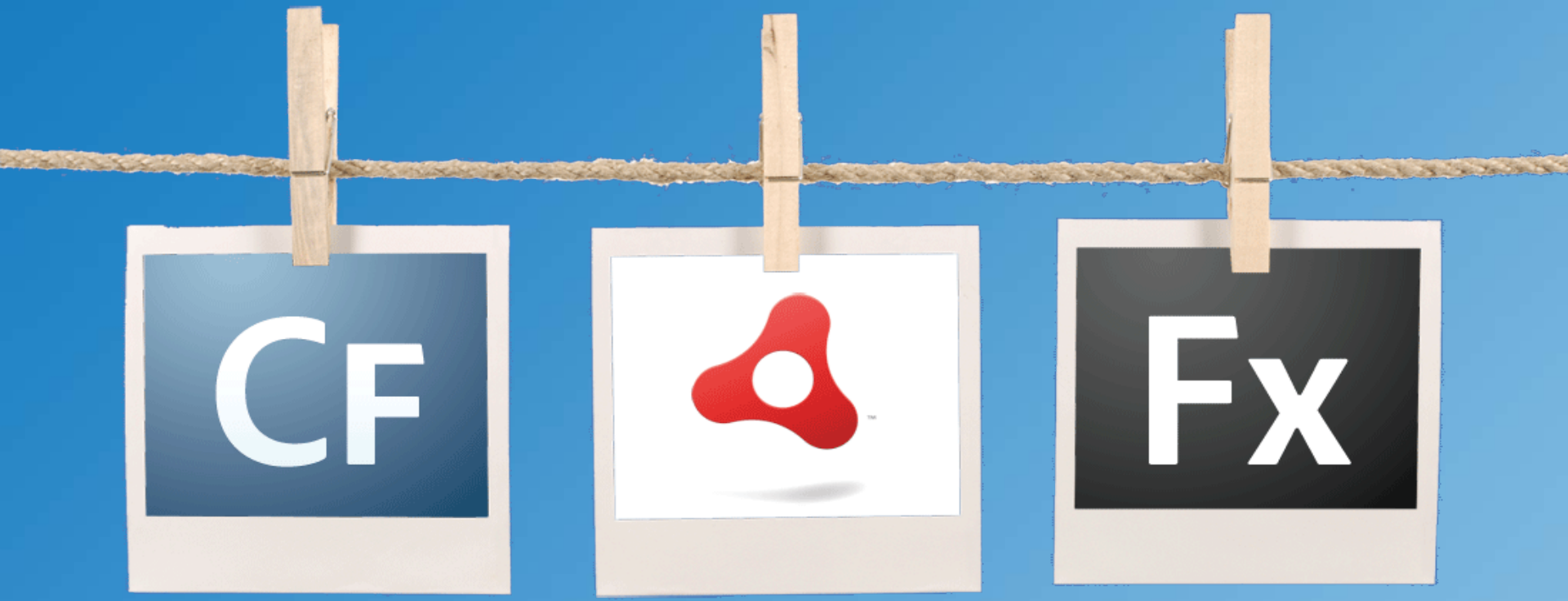


ColdFusion Integration:
Java, WebServices and XML

4th-6th June 2008
Edinburgh, Scotland

SCOTCH ON THE ROCKS

About this presentation

- My first talk covered
  the challenges of the enterprise.
- Enterprise needs integration technologies:
  - Java
  - XML
  - Web Services
- In this presentation
  - Illustrate the power of ColdFusion
    integration technologies for Java and XML

Overview

- Using JSP Tag Libraries
  - Globalization
- Using Java Libraries
  - Enabling Acrobat Shared Reviews with itext
- Calling Web Services:
  - Acrobat Reader Extensions with LiveCycle ES
- Using XML
  - Single Sign On & Apache Java Security Library

Using JSP Tags: Globalization

Globalization an overview

- First some terms
- Internationalization
  - i18N: Preparing your code to support different
    - Languages
    - Time zones, Time formats
    - Calendars
    - Number formatting
    - Monetary symbols
- Localization
  - L10N
    - Implamanting or applying for a specific language or country
- Globalization
  - G11n = i18N + L10N

