NotesHTTP and NotesJSON

Tips, Tricks, Traps, and Testing

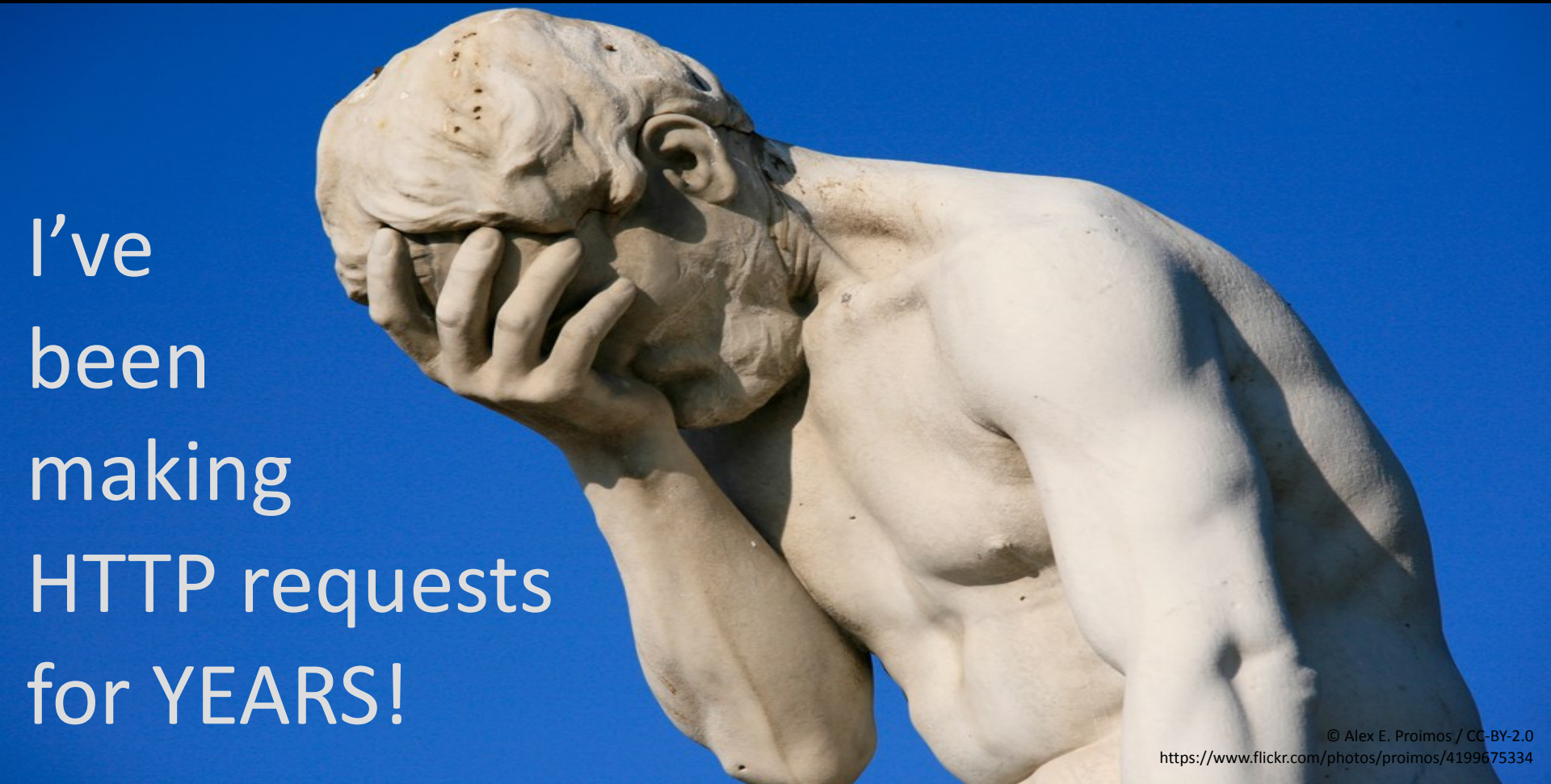
by Julian Robichaux

panagenda
Senior Software Engineer
www.panagenda.com

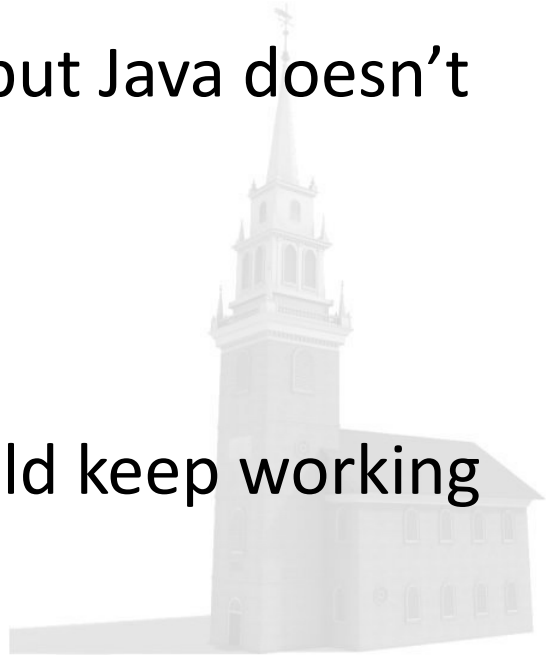
- Rules of this presentation
  - I will go very fast
  - You can download the slides and go over them later
  - Download link will be on the last slide
  - I tested everything on Notes 10.0.1 FP2
  - I tried very hard to be right, but tell me if something is wrong
  - Things might change in Notes 11 and beyond

- NotesHTTPRequest
  - https://www.ibm.com/support/knowledgecenter/en/SSVRGU_10.0.1/basic/H_NOTES_HTTPREQUEST_CLASS.html
- NotesJSON
  - https://www.ibm.com/support/knowledgecenter/en/SSVRGU_10.0.1/basic/H_NOTESJSONNAVIGATOR_CLASS.html
- DQL
  - https://doc.cwpcollaboration.com/appdevpack/docs/en/domino-query-language.html
- Proton task (node.js)
  - https://doc.cwpcollaboration.com/appdevpack/docs/en/proton-admin.html

we won't be talking about DQL and Proton, but they're REALLY COOL!



- No weird JS2J or COM object calls required
- Some places where LotusScript works but Java doesn't
- Works with **Nomad** clients
- but… anything that already works should keep working



HTTP Requests: the Basics

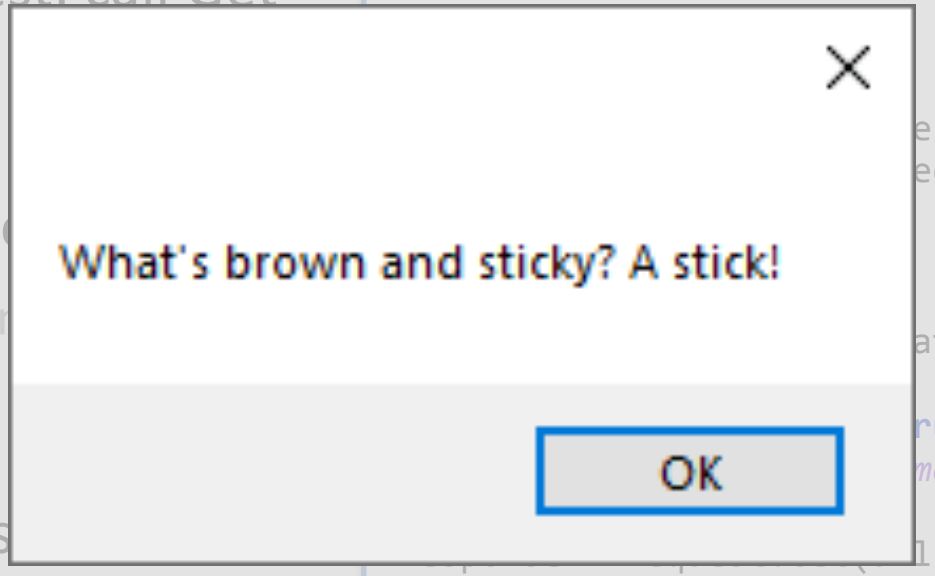
- Small node.js application
- Dad Jokes!
  - jokes courtesy of https://www.boredpanda.com/funny-dad-jokes-puns
- Included with the test database if you want to play around with it
  - http://nsftools.com/presentations/NotesHTTP.zip

