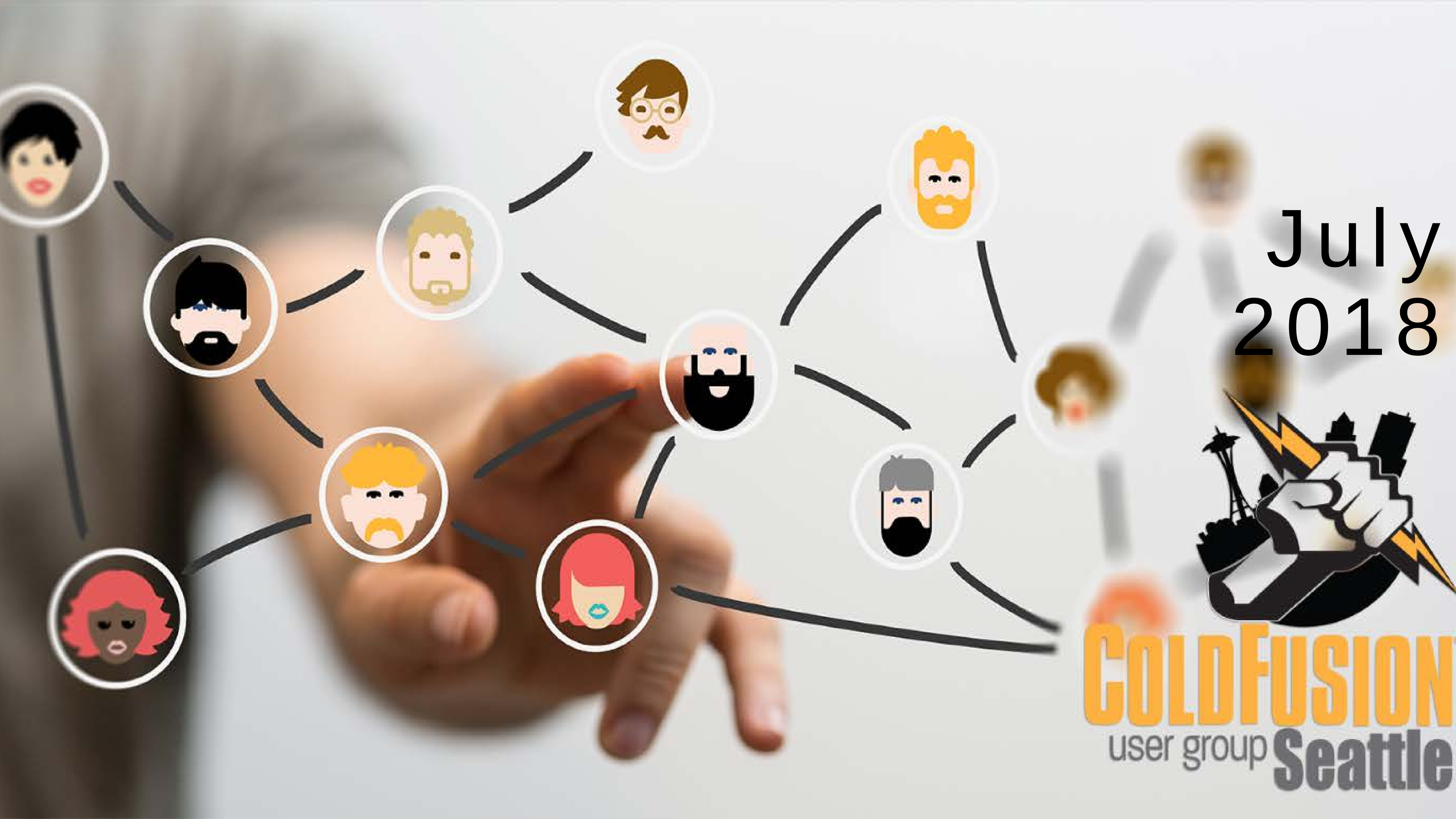


July 2018

ColdFusion user group Seattle

# Agenda

- Welcome to the Seattle ColdFusion User Group
- Introductions
- Goals
- Upcoming ColdFusion Events
- Add Your Bio to the Seattle CFUG site
- CFSpreadsheet demo by William Frankhouser of Everett CC / WilzDezign
- Fun with PDFs and ColdFusion by Jim Bates of Cherwell Software / Latreia
- Get Your Questions Answered. Ask the Group!
- Next Month’s Meeting

# Introductions

**Tell Us a Little About You**

- Your name
- Where You Work/What You Do
- What You’re Hoping to Get From Attending Our User Group Meetings

# Goals

- Assist ColdFusion Developers Throughout the Pacific Northwest
- Promote ColdFusion Developers Throughout the Pacific Northwest
- Connect Employers with ColdFusion Developers
- Establish a Community of Friendship Between ColdFusion Developers
- Provide Speaking Opportunities for ColdFusion Developers
- Change the Perception of ColdFusion as a viable platform

# Upcoming ColdFusion Events

**Adobe ColdFusion Summit 2018**
Hard Rock Hotel and Casino – Las Vegas
October 1st – 3rd
Register at https://cfsummit.adobeevents.com/register/

- Announce Pricing (until Jun 30th) - $249.00
- Advance Pricing (until Aug 31st)  - $349.00
- Full Price (Sep 1 to Oct 3rd) - $399.00

# CF Spreadsheet Basics

**William Frankhouser**
Everett CC / Wilz Dezign

# CF Basics - CFSpreadsheet

- Goal to create and update easy to use spreadsheets for reports with simple formatting using <cfspreadsheet>
- Functionality existed before using drivers, parsing of data, changing MIME type and using cfheader to export the data out.
- Adobe introduced CFSpreadsheet in ColdFusion 9
- Minor attribute updates in 9.0.1 and 11 added autosize
- Not Supported in Lucee
- However, extension available for Lucee 5 via Leftbower

# CFSpreadsheet Basics

- READ a Spreadsheet
- UPDATE a Spreadsheet
- WRITE a Spreadsheet

# CFSpreadsheet READ

```
<cfspreadsheet
        action="read"
        src = "filepath"
        columns = "range"
        columnnames = "comma-delimited list"
        excludeHeaderRow = "true | false"
        format = "CSV|HTML"
        headerrow = "row number"
        name = "text"
        query = "query name"
        rows = "range"
        sheet = "number"
        sheetname = "text">
```

# CFSpreadsheet UPDATE

```
<cfspreadsheet
        action="update"
        filename = "filepath"
        format = "csv"
        name = "text"
        password = "password"
        query = "query name"
        sheetname = "text" >
```

# CFSpreadsheet WRITE

```
<cfspreadsheet
        action="write"
        filename = "filepath"
        format = "csv"
        name = "text"
        overwrite = "true | false"
        password = "password"
        query = "queryname"
        sheetname = "text"
        autosize = "true | false" >
```

# CFSpreadsheet Functions

Full Function List

- **Add**
  SpreadSheetAddAutofilter, SpreadsheetAddColumn, SpreadsheetAddFreezePane, SpreadsheetAddImage, SpreadsheetAddInfo, SpreadSheetAddPagebreaks, SpreadsheetAddRow, SpreadsheetAddRows, SpreadsheetAddSplitPane
- **Create**
  SpreadsheetCreateSheet
- **Delete**
  SpreadsheetDeleteColumn, SpreadsheetDeleteColumns, SpreadsheetDeleteRow, SpreadsheetDeleteRows
- **Format**
  SpreadsheetFormatCell, SpreadsheetFormatCellRange, SpreadsheetFormatColumn, SpreadsheetFormatColumns, SpreadsheetFormatRow, SpreadsheetFormatRows