Globalizing a CF Application

- Fonts
  - Unicode font
- ColdFusion pages

```
<cfprocessingdirective pageEncoding="utf-8">
  <cfscript>
    SetEncoding("form","utf-8");
    SetEncoding("url","utf-8");
  </cfscript>
<cfcontent type="text/html; charset=utf-8">
```

- ColdFusion tags

```
<cfmail charset=“utf-8”
```

- Database
  - MS SQL Server: nvarchar
  - Mysql: utf-8

Globalizing Display Text

- ColdFusion has good support for
  - Time zones, Time formats
  - Calendars
  - Number formatting
  - Monetary symbols
- To support display text
  - Roll your own
  - Use Java tag library
- Go through all existing pages
  Replace all display text with <fmt> tags

FMT Example - Before

```
<p>
You have #failureCount# failed logins.
You are locked out until
  #dateFormat(until,'dddd dd mmm')#
@ #timeFormat(until,'h:MM tt')#
</p>
```

FMT Example - After

```
<cfimport taglib="/WEB-INF/fmt.tld" prefix="fmt">
<p>
<fmt:message key="FailedLogin">
   <fmt:param>
       #FailureCount#
   </fmt:param>
   <fmt:param>
       #lsdateFormat(Until,short')#
   </fmt:param>
   <fmt:param>
       #lstimeFormat(Until,short')#
   </fmt:param>
</fmt:message>
</p>
```