- Create a request, call **Get**
  - easy peasy
- **TimeoutSec** is optional
  - nice for testing, default is 30 seconds
- **PreferStrings** is important
  - otherwise you might get a byte array back!

```
Dim session As New NotesSession
Dim request As NotesHTTPRequest
Dim url As String
Dim response As String

Set request = session.CreateHTTPRequest()
request.TimeoutSec = 5
request.PreferStrings = True
url = "http://dadjokes.demo.local:8080/getjoke"

response = request.Get(url)
MessageBox response
```

A Simple Request: GET a Web Page

- Create a request, call Get
  - easy peasy
- TimeoutSec is
  - nice for testi
    30 seconds
- PreferStrings is
  - otherwise you might get a byte array back!




```
ession
equest

ateHTTPRequest()

rue
mo.local:8080/getjoke"

1)
MessageBox response
```

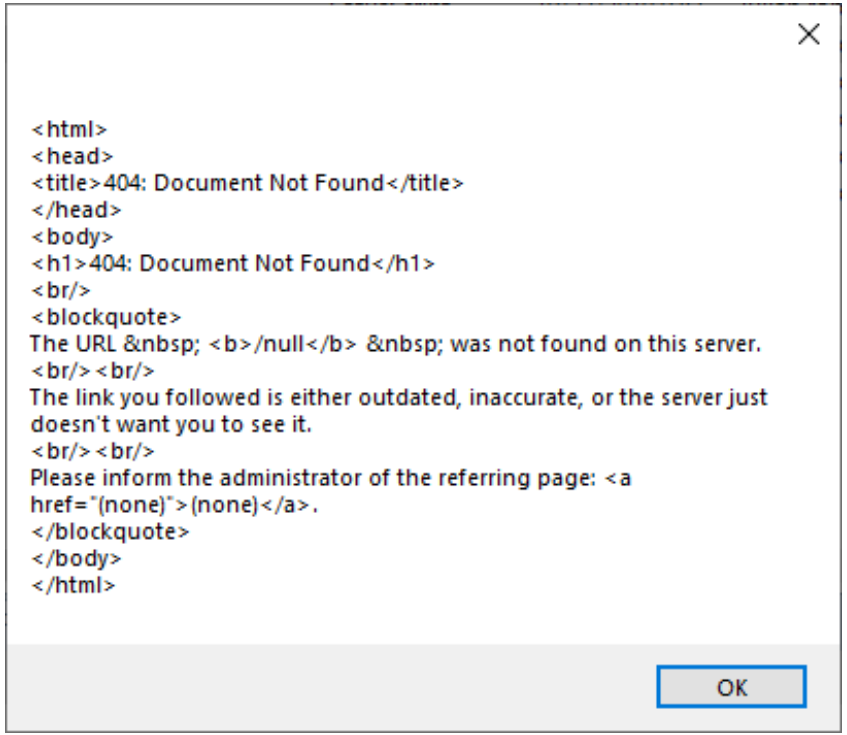
A Simple Request: POST to a Web Page

- Create a request, call **Post**
  - OMG this is so simples!
- Also **Put** and **DeleteResource**
  - but those are pretty weird

```
Dim session As New NotesSession
Dim request As NotesHTTPRequest
Dim url As String
Dim response As String

Set request = session.CreateHTTPRequest()
request.TimeoutSec = 5
request.PreferStrings = True
url = "http://dadjokes.demo.local:8080/sendjoke"
body = "People say they pick their nose, " &_
       "but I was born with mine"

response = request.Post(url, body)
MessageBox response
```

- Use **request.ResponseCode** to check return codes
  - **String**, not Integer!
  - I don't know why, I'm just the messenger…
- Common response codes
  - **200** = OK
  - **404** = Not Found
  - **400** = Bad Request
  - **500/503** = Server Error
  - **301/307/308** = Redirect
  - **401** = Not Authorized
  - **403** = Forbidden
  - **418** = I'm a Teapot (RFC 2324)

- **Always** check the response code!
- Non-200 responses often return **HTML error text**



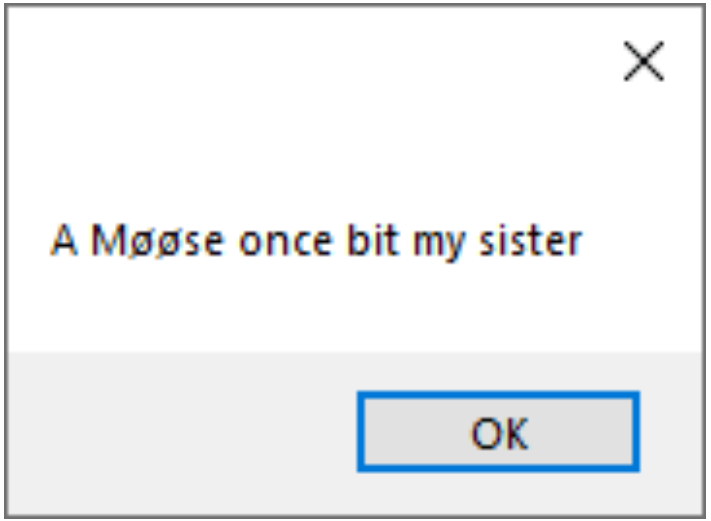
- NotesHTTPRequest **doesn't use the proxy servers** set up in your Location document

- If there's a way to make it use a proxy, I don't know what it is

Hellö Wörld

- Everything always works when you use **ASCII** characters
- When you test, **add an ümlaüt!** See what breaks.
  - Why? In ISO-8859-1/Latin1/Win-1252 encoding, **ö = 0xF6** and **ü = 0xFC**, which are invalid in UTF-8
- Luckily the web has standardized on **UTF-8**
  - But more on this in a moment…

- Incoming text converted for you




```
Dim session As New NotesSession
Dim request As NotesHTTPRequest
Dim url As String
Dim response As String

Set request = session.CreateHTTPRequest()
request.TimeoutSec = 5
request.PreferStrings = True
url = "http://dadjokes.demo.local:8080/getutf8"

response = request.Get(url)
MessageBox response
```

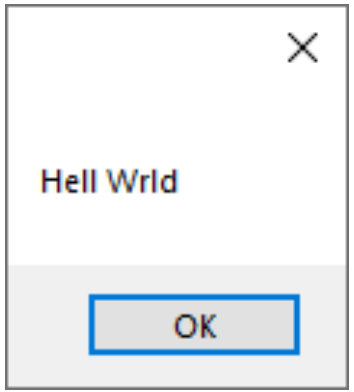
- Outgoing text always sent as UTF-8
  - used to use the platform string encoding
  - this was fixed in **Notes 10.0.1 FP2**!

```
Dim session As New NotesSession
Dim request As NotesHTTPRequest
Dim url As String
Dim response As String

Set request = session.CreateHTTPRequest()
request.TimeoutSec = 5
request.PreferStrings = True
url = "http://dadjokes.demo.local:8080/sendjoke"
body = "Hellö Wörld"

response = request.Post(url, body)
MessageBox response
```

- Just in case you ever need to **convert to UTF-8** by hand
  - you should never have to do this
  - but it might be useful somewhere else
- For really long strings, call Read() until EOF

```
Function ConvertToUTF8 (txt As Variant) As Variant
      Dim session As New NotesSession()
      Dim stream As NotesStream
      Dim db As NotesDatabase
      Dim doc As NotesDocument
      Dim body As NotesMIMEEntity

      session.ConvertMIME = False
      Set db = session.CurrentDatabase
      Set doc = db.CreateDocument
      Set body = doc.CreateMIMEEntity

      Set stream = session.CreateStream()
      Call stream.WriteText(txt)
      Call body.SetContentFromText(stream, _
                  "text/plain;charset=UTF-8", ENC_NONE)

      Set stream = session.CreateStream()
      Call body.GetContentAsBytes(stream)
      stream.Position = 0
      ConvertToUTF8 = stream.Read()    '** max 65,535 bytes
End Function
```