Plus Many More…

# Fun with PDFs …and Coldfusion

Jim Bates

# Overview

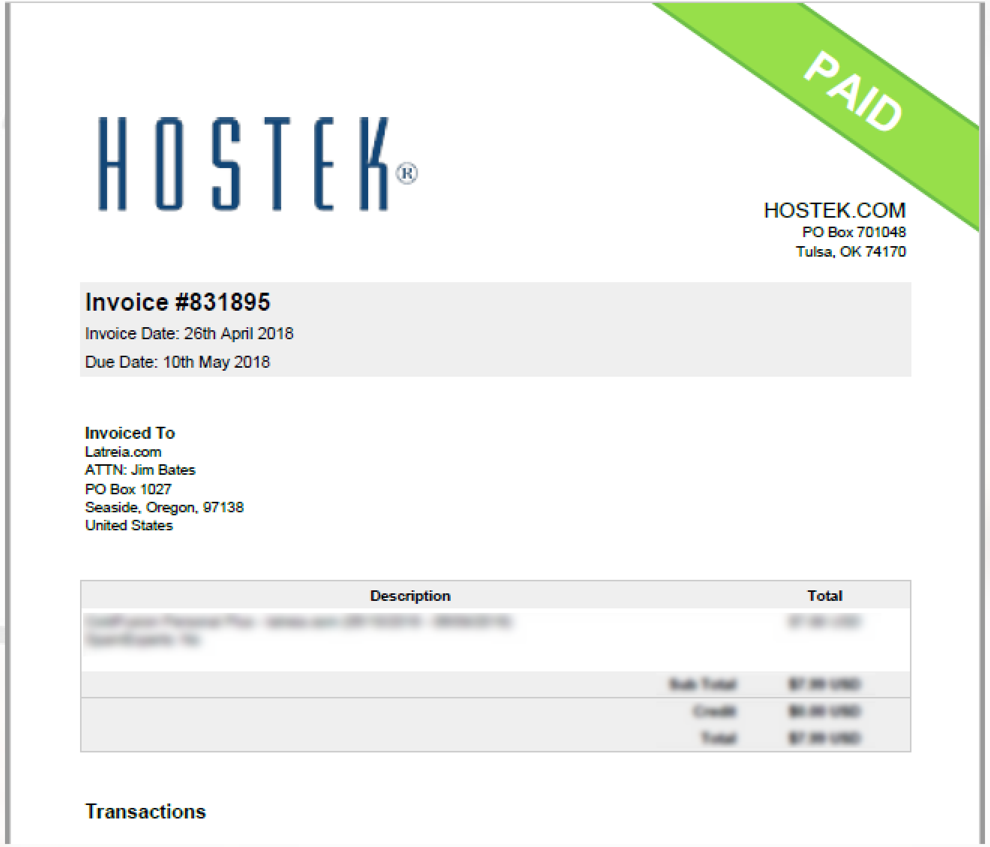
- What, why and how?
- Basic pattern
- Four specific functions:
    - cfdocument
    - cfreport
    - cfpdf
    - cfpdfform

# What and Why?

- Use CF to **add dynamic data** to PDF
  - In an MVC world, PDF as the view
- Why? **Consistent** user experience
  - Printable
  - Emailable
  - Offline data storage
- Nice formatting
- Ubiquitous software

# How can you use it?

- Invoicing
- Customized document creation
  - Form letters
  - Certificates
- Form data import
- Reports
  - Charts and graphs
  - Mailing labels
  - Coupons - barcode, QR codes
  - Tickets



PAID

HOSTEK®

HOSTEK.COM
PO Box 701048
Tulsa, OK 74170

**Invoice #831895**

Invoice Date: 26th April 2018

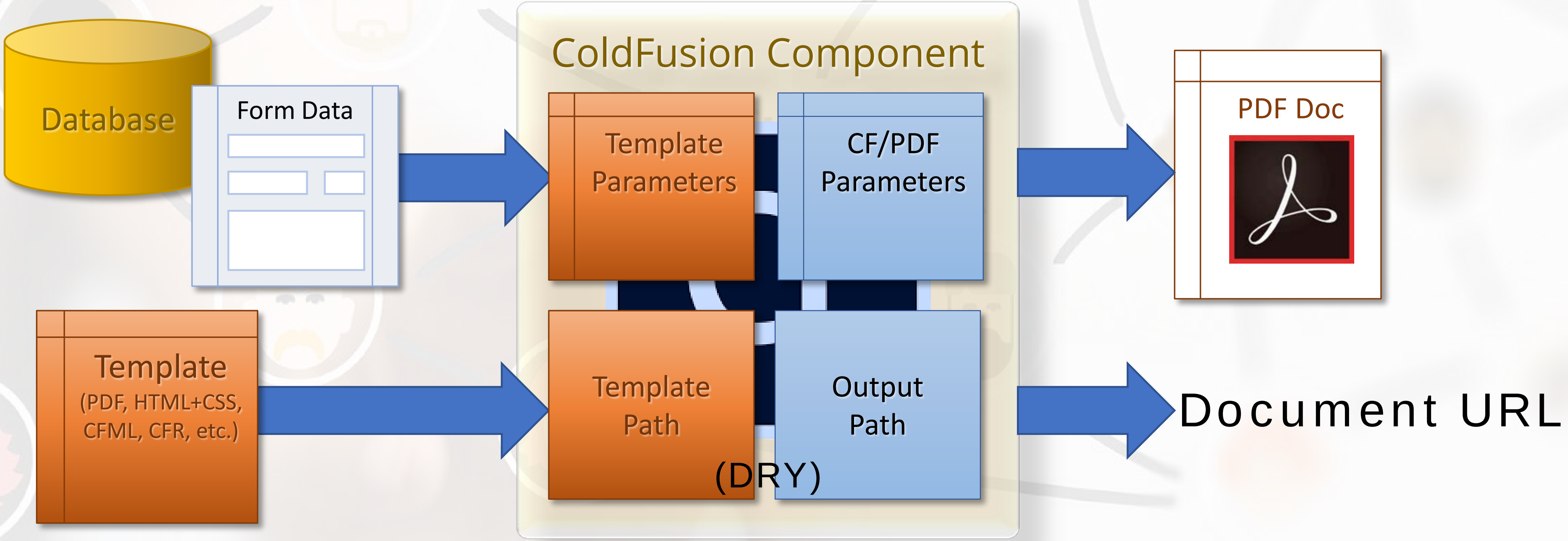
Due Date: 10th May 2018

**Invoiced To**
Latreia.com
ATTN: Jim Bates
PO Box 1027
Seaside, Oregon, 97138
United States

| Description | Total |
|---|---|
| [illegible] | [illegible] |
| Sub Total | [illegible] |
| Credit | [illegible] |
| Total | [illegible] |

**Transactions**

# Basic Pattern



# Tonight's examples

- PDF publishing using **cfdocument**
  - Simple certificate
- Banded Reports with **cfreport**
  - Avery labels
- PDF concatenation using **cfpdf** and **DDX**
  - Assemble a brochure
- PDF form data ingestion with **cfpdfform**
  - Feedback form

# cfdocument

CFML + CSS to PDF

# cfdocument

- Use cases
  - Certificates, posters, signage
  - Output code examples
- Anything that:
  - Requires consistent UX
  - May be printed
  - Is distributed digitally