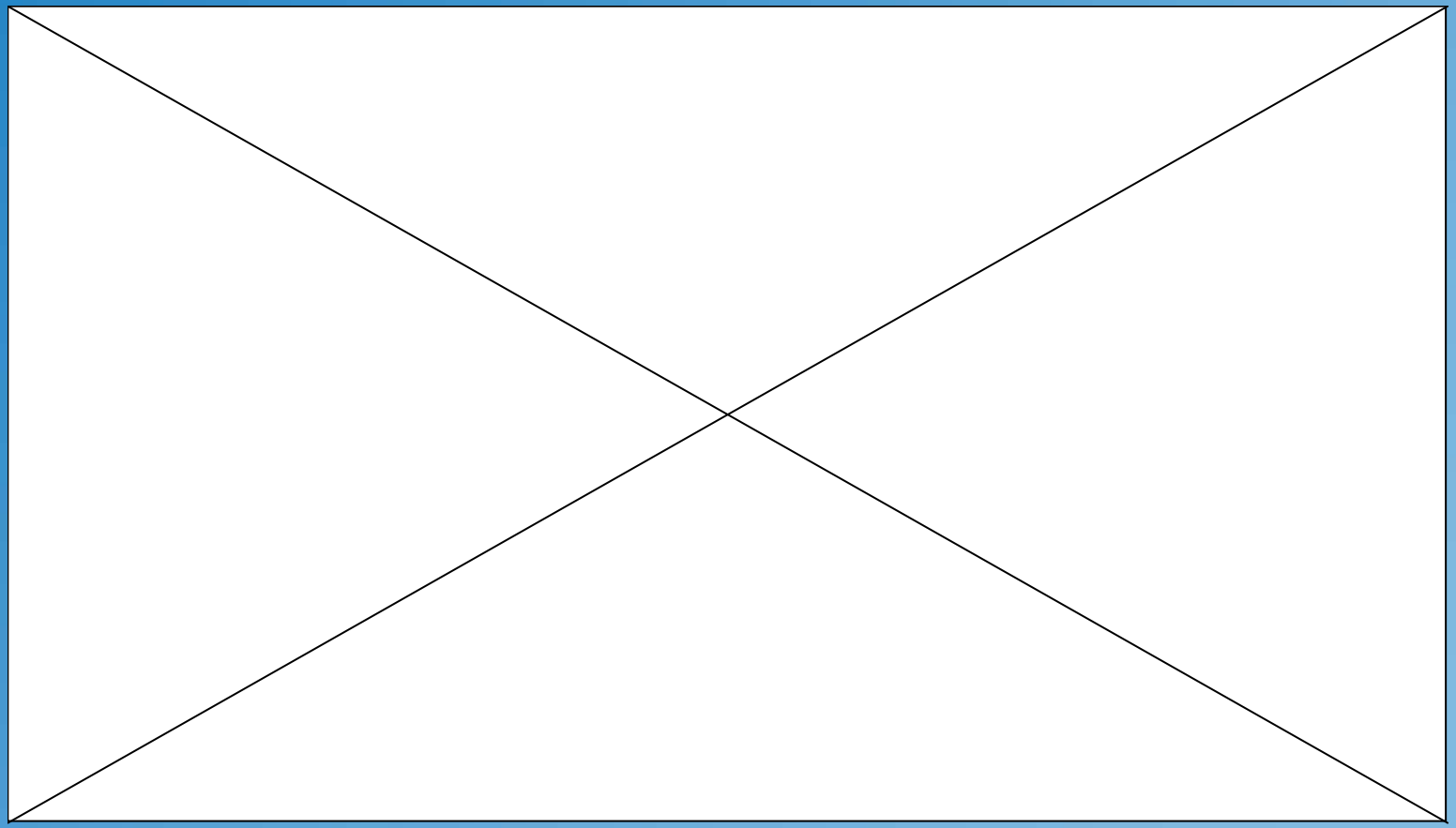
Locale Files

- Create locale files for specific languages
  - We use BabelFish
- Locale files allow parameterized strings

```
FailedLogin  = You have {0} failed logins.
  You are locked out until {1} @ {2}
Sie haben das Limit von {0}
  aufeinander folgenden fehlgeschlagenen
  Anmeldeversuchen \u00FCberschritten und wurden
  bis {1} @ {2} automatisch gesperrt
Przekroczy\u0142e\u015B limit {0}
  nieudanych pr\u00F3b zalogowania si\u0119
  i Tw\u00F3j login zosta\u0142 automatycznie
  zablokowany do {1} @ {2}
```

ColdFusion <fmt> Tags

- [Illustration]



Using Java Libraries: iText

4th-6th June 2008
Edinburgh, Scotland

SCOTCH ON THE ROCKS

Using Java Libraries

- iText is a library to manipulate Pdfs
- iText is part of CF8 (used by CFPDF)
- iText can add javascript to a PDF
- Tag: uses it to automate PDF reviews
- The following is a simple example (see my blog for more)

Using iText in JSP

```
<%@page import="...itext.jar"%>
<%
String ipFileName = "...“;
String jscript = "...";
String scriptName = "...";

PdfReader reader = new PdfReader(ipFileName );
PdfStamper stamper= PdfStamper(reader,opFile);
PdfWriter writer = stamper.getWriter();
writer.addJavascript(scriptName, jscript);
stamper.close();
%>
```

Using iText in ColdFusion

- [Illustration]

Illustration iText in action

- [Illustration/demo]

WebServices: Livecycle ES

4th-6th June 2008
Edinburgh, Scotland

SCOTCH ON THE ROCKS

Using WebServices

- LiveCycle DS = Data Services (Flex)
- LiveCycle ES = Enterprise Services
  - PDF Workflow tool
  - Webservices
  - Email Inbox
  - Watch Folders
- Contains document transformations
  - PDFGenerator (Word, Excel -> PDF)
  - Acrobat Reader Extensions
- Transformations have WebService I/f

Calling Livecycle

- Services defined by wsdl
- ColdFusion hides the complexity of WSDL
- Calling web service:

```
<cfscript>
// Create WebService
  rExtWS = createObject("webservice", wsdl, creds);

//Call method
  resp = rExtWS.applyUsageRights
  (inPDFDoc, uName, pass, options);

//Convert result from 64encoding
  pdfBinary = resp .binaryData;
</cfscript>
```

Match the WSDL

```
<complexType name="BLOB">
    <sequence>
        <element name="contentType" type="xsd:string"/>

        <element name="binaryData“
        type="xsd:base64Binary"/>

        <element name="attachmentID" type="xsd:string"/>
        <element name="remoteURL" type="xsd:string"/>
    </sequence>
</complexType>
```

WSDL -> ColdFusion

ColdFusion

```
inPDFDoc = structNew();

inPDFDoc['binaryData'] = pdfBase64;
inPDFDoc['contentType'] = "application/pdf";
inPDFDoc['attachID'] = javacast('null','');
inPDFDoc['remoteURL'] = javacast('null','');
```