- **ISO-8859-1** responses might be broken
  - edge case, but be aware




```
Dim session As New NotesSession
Dim request As NotesHTTPRequest
Dim url As String
Dim response As String

Set request = session.CreateHTTPRequest()
request.TimeoutSec = 5
request.PreferStrings = True
url = "http://dadjokes.demo.local:8080/get8859"

'** response is "Hellö Wörld" ISO-8859-1 encoded
response = request.Get(url)
MessageBox response
```

- If you ever REALLY want to see the bits and bytes going across the wire, use **WireShark**
  - http://wireshark.org

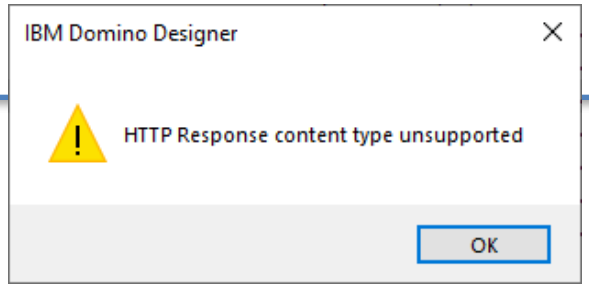
Binary Data

- Sort of works
- **64k** limit
  - Why? Largest byte array for LS
- Probably not very useful

```
Dim session As New NotesSession
Dim request As NotesHTTPRequest
Dim url As String
Dim response As String

Set request = session.CreateHTTPRequest()
url = "http://dadjokes.demo.local:8080/sendfile"

Dim stream As NotesStream
Set stream = session.CreateStream()
Call stream.Open("c:\notes\data\log.ntf", "Binary")

response = request.Post(url, stream.Read()) '** only 64k
MessageBox response
```

- Doesn't work 😭
- You'd be stuck with a 64k limit anyway

```
Dim session As New NotesSession
Dim request As NotesHTTPRequest
Dim url As String
Dim response As String

Set request = session.CreateHTTPRequest()
url = "https://www.panagenda.com/favicon.ico"

response = request.Get(url)
MessageBox response
```



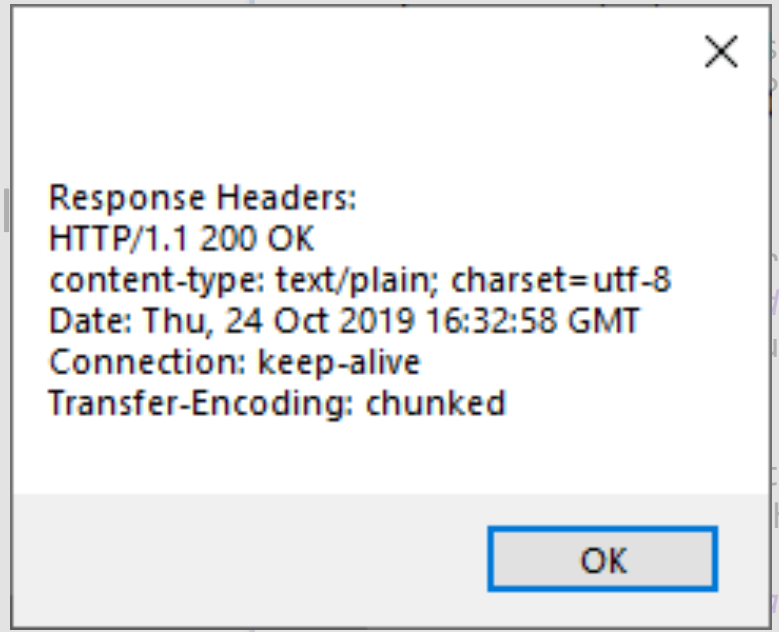
HTTP Headers

- Outgoing
  - Content-Type
  - Authorization
  - Cookie
- Incoming
  - Set-Cookie
  - Content-Security-Policy
    - upgrade-insecure-requests
  - Strict-Transport-Security

- **GetResponseHeaders** is an array of Strings
- Format of **key: value**
- Might be multiple headers for the same key!

```
Dim session As New NotesSession
Dim request As NotesHTTPRequest
Dim url As String
Dim response As String

Set request = session.CreateHTTPRequest()
url = "http://dadjokes.demo.local:8080/headers"
response = request.Get(url)

Dim rh As String
ForAll header In request.GetResponseHeaders()
    rh = rh & Chr(10) & header
End ForAll
MessageBox "Response Headers: " & rh
```

- GetResponseHeaders is an array of Strings
- Format of key: val
- Might be multiple for the same key!

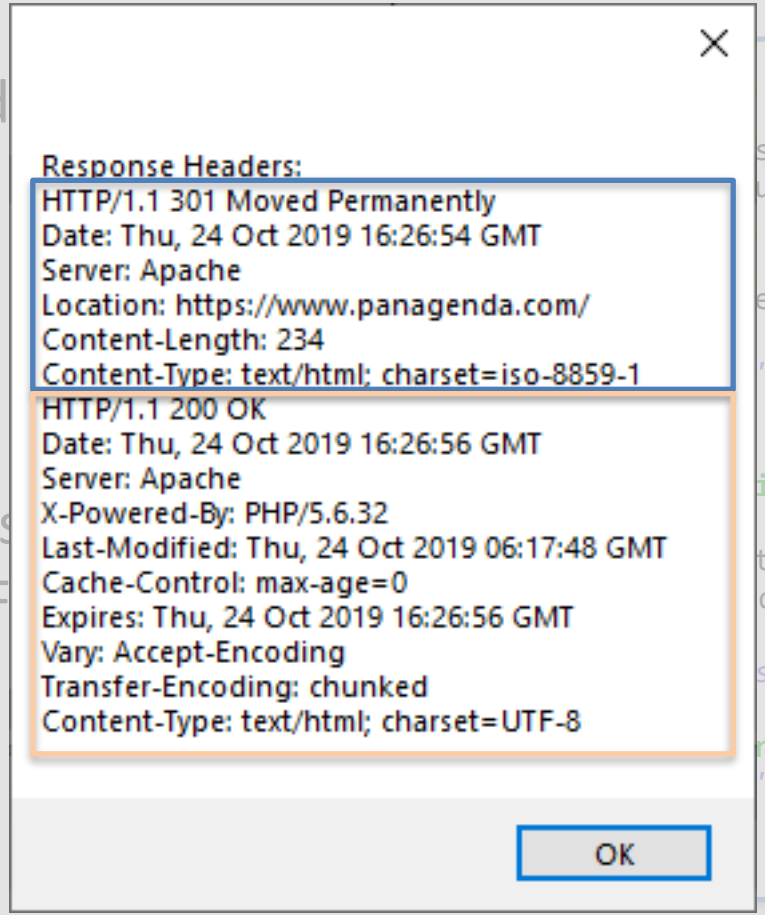



```
Session
Request

eateHTTPRequest()
demo.local:8080/headers"
url)

.GetResponseHeaders()
header

aders: " & rh
```

- Will not follow **redirects** by default
- Set **MaxRedirects**
- You will get **ALL** response headers, and the **FIRST** response code

```
Dim session As New NotesSession
Dim request As NotesHTTPRequest
Dim url As String

Set request = session.CreateHTTPRequest()
request.MaxRedirects = 5
url = "http://panagenda.com"
Call request.Get(url)

'** ALL headers for ALL redirects (in order)
Dim rh As String
ForAll header In request.GetResponseHeaders()
    rh = rh & Chr(10) & header
End ForAll
MessageBox "Response Headers: " & rh

'** FIRST response code; redirect will be a 30x
MessageBox "Response code: " & request.ResponseCode
```

- Will not follow red
by default
- Set MaxRedirects
- You will get ALL res
headers, and the F
response code