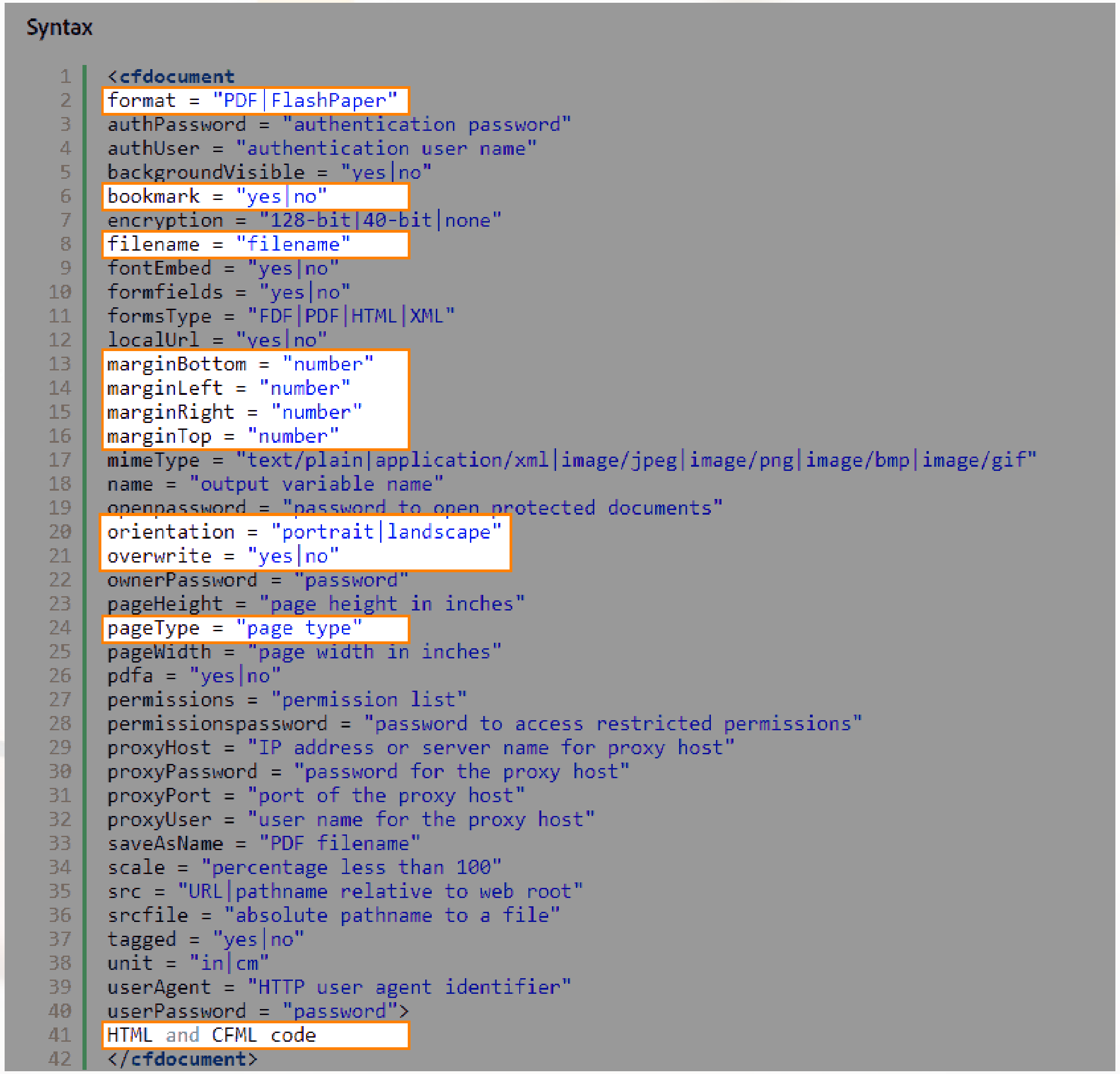
# cfdocument

- Easiest way to generate PDFs
- Block of cfml & html wrapped in <cfdocument>
- Additional feature tags
    - <cfdocumentSection>

      *Generates page break, formatting, gives name for TOC*
    - <cfdocumentItem>

      *Force page break, special header/footer*

https://helpx.adobe.com/coldfusion/cfml-reference/coldfusion-tags/tags-d-e/cfdocument.html



Syntax

```
<cfdocument
format = "PDF|FlashPaper"
authPassword = "authentication password"
authUser = "authentication user name"
backgroundVisible = "yes|no"
bookmark = "yes|no"
encryption = "128-bit|40-bit|none"
filename = "filename"
fontEmbed = "yes|no"
formfields = "yes|no"
formsType = "FDF|PDF|HTML|XML"
localUrl = "yes|no"
marginBottom = "number"
marginLeft = "number"
marginRight = "number"
marginTop = "number"
mimeType = "text/plain|application/xml|image/jpeg|image/png|image/bmp|image/gif"
name = "output variable name"
openpassword = "password to open protected documents"
orientation = "portrait|landscape"
overwrite = "yes|no"
ownerPassword = "password"
pageHeight = "page height in inches"
pageType = "page type"
pageWidth = "page width in inches"
pdfa = "yes|no"
permissions = "permission list"
permissionspassword = "password to access restricted permissions"
proxyHost = "IP address or server name for proxy host"
proxyPassword = "password for the proxy host"
proxyPort = "port of the proxy host"
proxyUser = "user name for the proxy host"
saveAsName = "PDF filename"
scale = "percentage less than 100"
src = "URL|pathname relative to web root"
srcfile = "absolute pathname to a file"
tagged = "yes|no"
unit = "in|cm"
userAgent = "HTTP user agent identifier"
userPassword = "password">
HTML and CFML code
</cfdocument>
```

# cfdocument in cfc function

```
/**
* @displayname Create Preface
* @hint Generates a personalized Preface
* @description Uses template to generate personalized Preface
* @output false
* @returnFormat json
* @secureJSON yes
* @verifyClient yes
*/
public string function createPreface(required string name, required string template, required any suffix)
{
     var tempPath = instance.tempPath;
     var prefacePath = tempPath & "Preface-#arguments.suffix#.pdf";

     cfdocument( format="pdf", filename=expandPath(prefacePath), overwrite=true, bookmark="yes" ) {
          cfdocumentSection(name="Preface") {
              include "/classes/templates/#arguments.template#";
          }
     }

     return prefacePath;
}
```

# cfdocument in cfc function

```
<!--- Create Preface --->
<cffunction name="createPreface" access="public" output="false" returntype="string" returnformat="JSON"
    hint="Generates a personalized Preface" secureJSON="Yes" verifyClient="Yes">

    <cfargument name="name" type="string" required="yes">
    <cfargument name="template" type="string" required="yes">
    <cfargument name="suffix" type="any" required="yes">

    <cfset variables.tempPath = instance.tempPath>
    <cfset variables.prefacePath = tempPath & "Preface-#arguments.suffix#.pdf">

    <cfdocument format="pdf" filename=prefacePath overwrite=true bookmark="yes">
         <cfdocumentSection name="Preface">
            <cfinclude template="/classes/templates/#arguments.template#.cfm">
         </cfdocumentSection>
    </cfdocument >

    <cfreturn prefacePath>
</cffunction>
```

So where are margins, page type, etc?

- Template has defaults
- Parameters passed in the Arguments scope are visible to the included template.
- So they are passed in via the argumentCollection, and rendered by the <cfoutput> in the template.

# practical considerations

- Customized filenames need to be unique
  - So customer files don’t get muddled
  - Use a suffix (random number or datetime) to keep unique
- Be sure to include file cleanup
  - After timespan – *and/or* –
  - Programmatic: after confirm email sent, etc.
  - Serve with cfcontent with deletefile="yes"
- I recommend creating a template .cfm, with embedded CSS
  - Output whatever variables needed
  - Allows separation of concerns ☺

# cfdocument example

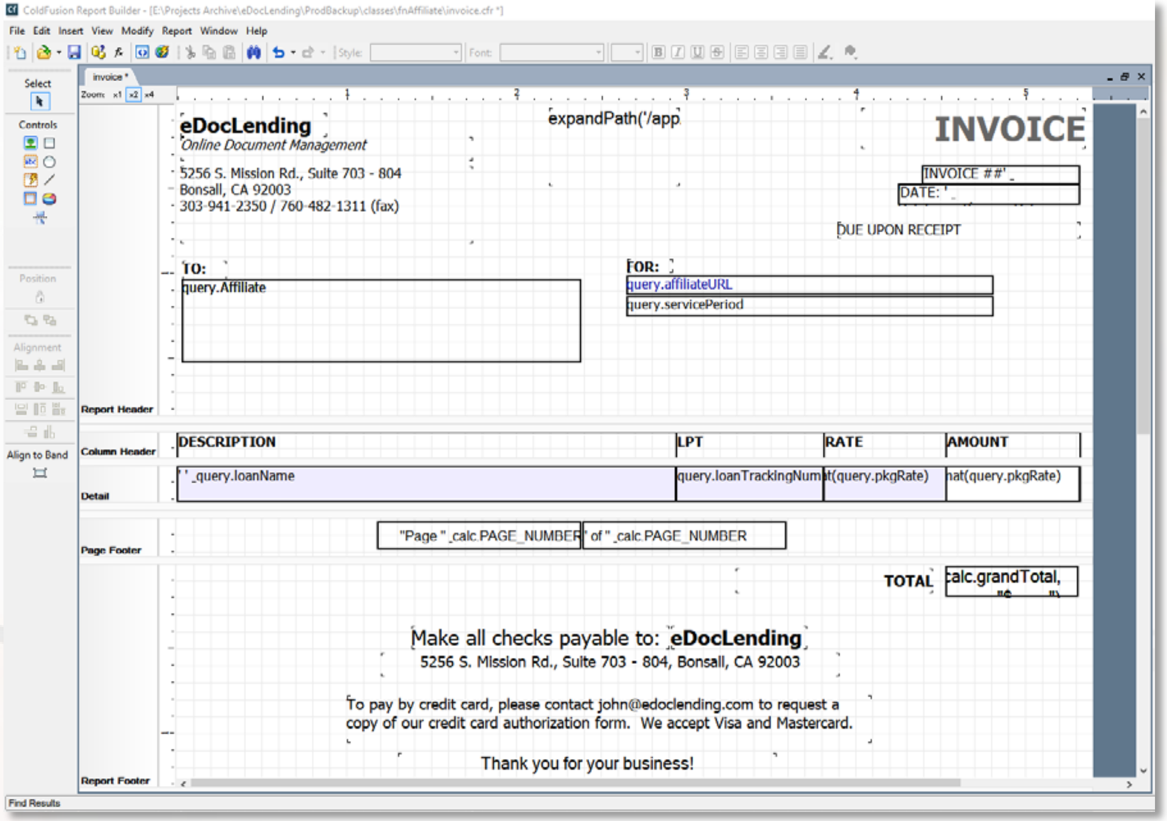
- Quick demo:
  - Log into http://training.latreia.com as cfug / seattle
  - My courses > Fun with PDF
  - Click ‘eyecon’ on cfdocument to expand the topic
  - Click demo of cfdocument.
- Fill in name, select presentation
- Click button ‘Make the Certificate!’