WebService Hints

- The parameters have to match exactly
- Carefully check the WSDL
- and check the Java

```
creds = structnew();
creds['username']=‘reader;
creds['password']='password';
creds['saveJava']= TRUE;       // FALSE in prod
creds['refreshWSDL']= TRUE;    // FALSE in prod

rExtWS = createObject("webservice", wsdl, creds );
```

WebServices Hints :2

- Conversion of CF types to WS types is good
- When it doesn't work
  - Use javacast() for simple types
  - Use java objects for complex types
- ColdFusion uses HTTP/1.0
  - Usually OK
- It doesn't work when:
  - When calling MS SQL Server Web Services
  - OR you need NTLM authentication
- You need to hack the stack
  - Use Jakarta Commons HTTPClient library

Calling Livecycle: Illustration

- [Demo/Illustration]

Using XML: SSO

Single Sign On: Overview

- Biggest usability feature
- Users want to have just one username and password, for everything
- They want to use their corporate password on our (external) systems
- Implementations frequently use SAML
- SAML is an XML format

SSO: SAML

- Start with user identity

```
<saml:AuthenticationStatement>
  <saml:Subject> <saml:NameIdentifier>
    DavidRutter
  </saml:NameIdentifier> </saml:Subject>
</saml:AuthenticationStatement>
```

- Add time stamp

```
<saml:Assertion IssueInstant="2007-11-04T14:04:24Z">
   <saml:Conditions NotBefore=“…" NotOnOrAfter=“…"/>
   <saml:AuthenticationStatement>
       <saml:Subject><saml:NameIdentifier>
       DavidRutter
       </saml:NameIdentifier></saml:Subject>
   </saml:AuthenticationStatement>
 </saml:Assertion>
```

SSO: SAML (2)

- Create a digest to prevent tampering,

```
<ds:DigestValue>
/UlguevI2sppqGHnuZQV
</ds:DigestValue>
```

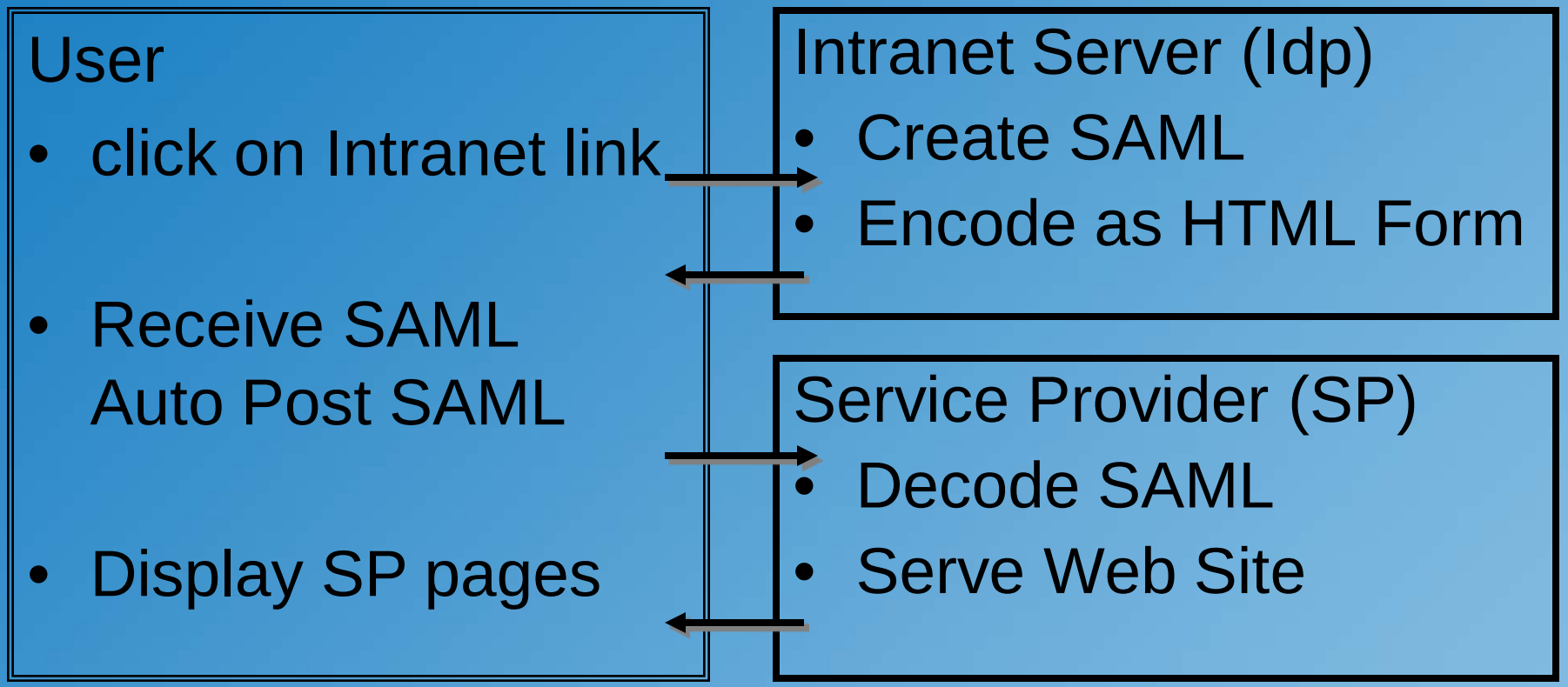
- Sign the digest using private key

```
<ds:SignatureValue>
ID0Pr3EMyqvLilnZ0
</ds:SignatureValue>
```