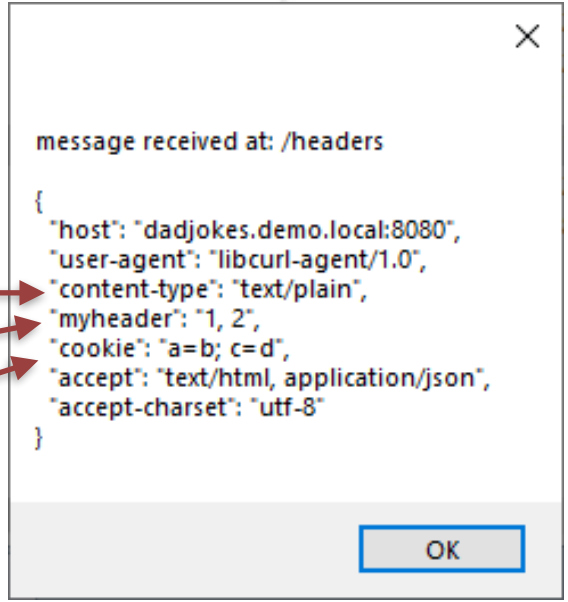

```
sion
uest

eHTTPRequest()

"

irects (in order)

tResponseHeaders()
der

s: " & rh

redirect will be a 30x
" & request.ResponseCode
```

- **SetHeaderField**
- **Not** case-sensitive
- Make sure there are **no linefeeds** in the header fields!

```
Dim session As New NotesSession
Dim request As NotesHTTPRequest
Dim url As String
Dim response As String

Set request = session.CreateHTTPRequest()
request.TimeoutSec = 5
request.PreferStrings = True
url = "http://dadjokes.demo.local:8080/xmljoke"
body = "<joke>I used to hate facial hair," & _
            " but then it grew on me</joke>"

Call request.SetHeaderField("Content-Type", _
        "application/xml; charset=utf-8")
response = request.Post(url, body)
MessageBox response
```

- Special rules if you set the same header **multiple times**
  - **Content-Length** -> always ignored (computed for Post)
  - **Content-Type** -> only the first one will be used
  - **Cookie** -> concatenated with "; "
  - **Everything else** -> concatenated with ", "

```
Call request.SetHeaderField("Content-Type", "text/plain")
Call request.SetHeaderField("Content-Type", "whatever")
Call request.SetHeaderField("MyHeader", "1")
Call request.SetHeaderField("MyHeader", "2")
Call request.SetHeaderField("Cookie", "a=b")
Call request.SetHeaderField("Cookie", "c=d")
Call request.SetHeaderField("Content-Length", "5")
```



- Basic authorization uses a **Base64-encoded** header
- Always use **SSL**!!!
  - Base64 is **NOT** encryption

```
Dim session As New NotesSession
Dim request As NotesHTTPRequest
Dim url As String
Dim response As String

Set request = session.CreateHTTPRequest()
request.PreferStrings = True
url = "http://dadjokes.demo.local:8080/auth"

Call request.SetHeaderField("Authorization", _
        "Basic " + base64("username:password"))
response = request.Get(url)
MessageBox response
```

- For Base64 I like to use an **XSD_DATATYPE_CONVERTER**
  - from the web services libraries (lsxsd.lss)
  - important to **remove linefeeds**!
  - Search the web for other LotusScript options
- Assumes **ASCII** for username and password

```
%Include "lsxsd.lss"       '** after Option Declare

Function base64 (txt As String) As String
    Dim session As New NotesSession
    Dim stream As NotesStream
    Dim i As Integer
    ReDim c(Len(txt)-1) As Byte

    For i = 1 To Len(txt)
        c(i-1) = Asc(Mid(txt, i, 1))
    Next

    '** can't use WriteText because it's UTF-16
    Set stream = session.CreateStream()
    Call stream.Write(c)

    Dim converter As New XSD_DATATYPE_CONVERTER
    Dim b64 As String
    b64 = converter.NotesStreamToBase64Ext(stream)
    base64 = Replace(b64, Chr(13)+Chr(10), "")
End Function
```

- **Always use SSL!!!**
- A website with **Basic Authentication** will generally give a response code of **401** if login fails
- A website with **Session-based Authentication** might return a **200/OK** response and a login form…

- If you use Session-based authentication on your Domino server, you can set up **Web Site Rules** to allow Basic Authentication for specific URLs

- Nice overview here:
  - https://www-10.lotus.com/ldd/ddwiki.nsf/dx/Authenticating_Domino_REST_Service_Requests

- Session-based authentication (anything with a login form) uses **cookies** to pass authentication credentials

- Servers have different options for generating the cookie
  - some require a login form
    some have a REST endpoint
    some allow Basic Authentication to create a cookie

- Use this **session cookie** in subsequent requests

Cookies!

- Browsers take care of this for us, invisibly
  - we are blissfully ignorant of cookies as we browse the web
- In this context, we have to **manage them ourselves**
- Primary concern is session cookies
  - otherwise you can ignore them

```
== Server -> Client ==

Set-Cookie: SID=31d4d96e407aad42; Path=/; Secure; HttpOnly
Set-Cookie: lang=en-US; Path=/; Domain=example.com
Set-Cookie: breadcrumbs=off


== Client -> Server ==

Cookie: SID=31d4d96e407aad42; lang=en-US; breadcrumbs=off
```

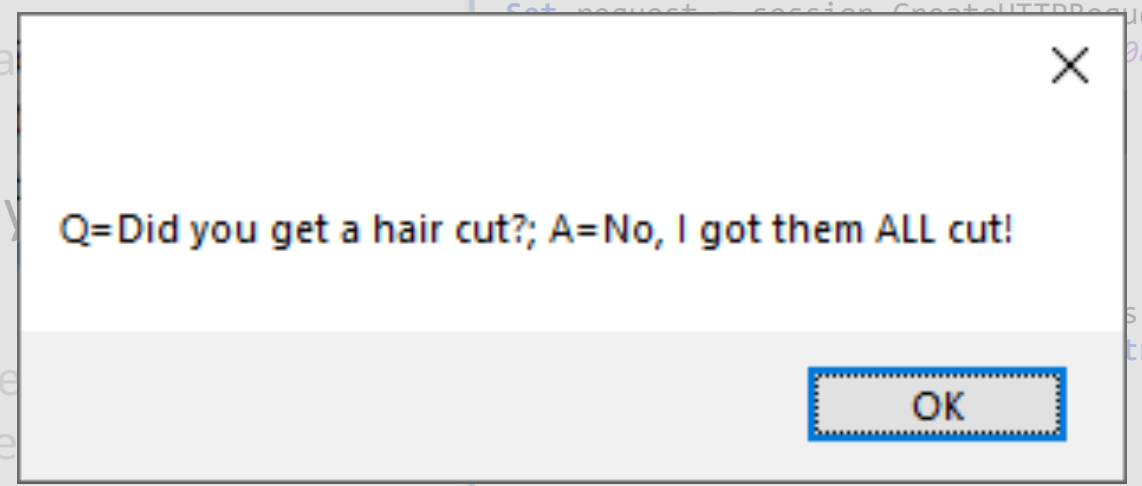
https://tools.ietf.org/html/rfc6265

- Get ALL **Set-Cookie** headers
  - there may be more than one
- Throw away everything after ";"
  - if there is one… not required

```
Dim session As New NotesSession
Dim request As NotesHTTPRequest
Dim url As String

Set request = session.CreateHTTPRequest()
url = "http://dadjokes.demo.local:8080/cookiejoke"
Call request.Get(url)

Dim i As Integer
Dim headers As Variant
Dim cookies As Variant

headers = request.GetResponseHeaders()
ReDim cookies(UBound(headers)) As String

For i = 0 To UBound(headers)
  If (Left(LCase(headers(i)), 10) = "set-cookie") Then
    cookies(i) = StrLeft(Mid(headers(i)+";", 12), ";")
  End If
Next
cookies = FullTrim(cookies)
MessageBox Join(cookies, "; ")
```