# cfreport

Banded reports using CF Report Builder

# cfreport

- In 2005, with ColdFusion MX7, Macromedia released ColdFusion Report Builder
  - Banded report generator
  - Fairly decent
- Use query builder or write custom query
- Output is a .cfr **template** file



ColdFusion Report Builder - [E:\Projects Archive\eDocLending\ProdBackup\classes\fnAffiliate\invoice.cfr *]

**eDocLending**
*Online Document Management*

5256 S. Mission Rd., Suite 703 - 804
Bonsall, CA 92003
303-941-2350 / 760-482-1311 (fax)

expandPath('/app

**INVOICE**

INVOICE ##'_
DATE: '_

DUE UPON RECEIPT

**TO:**
query.Affiliate

**FOR:**
query.affiliateURL
query.servicePeriod

| DESCRIPTION | LPT | RATE | AMOUNT |
|---|---|---|---|
| ' ' _query.loanName | query.loanTrackingNum | t(query.pkgRate) | nat(query.pkgRate) |

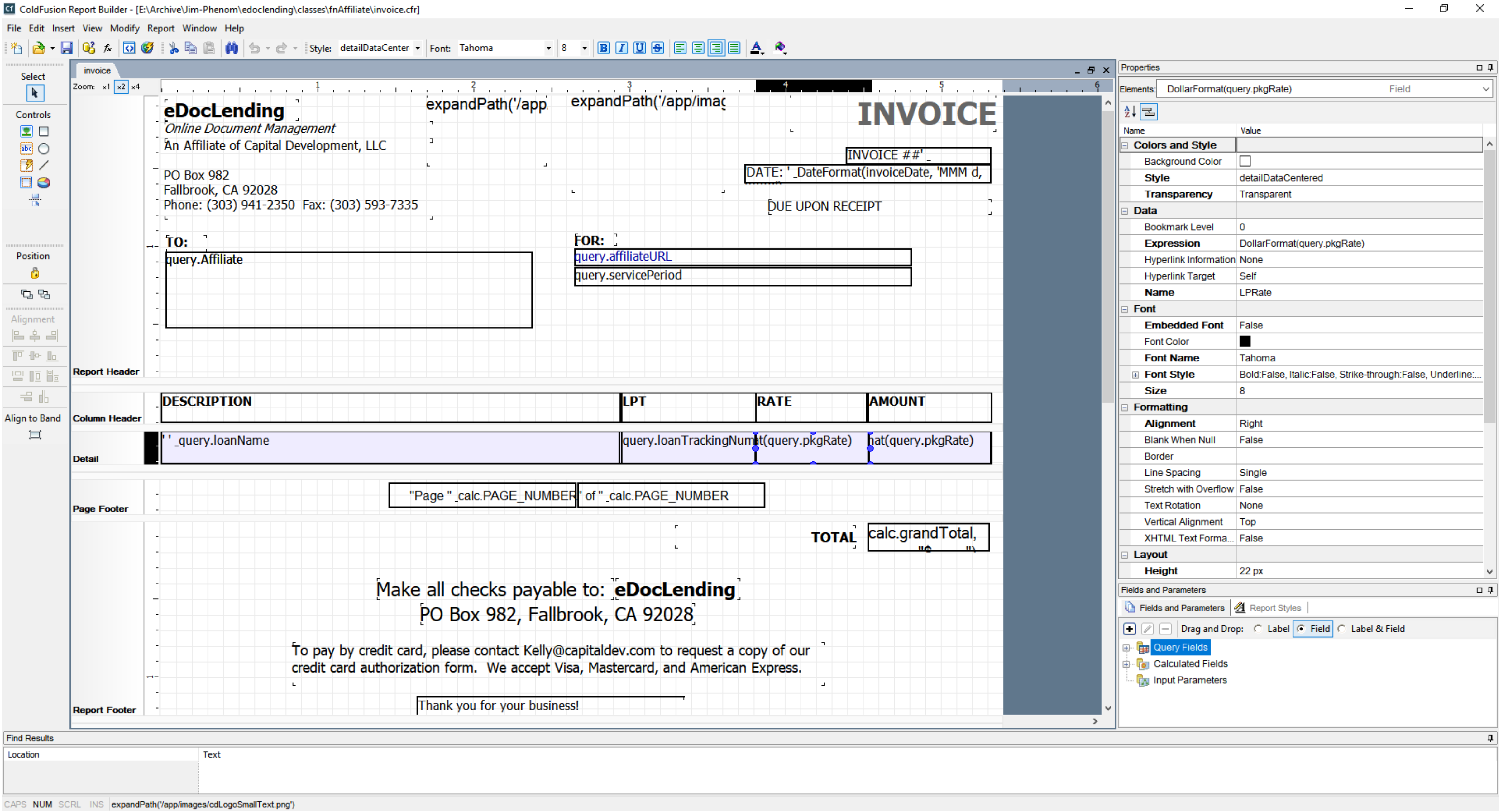
"Page " _calc.PAGE_NUMBER | " of " _calc.PAGE_NUMBER

**TOTAL** calc.grandTotal,

Make all checks payable to: **eDocLending**
5256 S. Mission Rd., Suite 703 - 804, Bonsall, CA 92003

To pay by credit card, please contact john@edoclending.com to request a
copy of our credit card authorization form. We accept Visa and Mastercard.

Thank you for your business!



ColdFusion Report Builder - [E:\Archive\Jim-Phenom\edoclending\classes\fnAffiliate\invoice.cfr]

**eDocLending**
*Online Document Management*
An Affiliate of Capital Development, LLC

PO Box 982
Fallbrook, CA 92028
Phone: (303) 941-2350 Fax: (303) 593-7335

expandPath('/app

expandPath('/app/imag

**INVOICE**

INVOICE ##' _

DATE: ' _DateFormat(invoiceDate, 'MMM d,

DUE UPON RECEIPT

**TO:**

query.Affiliate

**FOR:**

query.affiliateURL

query.servicePeriod

Report Header

| DESCRIPTION | LPT | RATE | AMOUNT |
|---|---|---|---|
| ' ' _query.loanName | query.loanTrackingNum | t(query.pkgRate) | nat(query.pkgRate) |

Column Header

Detail

"Page " _calc.PAGE_NUMBER ' of " _calc.PAGE_NUMBER

Page Footer

**TOTAL** calc.grandTotal,

Make all checks payable to: **eDocLending**
PO Box 982, Fallbrook, CA 92028

To pay by credit card, please contact Kelly@capitaldev.com to request a copy of our credit card authorization form. We accept Visa, Mastercard, and American Express.

Thank you for your business!