- Add identity of organization's user

```
<ds:X509Certificate>
MIIC6jCCAd…
</ds:X509Certificate>
```

SSO: POST Profile



SSO in ColdFusion

- Several commercial and open source implementations
- None are built into ColdFusion
- The following uses
  - Open Source Apache XMLSecurity library
  - Java and C++ implementations

SAML in CF Idp:1

- Get variables, e.g. from database

```
<cfscript>
   id = "drutter@tag.com";
   email ="drutter@tag.com";
   givenname = "David";
   surname = "Rutter";
   AssertionID = CreateUUID();
   nowDateTime = …
   nowDateTimePlus1 = …
</cfscript>
```

SAML in CF Idp:2

```
<cfoutput>
<cfxml variable="samlXML">
<samlp:Response>
   <saml:Assertion AssertionID="#AssertionID#"
   IssueInstant="#nowDateTime#">
        <saml:Conditions NotBefore="#nowDateTime#“
        NotOnOrAfter="#nowDateTimePlus1#"/>
        <saml:AuthenticationStatement>
        <saml:Subject>
        <saml:NameIdentifier>#id#</saml:NameIdentifier>
        </saml:Subject>
        </saml:AuthenticationStatement>
   </saml:Assertion>
</samlp:Response>
</cfxml>
</cfoutput>
```