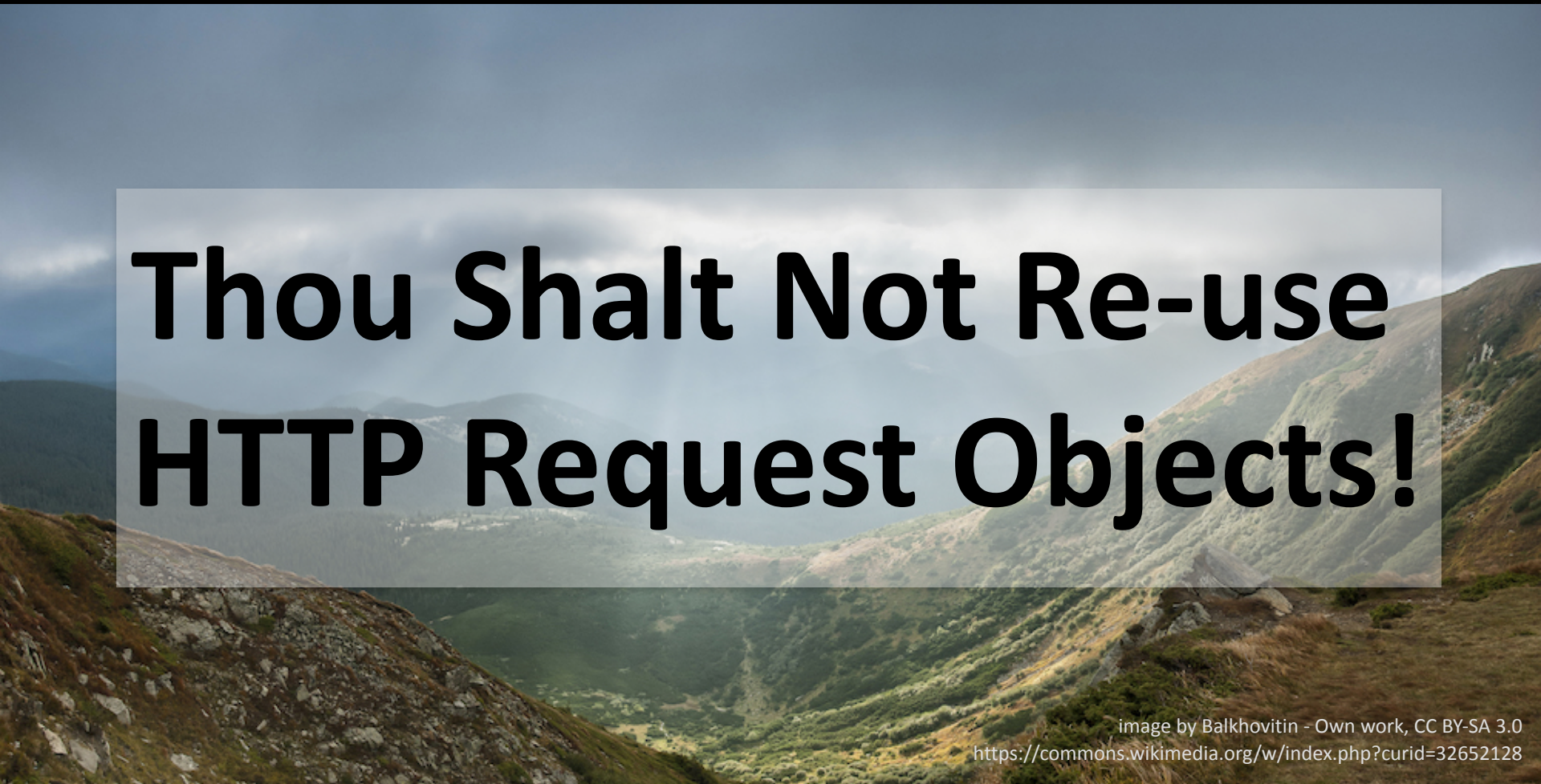
- Get ALL Set-Cookie headers
  - there ma
- Throw away after ";"
  - but there ";" at the

```
Dim session As New NotesSession
Dim request As NotesHTTPRequest
Dim url As String

Set request = session.CreateHTTPRequest()
                                    080/cookiejoke"

                                    s()
                                    tring

                                    "set-cookie") Then
    cookies(i) = StrLeft(Mid(headers(i)+";", 12), ";")
  End If
Next
cookies = FullTrim(cookies)
MessageBox Join(cookies, "; ")
```



- **Concatenate** all the cookies you want to send using "; "
  - there must be **a space after ";"**
  - no ";" if there is only one cookie
  - order doesn't matter
- Use **request.SetHeaderField()**
  - you can also call SetHeaderField("Cookie", "…") **multiple times**
  - it will take care of concatenation for you
- BUT, there's **an important caveat…**

Thou Shalt Not Re-use HTTP Request Objects!



Every Transaction Needs a New Request

- Get the cookie with one request, then **create a new request** to send it back
- **Every** transaction needs a new request

this is super important! →

```
Dim session As New NotesSession
Dim request As NotesHTTPRequest
Dim url As String

Set request = session.CreateHTTPRequest()
url = "http://dadjokes.demo.local:8080/cookiejoke"
Call request.Get(url)

Dim i As Integer
Dim headers As Variant, cookies As Variant
headers = request.GetResponseHeaders()
ReDim cookies(UBound(headers)) As String

For i = 0 To UBound(headers)
  If (Left(LCase(headers(i)), 10) = "set-cookie") Then
    cookies(i) = StrLeft(Mid(headers(i)+";", 12), ";")
  End If
Next
cookies = FullTrim(cookies)

Set request = session.CreateHTTPRequest()
Call request.SetHeaderField("Cookie", Join(cookies, "; "))
MessageBox request.Post(url, "Here's your damn cookie")
```

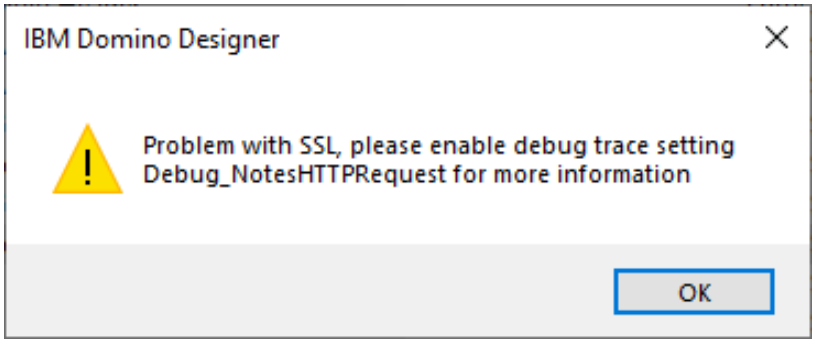
- So, for Session-Based Authentication:
  - **get a session cookie** from a REST endpoint (or whatever)
  - **store** the session cookie
  - **create a new request object**
  - **add** the session cookie to the request
  - send the request

SSL/HTTPS

- If a website uses an SSL certificate from a **well-known CA**, HTTPS requests should just work
  - anything your browser doesn't complain about is generally fine

```
Dim session As New NotesSession
Dim request As NotesHTTPRequest
Dim url As String
Dim response As String

Set request = session.CreateHTTPRequest()
request.PreferStrings = True
url = "https://www.hcltech.com/robots.txt"

response = request.Get(url)
MessageBox response
```

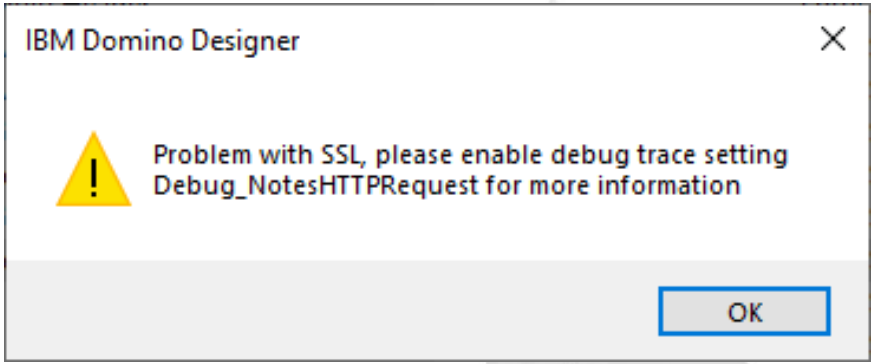
- **Self-signed certificates** can cause problems




```
Dim session As New NotesSession
Dim request As NotesHTTPRequest
Dim url As String
Dim response As String

Set request = session.CreateHTTPRequest()
request.PreferStrings = True
url = "https://dadjokes.demo.local:8081/ssltest"

response = request.Get(url)
MessageBox response
```

- It sure was nice of HCL to tell us how to debug!