Report Footer

Properties

Elements: DollarFormat(query.pkgRate) Field

| Name | Value |
|---|---|
| **Colors and Style** | |
| Background Color | □ |
| **Style** | detailDataCentered |
| **Transparency** | Transparent |
| **Data** | |
| Bookmark Level | 0 |
| **Expression** | DollarFormat(query.pkgRate) |
| Hyperlink Information | None |
| Hyperlink Target | Self |
| **Name** | LPRate |
| **Font** | |
| **Embedded Font** | False |
| Font Color | ■ |
| **Font Name** | Tahoma |
| **Font Style** | Bold:False, Italic:False, Strike-through:False, Underline:... |
| **Size** | 8 |
| **Formatting** | |
| **Alignment** | Right |
| Blank When Null | False |
| Border | |
| Line Spacing | Single |
| Stretch with Overflow | False |
| Text Rotation | None |
| Vertical Alignment | Top |
| XHTML Text Forma... | False |
| **Layout** | |
| **Height** | 22 px |

Fields and Parameters

Drag and Drop: ○ Label ● Field ○ Label & Field

expandPath('/app/images/cdLogoSmallText.png')

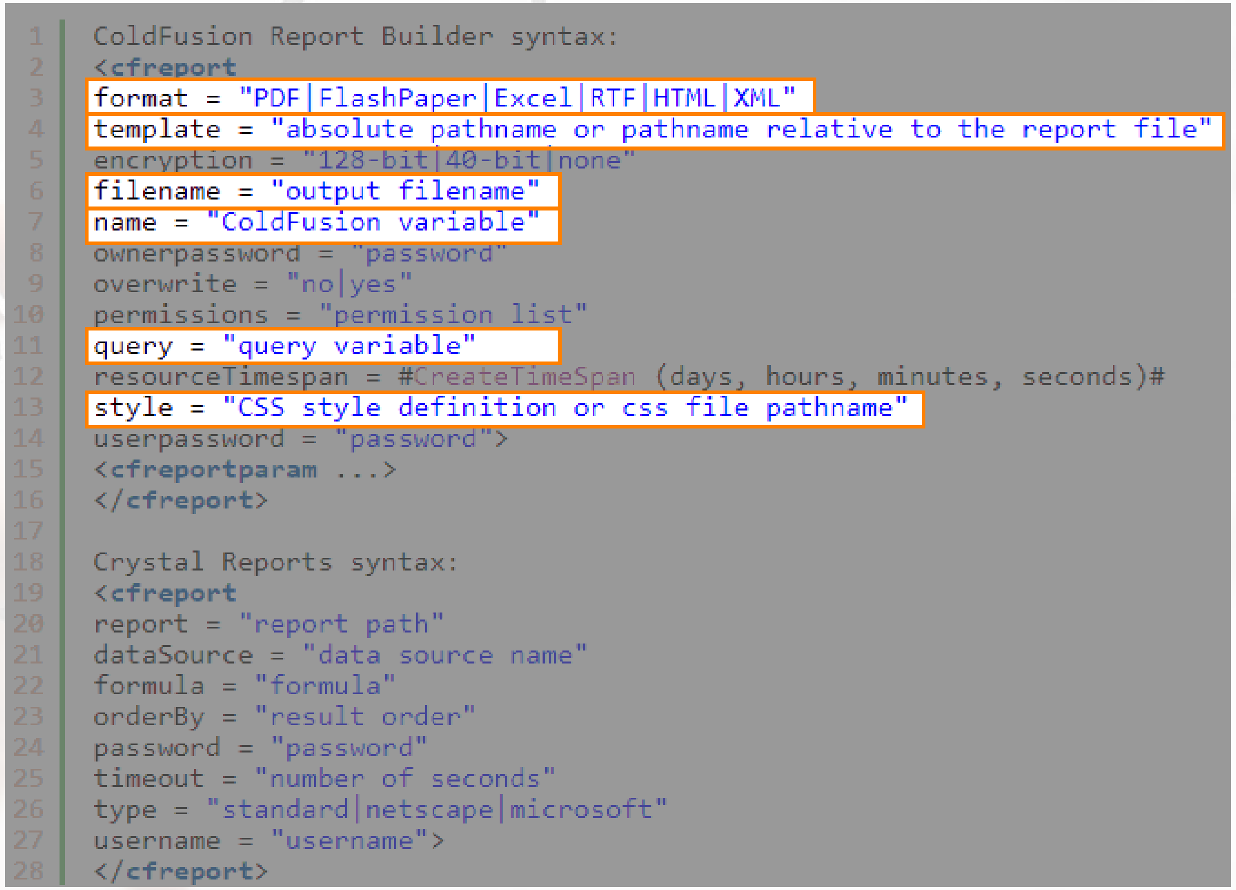
# cfreport

- Use cases:
  - Banded reports
  - Invoicing
  - Labels
- Anything that:
  - Has strict format for multiple data elements
  - Requires visual presentation of data
  - Will be printed and/or distributed digitally

# cfreport

- <cfreport>
  - Uses .cfr **template** with **data**
- Additional feature tag
  - <cfreportparam>

    *Pass input parameters, override query data in subreports and charts, override styles*



```
ColdFusion Report Builder syntax:
<cfreport
format = "PDF|FlashPaper|Excel|RTF|HTML|XML"
template = "absolute pathname or pathname relative to the report file"
encryption = "128-bit|40-bit|none"
filename = "output filename"
name = "ColdFusion variable"
ownerpassword = "password"
overwrite = "no|yes"
permissions = "permission list"
query = "query variable"
resourceTimespan = #CreateTimeSpan (days, hours, minutes, seconds)#
style = "CSS style definition or css file pathname"
userpassword = "password">
<cfreportparam ...>
</cfreport>

Crystal Reports syntax:
<cfreport
report = "report path"
dataSource = "data source name"
formula = "formula"
orderBy = "result order"
password = "password"
timeout = "number of seconds"
type = "standard|netscape|microsoft"
username = "username">
</cfreport>
```

https://helpx.adobe.com/coldfusion/cfml-reference/coldfusion-tags/tags-r-s/cfreport.html

# using cfreport

```
<cfset template = expandPath("/templates/Avery5162.cfr")>

<cfquery name="qLabels" datasource="training">
   SELECT Firstname, Lastname,
          CASE
                WHEN Address2 IS NULL THEN Address1
                WHEN datalength(Address2) = 0 THEN Address1
                ELSE Address1 + CHAR(13)+CHAR(10) + Address2
          END AS Address,
          City, State_Prov, ZIP_Postal
     FROM Addresses ORDER BY Lastname, Firstname
</cfquery>

<cftry>
   <cfreport format="PDF" template="#template#" query="qLabels" name="labels" />
   <cfheader name="Content-Disposition" value="attachment;filename=Avery5162.pdf">
   <cfcontent type="application/pdf" variable="#labels#">
    <cfcatch>
       <cfdump var="#cfcatch#">
    </cfcatch>
</cftry>
```

I let the database do the work wherever it is appropriate…

# cfreport in a cfc function

```
/**
 * @displayname Publish Report
 * @description Given path to cfreport, recordset and output path, publishes PDF report
 * @hint .cfr path and output path are URLs from server root
 * @output false
 * @returnFormat json
 * @secureJSON yes
 * @verifyClient yes
 */
public string function publishReport( required string templatePath, required query recordSet, required string outputPath, required string filename )
{
    var result = "Must specify templatPath (path of .cfr file), a valid recordset, outputPath (path to PDF file) and filename for report PDF.";
    try {
        templatePath = expandPath(arguments.templatePath);
        outputPath = expandPath(arguments.outputPath);
        if( right(outputPath, 1) != '\' ) {
            outputPath &= '\';
        }
        outputPath &= arguments.filename;
        cfreport( format="PDF", template="#templatePath#", query="recordSet", filename="#outputPath#", overwrite="yes" );
        result = outputPath;
    }
    catch (any excpt) {
        result = excpt;
    }
    return serializeJSON(result);
}
```

# practical considerations

- Good use for anything with repetitive, small elements
  - Name tags
  - Tickets
  - Asset tags for IT items managed in CMDB
- Best to create a CFC that takes template and query as arguments
  - You can use same dataset with multiple files
  - (I printed item labels with just names, and shipping labels)
- Don’t persist the PDF, only the data

# cfreport example

- Quick demo:
  - Log into http://training.latreia.com as cfug / seattle
  - My courses > Fun with PDF
  - Click ‘eyecon’ on **cfreport** to expand the topic
  - Click demo of cfreport.
- Click button ‘Generate Labels!’

# cfpdf

Manipulating existing PDFs

# cfpdf

- Use cases
  - Merge documents into single PDF
  - Apply/remove header/footer
  - Apply password/encryption
  - Apply/remove watermark
  - Output to file system
  - Create thumbnails
  - Extract text or images

Believe it or not, I have had occasion to do almost all these and more, for both corporate customers and Defense contract work.

# cfpdf

- <cfpdf>
  - The 'Swiss Army knife' of PDF manipulation
  - Used for manipulating **existing PDF files**, rather than building them
  - Many options, some totally bizarre *(watermarks, redaction, etc.)*
- Additional feature tag
  - <cfpdfparam>
    *Passes parameters for merge operations*

The code examples for CFPDF take up more lines of code than the crew compliment of NCC-1701 (431).