SAML in CF Idp:3 Signing

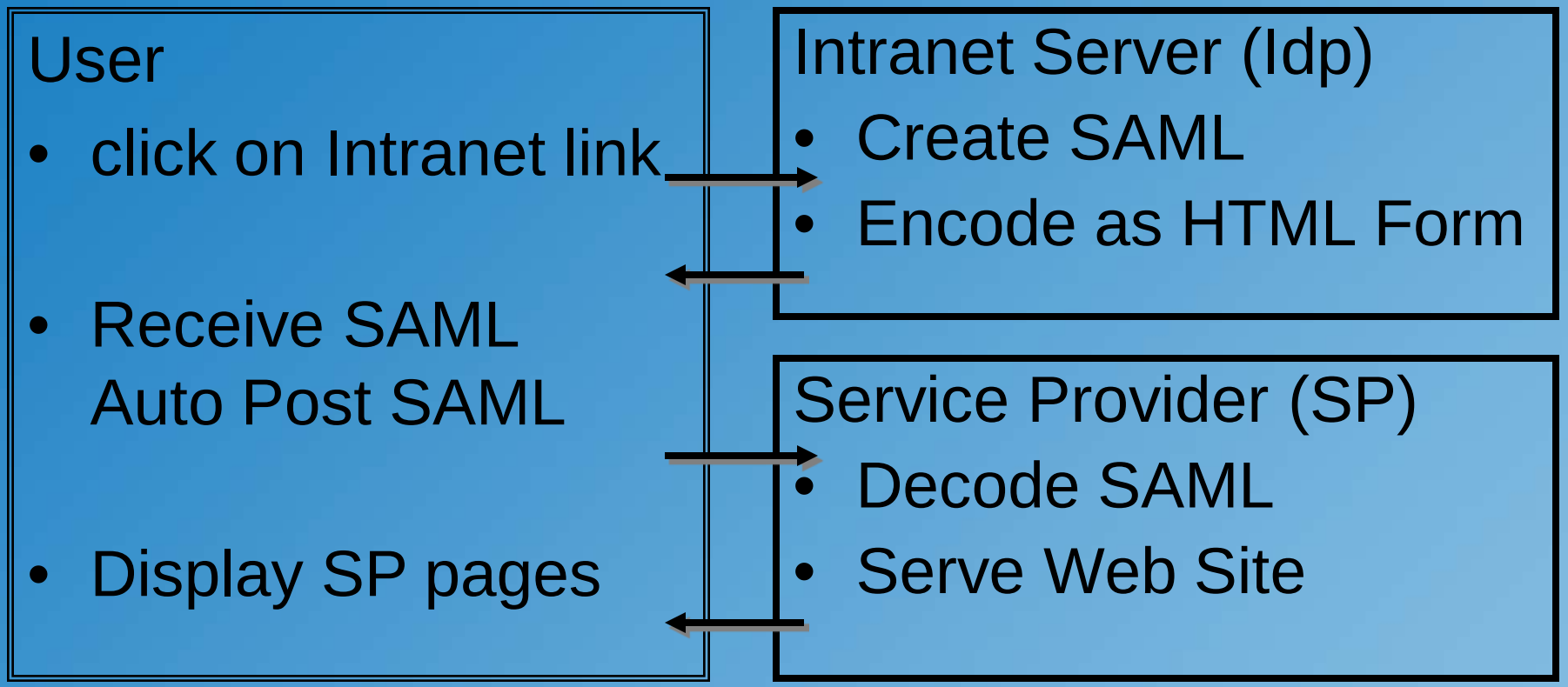
```
XMLSigClass = CreateObject("Java",
    "org.apache.xml.security.signature.XMLSignature");

// Find the right parts of the XML
samlAElem = samlXML.getDocumentElement();
samlADoc = samlAssertionElement.GetOwnerDocument();

// Sign the document
sig = XMLSignatureClass.init(samlADoc , “”, sigType);
samlAElem.insertBefore(sig.getElement(), samlAElem.getFirstChild());
sig.addKeyInfo(publicKey);
sig.sign(privateKey);

// 64Bit Encode for embedding in HTML Form
encoded=BinaryEncode(CharsetDecode(samlXML,"utf-8"),"Base64");
```

SSO: POST Profile



SAML in CF SP:4 Parsing

```
// Service Provider: Decode
samlXML=CharsetEncode
   (BinaryDecode(encoded,"Base64"),"utf-8");

// Parse the XML doc for Sig info
doc=XmlParse(variables.xmlResponse).getDocumentElement();
sigElem = doc.getElementsByTagNameNS
   (SigNS,"Signature").item(0);
SignatureSpecNS=CreateObject(…).SignatureSpecNS;
xmlSignature = CreateObject(...).init(sigElem,nullStr);
keyInfo=xmlSignature.getKeyInfo();
keyResolver=CreateObject(…).init();
keyInfo.registerInternalKeyResolver(keyResolver);
x509cert = keyInfo.getX509Certificate();

// Verify Signature
isValid = xmlSignature.checkSignatureValue(x509cert);
```

SAML in CF SP:5 Extract

```
// Extract Validity Info
ssoissuer = getSVal(XPathIssuer);
cert = getSVal(XPathCertificates).XmlText;
reference = getSVal(XPathUniqueReference);
c = getSVal(XPathConditions);
before=DateConv(c.XmlAttributes.NotBefore);
after=DateConv(c.XmlAttributes.NotOnOrAfter);

// Verify Validity Info
// ...

// Extract User
ssouser =  getSVal(XPathNameId).xmltext;
```

Conclusion

- We have covered integration technologies
  - JSP Tag Libraries
  - Java Libraries
  - WebServices
  - XML
- We looked at
  - Globalization
  - iText
  - LiveCycle ES
  - SSO
- If you want to know more see my Blog
  - http://blog.tagworldwide.com