- Open your **notes.ini** file, add **Debug_NotesHTTPRequest=1** and try again!



- Debug info will be in the **console.log** file
  (in the **Notes\Data\IBM_TECHNICAL_SUPPORT** folder)

```
[082C:0002-080C] LSXBE HTTP debug information type: Generic Information:
[082C:0002-080C]   Trying 192.168.172.1...
[082C:0002-080C] LSXBE HTTP debug information type: Generic Information:
[082C:0002-080C]  TCP_NODELAY set
[082C:0002-080C] LSXBE HTTP debug information type: Generic Information:
[082C:0002-080C]  Connected to dadjokes.demo.local (192.168.172.1) port 8081 (#0)
[082C:0002-080C] LSXBE HTTP debug information type: Generic Information:
[082C:0002-080C]  ALPN, offering http/1.1
[082C:0002-080C] LSXBE HTTP debug information type: Generic Information:
[082C:0002-080C]  successfully set certificate verify locations:
[082C:0002-080C] LSXBE HTTP debug information type: Generic Information:
[082C:0002-080C]   CAfile: c:\IBM\Notes\Data\cacert.pem
[082C:0002-080C]   CApath: none
[082C:0002-080C] LSXBE HTTP debug information type: Generic Information:
[082C:0002-080C]  TLSv1.2 (OUT), TLS handshake, Client hello (1):
[082C:0002-080C] LSXBE HTTP debug information type: Generic Information:
[082C:0002-080C]  TLSv1.2 (IN), TLS handshake, Server hello (2):
[082C:0002-080C] LSXBE HTTP debug information type: Generic Information:
[082C:0002-080C]  TLSv1.2 (IN), TLS handshake, Certificate (11):
[082C:0002-080C] LSXBE HTTP debug information type: Generic Information:
[082C:0002-080C]  TLSv1.2 (OUT), TLS alert, Server hello (2):
[082C:0002-080C] LSXBE HTTP debug information type: Generic Information:
[082C:0002-080C]  SSL certificate problem: self signed certificate
[082C:0002-080C] LSXBE HTTP debug information type: Generic Information:
[082C:0002-080C]  Closing connection 0
```

```
[082C:0002-080C] LSXBE HTTP debug information type: Generic Information:
[082C:0002-080C]   Trying 192.168.172.1...
[082C:0002-080C] LSXBE HTTP debug information type: Generic Information:
[082C:0002-080C]  TCP_NODELAY set
[082C:0002-080C] LSXBE HTTP debug information type: Generic Information:
[082C:0002-080C]  Connected to dadjokes.demo.local (192.168.172.1) port 8081 (#0)
[082C:0002-080C] LSXBE HTTP debug information type: Generic Information:
[082C:0002-080C]  ALPN, offering http/1.1
[082C:0002-080C] LSXBE HTTP debug information type: Generic Information:
[082C:0002-080C]  successfully set certificate verify locations:
[082C:0002-080C] LSXBE HTTP debug information type: Generic Information:
[082C:0002-080C]   CAfile: c:\IBM\Notes\Data\cacert.pem
[082C:0002-080C]   CApath: none
[082C:0002-080C] LSXBE HTTP debug information type: Generic Information:
[082C:0002-080C]  TLSv1.2 (OUT), TLS handshake, Client hello (1):
[082C:0002-080C] LSXBE HTTP debug information type: Generic Information:
[082C:0002-080C]  TLSv1.2 (IN), TLS handshake, Server hello (2):
[082C:0002-080C] LSXBE HTTP debug information type: Generic Information:
[082C:0002-080C]  TLSv1.2 (IN), TLS handshake, Certificate (11):
[082C:0002-080C] LSXBE HTTP debug information type: Generic Information:
[082C:0002-080C]  TLSv1.2 (OUT), TLS alert, Server hello (2):
[082C:0002-080C] LSXBE HTTP debug information type: Generic Information:
[082C:0002-080C]  SSL certificate problem: self signed certificate
[082C:0002-080C] LSXBE HTTP debug information type: Generic Information:
[082C:0002-080C]  Closing connection 0
```

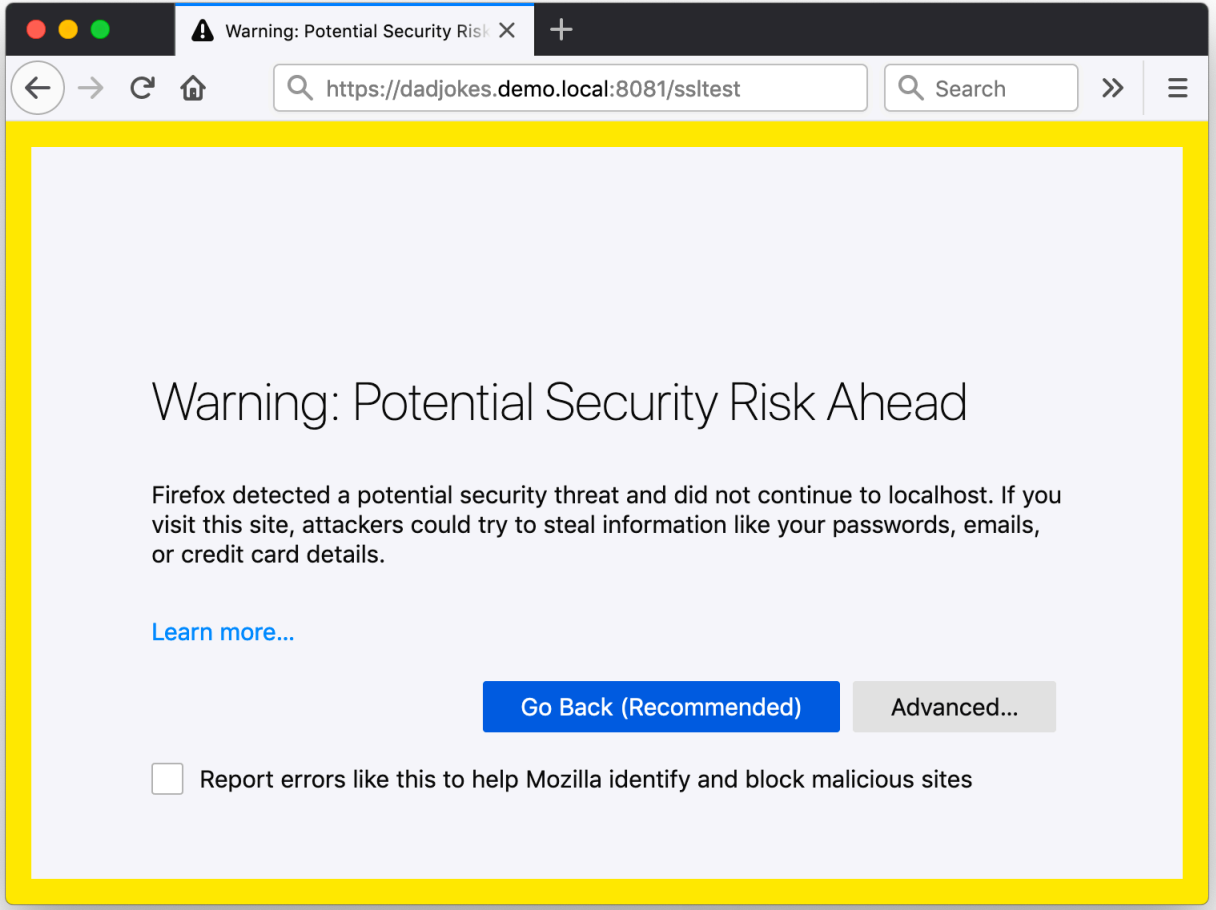
SSL certificate problem:
self signed certificate

```
[082C:0002-080C] LSXBE HTTP debug information type: Generic Information:
[082C:0002-080C]   Trying 192.168.172.1...
[082C:0002-080C] LSXBE HTTP debug information type: Generic Information:
[082C:0002-080C]  TCP_NODELAY set
[082C:0002-080C] LSXBE HTTP debug information type: Generic Information:
[082C:0002-080C]  Connected to dadjokes.demo.local (192.168.172.1) port 8081 (#0)
[082C:0002-080C] LSXBE HTTP debug information type: Generic Information:
[082C:0002-080C]  ALPN, offering http/1.1
[082C:0002-080C] LSXBE HTTP debug information type: Generic Information:
[082C:0002-080C]  successfully set certificate verify locations:
[082C:0002-080C] LSXBE HTTP debug information type: Generic Information:
[082C:0002-080C]   CAfile: c:\IBM\Notes\Data\cacert.pem
[082C:0002-080C]   CApath: none
[082C:0002-080C] LSXBE HTTP debug information type: Generic Information:
[082C:0002-080C]  TLSv1.2 (OUT), TLS handshake, Client hello (1):
[082C:0002-080C] LSXBE HTTP debug information type: Generic Information:
[082C:0002-080C]  TLSv1.2 (IN), TLS handshake, Server hello (2):
[082C:0002-080C] LSXBE HTTP debug information type: Generic Information:
[082C:0002-080C]  TLSv1.2 (IN), TLS handshake, Certificate (11):
[082C:0002-080C] LSXBE HTTP debug information type: Generic Information:
[082C:0002-080C]  TLSv1.2 (OUT), TLS alert, Server hello (2):
[082C:0002-080C] LSXBE HTTP debug information type: Generic Information:
[082C:0002-080C]  SSL certificate problem: self signed certificate
[082C:0002-080C] LSXBE HTTP debug information type: Generic Information:
[082C:0002-080C]  Closing connection 0
```