Check out the documentation here:

[https://helpx.adobe.com/coldfusion/cfml-reference/coldfusion-tags/tags-p-q/cfpdf.html](https://helpx.adobe.com/coldfusion/cfml-reference/coldfusion-tags/tags-p-q/cfpdf.html)

# cfpdf process ddx

- One of the more useful and bizarre features: **Dynamic Document Exchange**
    - Subset of the XML used in Adobe LiveCycle® Enterprise Suite
    - Bizarre: Proprietary *(AKA non-standard)* XML format
    - Useful: Assemble multiple PDF documents into a book
        - Headers and footers
        - Cover
        - Table of contents
        - Section breaks, etc.

# DDX

- Basic code looks deceptively simple:
  - XML namespace
  - PDF result location (“Out1”)
  - A couple documents
- Can add many features in the DDX
- But…
  - The DDX file is processed top to bottom, so necessary to add items in sequence… except styles
  - DDX is case-sensitive

```
<DDX xmlns="http://ns.adobe.com/DDX/1.0/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance"
xsi:schemaLocation="http://ns.adobe.com/DDX/
1.0/ coldfusion_ddx.xsd">
<PDF result="Out1">
  <PDF source="Doc1"/>
  <PDF source="Doc2"/>
</PDF>
</DDX>
```

Note the space!

https://www.adobe.com/devnet/coldfusion/articles/ddx_pdf.html

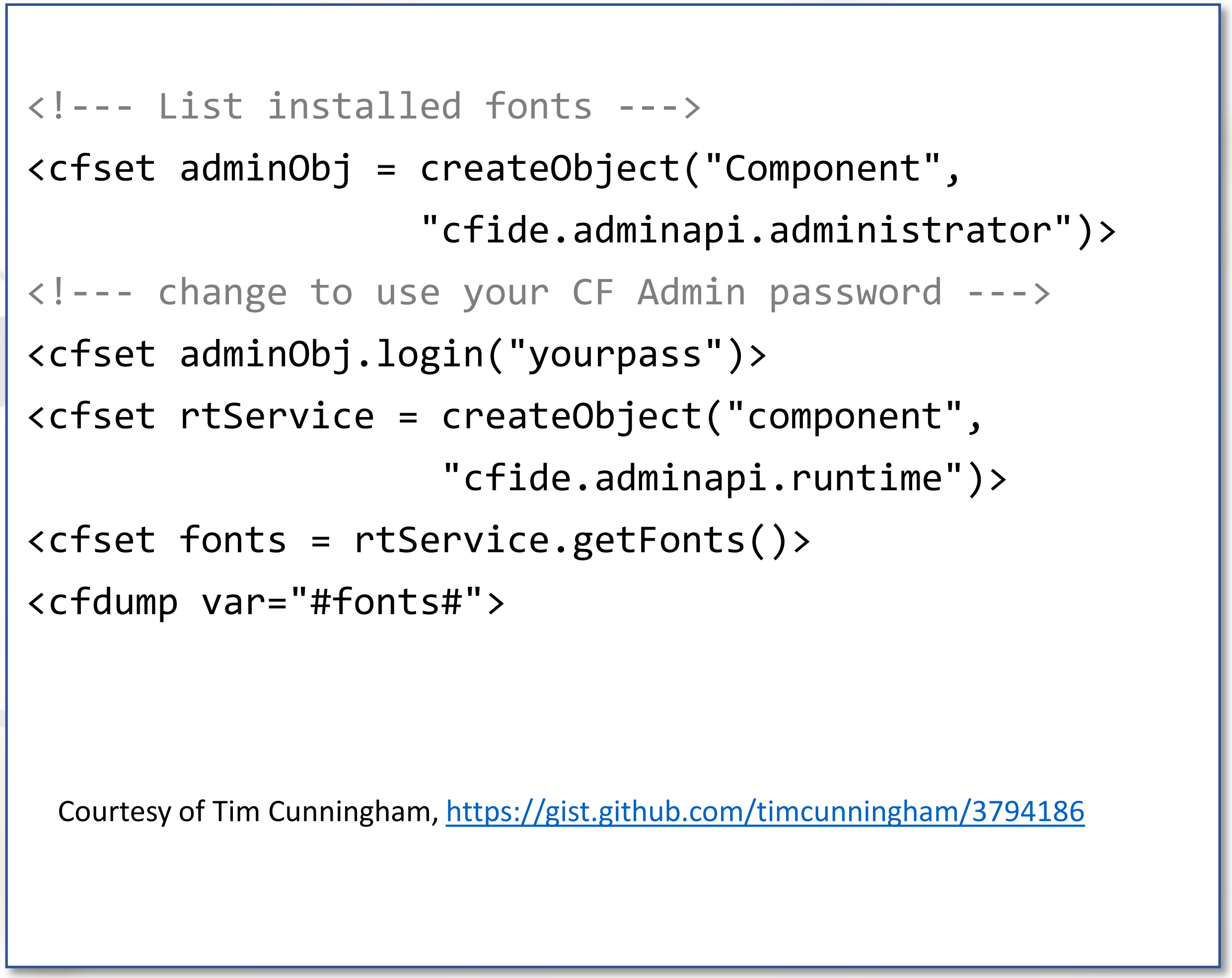
# cfpdf process ddx

- General process:
  1. Generate or assemble PDF docs *(if they are going into a TOC, be sure to set bookmarks)*
  2. Create the DDX file with the docs listed in order
  3. Validate the DDX file
  4. Add the docs to an inputStruct
  5. Define the destination in an outputStruct
  6. Use CFPDF to process the DDX
  7. Pat yourself on the head if it works.

```
<!--- confirm DDX is valid. --->
<cfif IsDDX("makeBook.ddx")>
   <!--- Create a structure for the input files. --->
   <cfset inputStruct=StructNew()>
   <cfset inputStruct.Title="Title.pdf">
   <cfset inputStruct.Doc1="Chap1.pdf">
   <cfset inputStruct.Doc2="Chap2.pdf">

   <!--- Create a structure for the output file. --->
   <cfset outputStruct=StructNew()>
   <cfset outputStruct.Out1="Book.pdf">

   <!--- Process DDX to generates the book. --->
   <cfpdf action="processddx" ddxfile="makeBook.ddx"
          inputfiles="#inputStruct#"
          outputfiles="#outputStruct#"

          name="myBook">
</cfif>
```

# cfpdf process ddx

- Another thing to remember: You can only use fonts installed ***on the server*** for text that is rendered by means of ddx.
    - (Headers, footers, etc.)



```
<!--- List installed fonts --->
<cfset adminObj = createObject("Component",
                  "cfide.adminapi.administrator")>
<!--- change to use your CF Admin password --->
<cfset adminObj.login("yourpass")>
<cfset rtService = createObject("component",
                   "cfide.adminapi.runtime")>
<cfset fonts = rtService.getFonts()>
<cfdump var="#fonts#">
```

Courtesy of Tim Cunningham, https://gist.github.com/timcunningham/3794186

# DDX

- Before attempting to do DDX, read Adobe online docs!

  https://helpx.adobe.com/coldfusion/cfml-reference/coldfusion-tags/tags-p-q/cfpdf.html

  https://helpx.adobe.com/coldfusion/developing-applications/working-with-documents-charts-and-reports/assembling-pdf-documents/using-ddx-to-perform-advanced-tasks.html

# cfpdf example

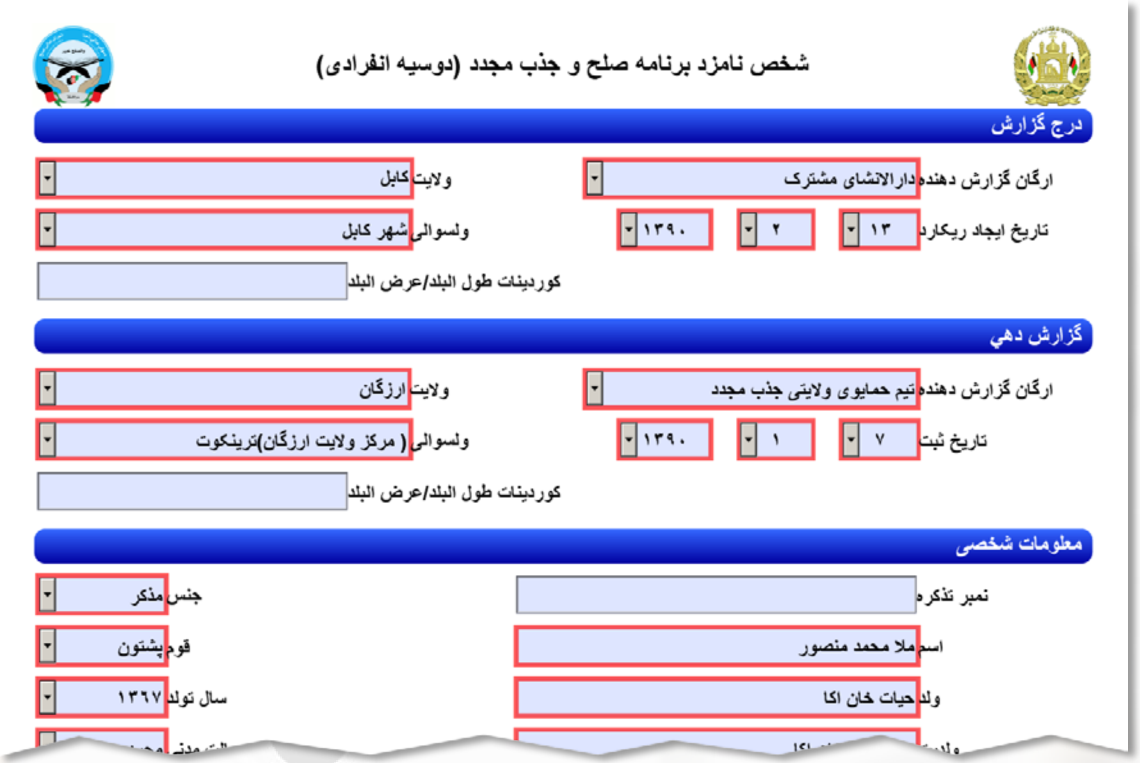
- Quick demo:
  - Log into http://training.latreia.com as cfug / seattle
  - My courses > Fun with PDF
  - Click 'eyecon' on **cfpdf** to expand the topic
  - Click on demo of cfpdf.
- Enter your name
- Click button 'Build the brochure!'

# cfpdfform

Data into and out of PDF forms

# cfpdfform

- PDF forms can be a great way to capture data offline
  - Ingest data online
- Forms are ubiquitous
- Many applications, from HR forms to surveys



شخص نامزد برنامه صلح و جذب مجدد (دوسیه انفرادی)

**درج گزارش**

ارگان گزارش دهنده دارالانشای مشترک — ولایت کابل

تاریخ ایجاد ریکارد ۱۳ ▾ ۲ ▾ ۱۳۹۰ ▾ — ولسوالی شهر کابل

کوردینات طول البلد/عرض البلد

**گزارش دهي**

ارگان گزارش دهنده تیم حمایوی ولایتی جذب مجدد — ولایت ارزگان

تاریخ ثبت ۷ ▾ ۱ ▾ ۱۳۹۰ ▾ — ولسوالی ( مرکز ولایت ارزگان)ترینکوت

کوردینات طول البلد/عرض البلد

**معلومات شخصی**

نمبر تذکره — جنس مذکر

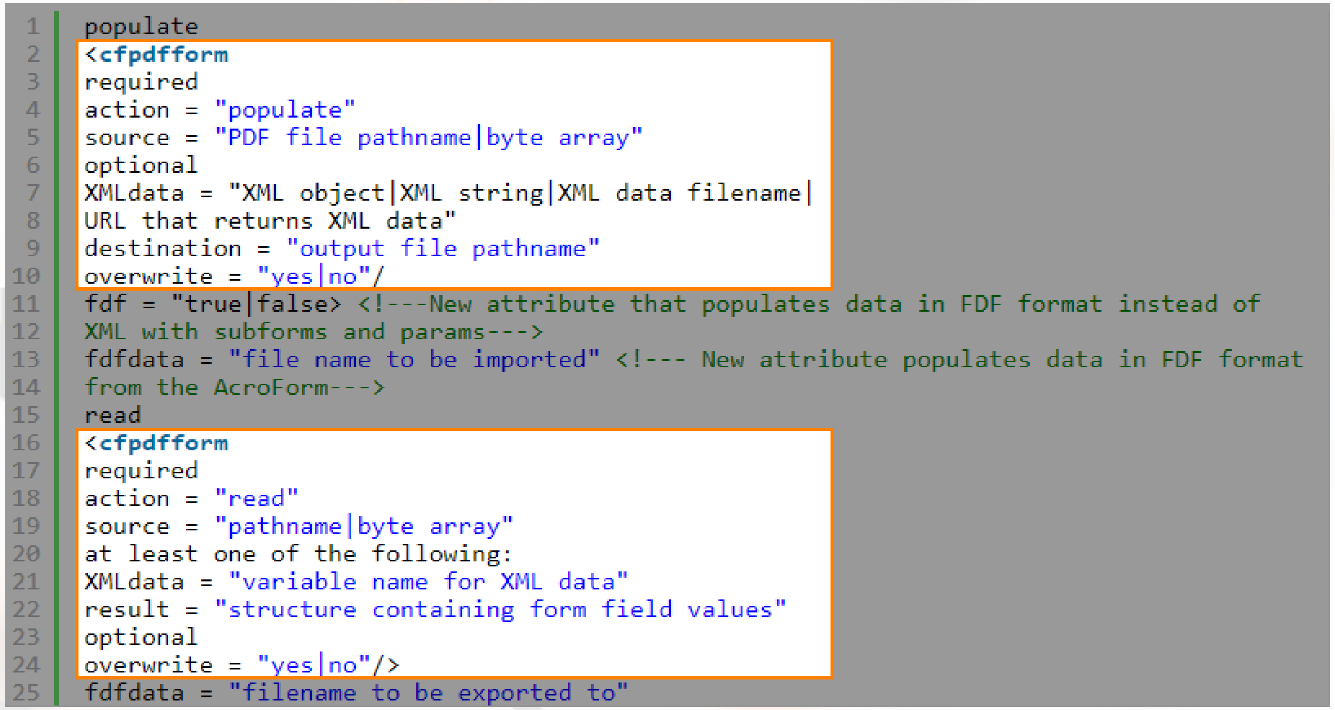
اسم ملا محمد منصور — قوم پشتون

ولد حیات خان اکا — سال تولد ۱۳۶۷

Portion of GIROA PDF – Reintegration / Government ID Card Application
*(individual from a village in Shahidi Hassas district of Uruzgan Province)*

# cfpdfform

- Populate fields on PDF form
- Read data from PDF forms
  - XFA or FDF
  - Store in CF struct or file
- <cfpdfformparam>
  *(for filling out forms, maps PDF form field to CF variable)*



```
populate
<cfpdfform
required
action = "populate"
source = "PDF file pathname|byte array"
optional
XMLdata = "XML object|XML string|XML data filename|
URL that returns XML data"
destination = "output file pathname"
overwrite = "yes|no"/
fdf = "true|false> <!---New attribute that populates data in FDF format instead of
XML with subforms and params--->
fdfdata = "file name to be imported" <!--- New attribute populates data in FDF format
from the AcroForm--->
read
<cfpdfform
required
action = "read"
source = "pathname|byte array"
at least one of the following:
XMLdata = "variable name for XML data"
result = "structure containing form field values"
optional
overwrite = "yes|no"/>
fdfdata = "filename to be exported to"
```

# cfpdfform

- Sadly, not in Lucee
- Ortus has an extension
  - Pricey but if you need it…
  - https://www.ortussolutions.com/products/ortuspdf