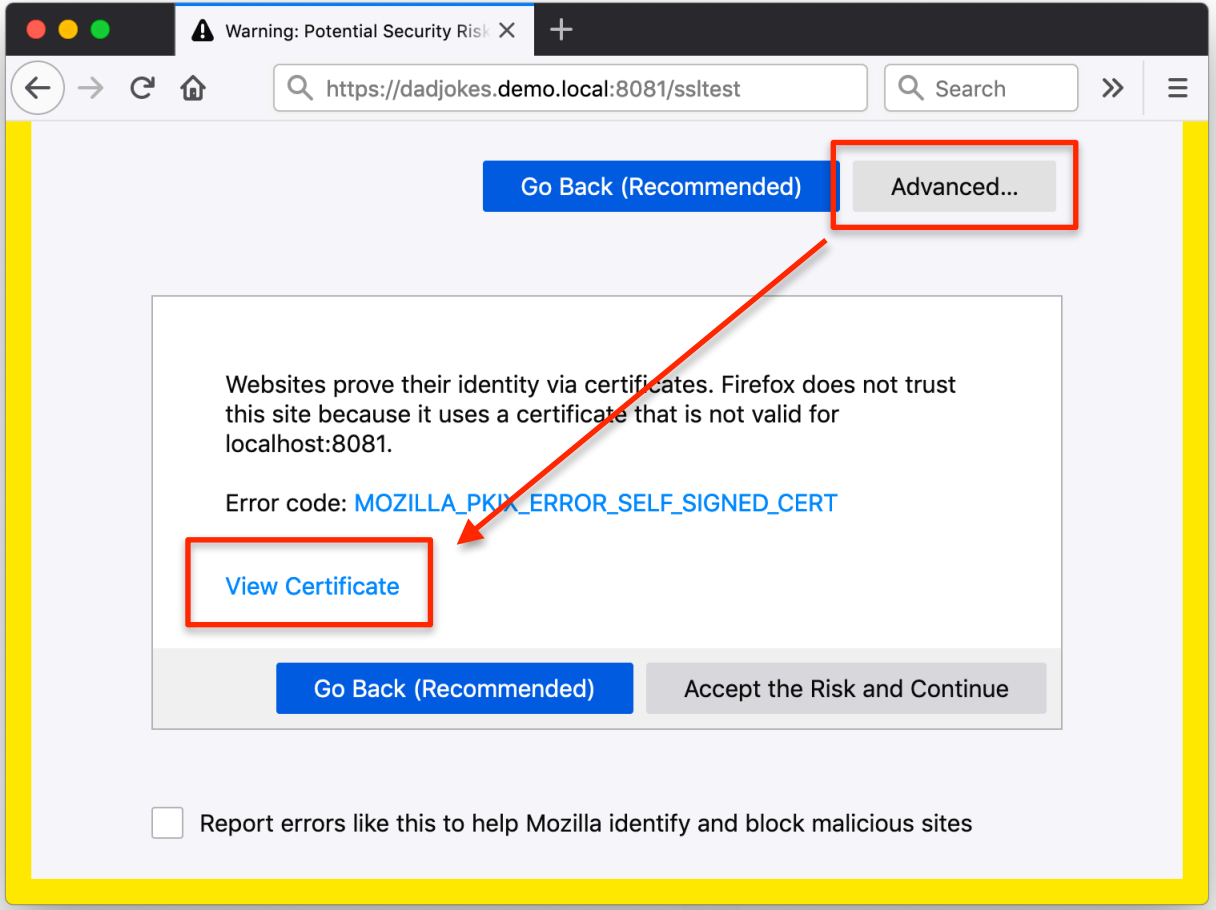
CAfile:
c:\IBM\Notes\Data\cacert.pem

- The **cacert.pem** file is a text file with Base64-encoded certificates recognized by NotesHTTPRequest
  - in your Notes/data directory
- If you have a **self-signed certificate**, you can add it!
- Technote:
  - https://support.hcltechsw.com/csm?id=kb_article&sys_id=c15650e01b333f8883cb86e9cd4bcb9e

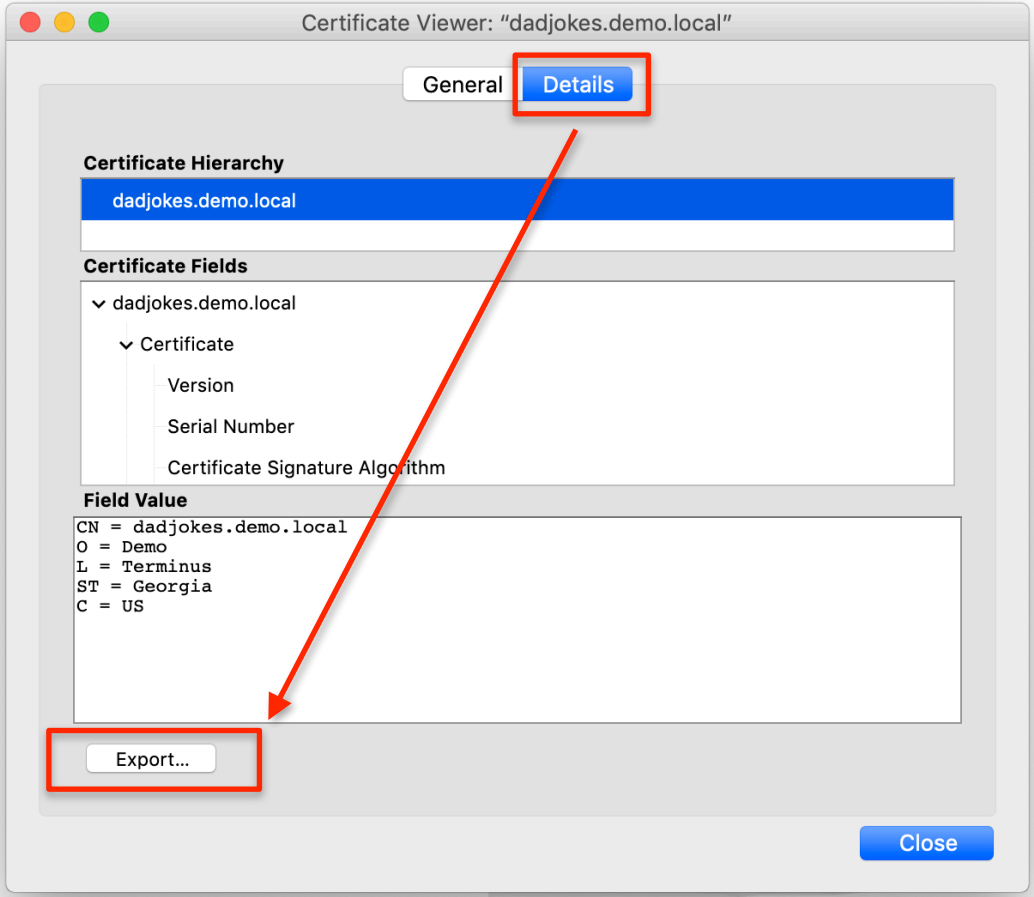
- Open the URL with **Firefox**
  - sorry, Internet Explorer won't help you here…
- Don't forget the **https://** 😉