# Using cfpdfform

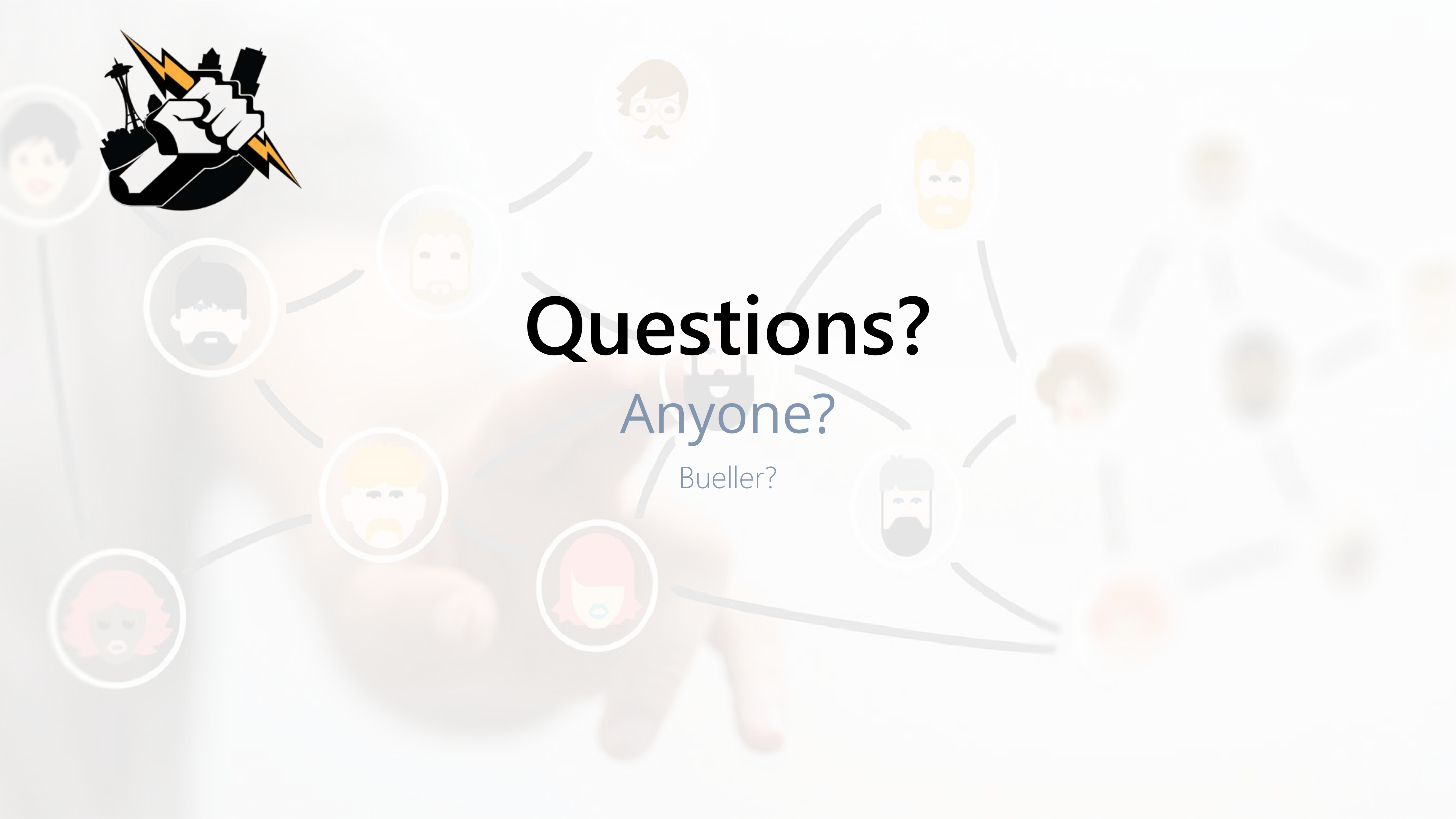
```
<cfpdfform source="#demoFile#" result="resultStruct" action="read"/>
<cfscript>
   /* WHITE LIST: No matter what is in the form file, only use input elements we trust
      Appearance, Comments, Content, Delivery, Examples, Interest, Presenter, Title
      Only Title is required */
   // String fields
   if(structKeyExists(resultStruct, 'Title') && isValid("string", resultStruct.Title) ) {
      Title = encodeForHTML(resultStruct.Title);
   }
        …etc…
</cfscript>
```

**SECURITY ALERT:**

PDF Form fields can contain ***anything***, and conceivably could be used as an attack vector. Scrub all field values and compare to a whitelist before doing anything with the data.

# cfpdfform example

- Quick demo:
  - Log into http://training.latreia.com as cfug / seattle
  - My courses > Fun with PDF
  - Click ‘eyecon’ on cfpdfform to expand the topic
  - Click on demo of cfpdfform.
  - Download form
  - Fill out and save *(open in Acrobat)*
  - Upload form
- Click button ‘Ingest data!’



# Questions?

Anyone?

Bueller?

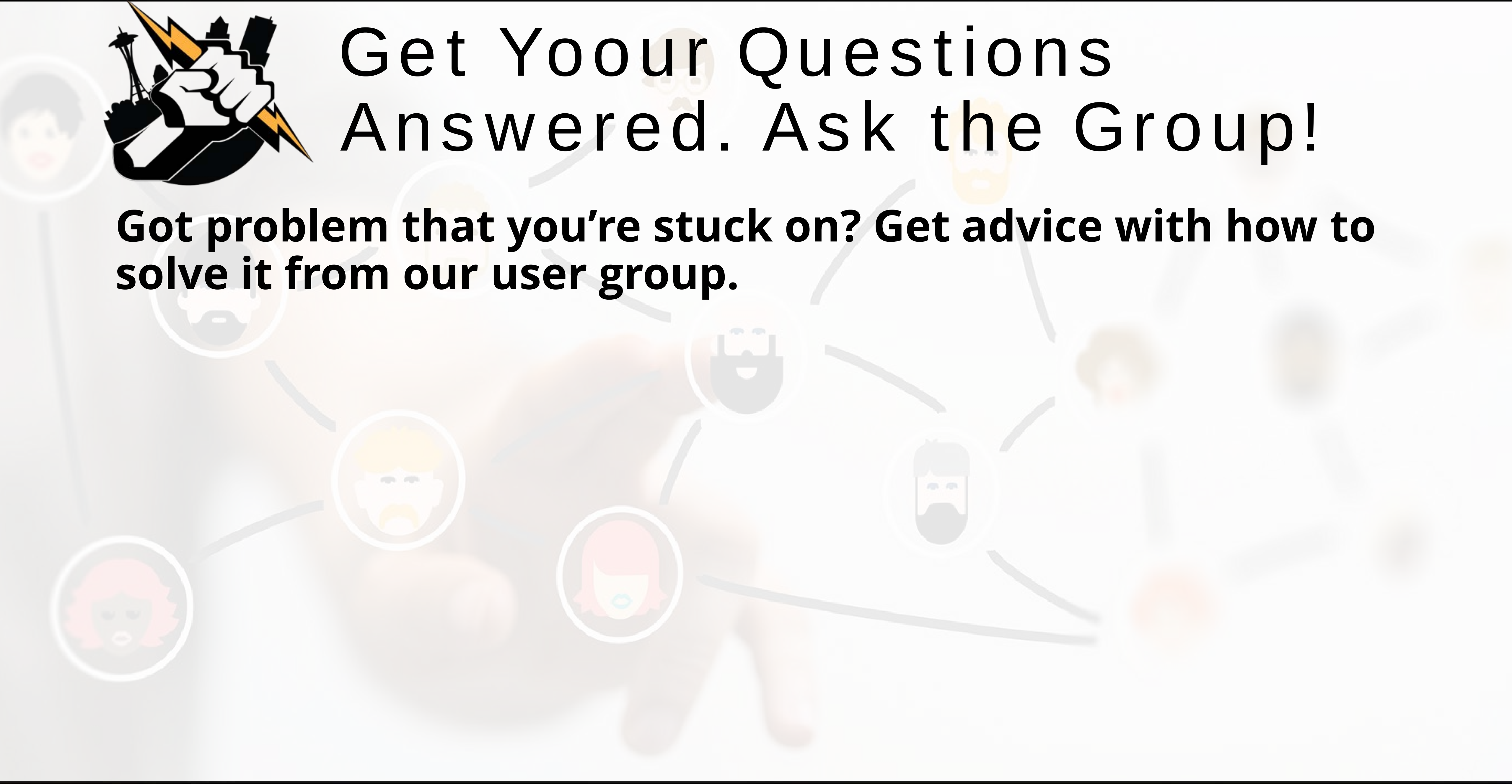
# thanks!

- Training Portal – This was an experiment, so may not leave up indefinitely.  Feel free to grab what you can.
  - http://training.Latreia.com
- Email me with questions any time
  - jim@Latreia.com

I think we’ve heard enough for tonight. May we please go home?

# Get Yooour Questions Answered. Ask the Group!

**Got problem that you’re stuck on? Get advice with how to solve it from our user group.**



# Next Month's Meeting

August 15, 2018 – WeWork – Lincoln Square
(if WeWork will have us back)

What would you like us to present on?