- Click the "**Advanced**" button
- Then click "**View Certificate**"



- Go to the “**Details**” tab of the certificate viewer
- Click the “**Export**” button



- Export **the entire certificate chain** as **PEM**

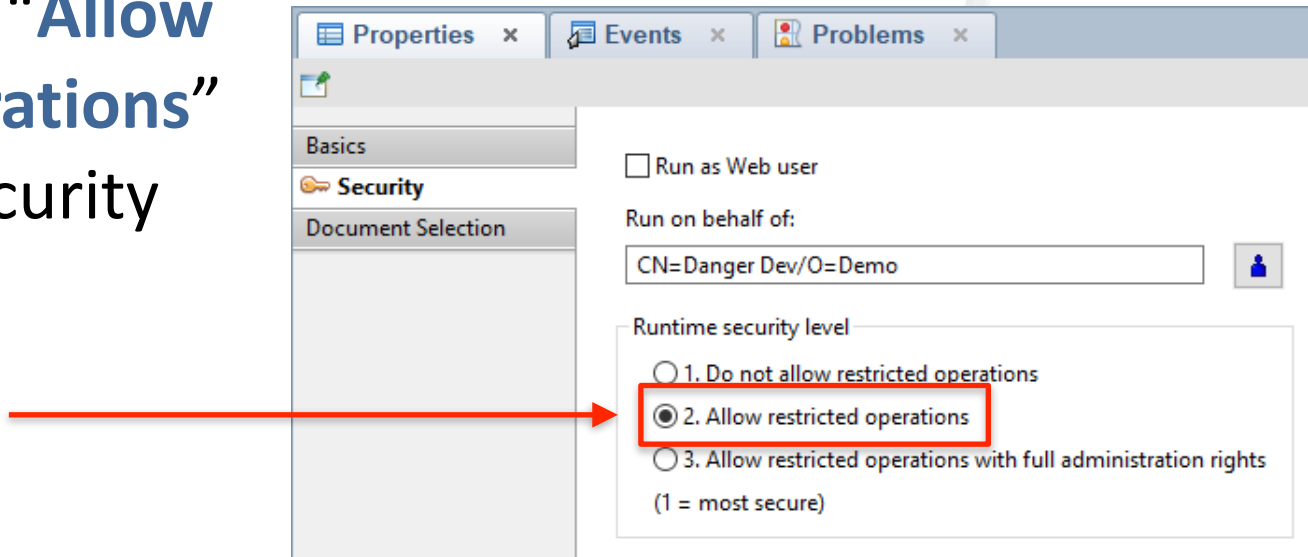
dadjokes.crt.txt

```
-----BEGIN CERTIFICATE-----
MIIFGjCCAwICCQCqsU/kMY/nkjANBgkqhkiG9w0BAQsFADBPMQswCQYDVQQGEwJV
UzEQMA4GA1UECAwHR2VvcmdpYTERMA8GA1UEBwwIVGVybWludXMxDTALBgNVBAoM
BERlbW8xDDAKBgNVBAMMA2QxMDAeFw0xOTEwMjIxOTIzMjdaFw00NjAzMjQxOTIz
MjdaME8xCzAJBgNVBAYTAlVTMRAwDgYDVQQIDAdHZW9yZ2lhMREwDwYDVQQHDAhU
ZXJtaW51czENMAsGA1UECgwERGVtbzEMMAoGA1UEAwwDZDEwMIICIjANBgkqhkiG
9w0BAQEFAAOCAg8AMIICCgKCAgEArui9W4blmuNuFFCOLceYmtDGcsN1BuNPXk5C
Xn6dprh7K5zAl8NOjfica0p5GycKVJ0j+KfnPtaJ2nqIOdriwOd8Yo/dpJJFtJQI
tWUNiTmV3rFpVuwzBqy0ktApwxOl8U52NFgUjtxw1Fs7CtzKKhcwi/2v6wC2zF63
H7X9FIcsq1UYm5HkNOUGvt1vOsW1tx9O6/x8o7K8u+ip9it4fWe4//cEzDDgTV2g
sRr2w7T2vL9zYpL3DPRJTc7pS/9dAw/2ZcaxC6ufnV/z36vlXQWdN2rYnELKZq+V
GR5r4qPw5egs/pA5DV/eJU9vfxGjtkAHgvkQkZhY0s7F+I/EEOwNlSjo7Qc0q2nB
XK2ErWYBw4AM0vCfr7beq2zz6hP5wG0HTigh5K9jXlqso36mQ2KDrDDGSOlLyZIz
3l4gxWVdBeLjvjiqVPDUY2NCWuET/TxG4QK/cscgZNQab6YuicXOdzmyMHDsF8Rf
hX9u3apimoSppm1B/Ww4oUpT1PS6ibFx5i2wKeIpkftqhV1sR5apvtf7RWr4EcY4
e/sU9uZ7WiIv2SBxwsA2uTJMc7rESbpqJQouod+IZ4EpO3qguekkS3+/HyhGh5gc
```

- Copy and paste the entire exported certificate text to the end of the **cacert.pem** file
  - on a Notes client, just retry the request
  - on a Domino server, you might have to restart HTTP
- These instructions are good as of **Notes 10.0.1 FP3**
  - should work on earlier versions too
  - **future** versions of Notes/Domino might have different or easier ways to do this

- Speaking of Domino…

- If you run an agent with NotesHTTPRequest **on a server**, you need to set "**Allow Restricted Operations**" on the agent security settings tab



JSON Data

- To send JSON, you have to **craft your own** JSON string 😒
- You often (but not always) need to set **Content-Type** header field

```
Dim session As New NotesSession
Dim request As NotesHTTPRequest
Dim url As String
Dim json As String
Dim response As String

Set request = session.CreateHTTPRequest()
url = "http://dadjokes.demo.local:8080/sendjoke"
json = |{
    "question": "Would you like your milk in a bag?",
    "answer": "No, you can leave it in the carton." }|

Call request.SetHeaderField("Content-Type", _
    "application/json; charset=utf-8")
response = request.Post(url, json)
MessageBox response
```

- You can use the **NotesJSONNavigator** class to read JSON
- Make sure to set **PreferJSONNavigator**
  - Notes 10.0.1 FP2
- **GetElementByPointer** is easiest way to read

```
Dim session As New NotesSession
Dim request As NotesHTTPRequest
Dim url As String
Dim jsonNav As NotesJSONNavigator

Set request = session.CreateHTTPRequest()
url = "http://dadjokes.demo.local:8080/jsonjoke"

request.PreferJSONNavigator = True
Set jsonNav = request.Get(url)

Dim jokes As String, lf As String
lf = Chr(10)
jokes = _
lf & jsonNav.GetElementByPointer("/jokes/0/question").Value & _
lf & jsonNav.GetElementByPointer("/jokes/0/answer").Value & _
lf & jsonNav.GetElementByPointer("/jokes/1/question").Value & _
lf & jsonNav.GetElementByPointer("/jokes/1/answer").Value

MessageBox jokes
```

How to Read Incoming JSON

```
/jokes                {
                        "jokes":
                        [
/jokes/0                  {
/jokes/0/question           "question": "How do I look?",
/jokes/0/answer             "answer":   "With your eyes!"
                          },
/jokes/1                  {
/jokes/1/question           "question": "Are you listening to me?",
/jokes/1/answer             "answer":   "That's a strange way to start a conversation..."
                          }
                        ]
                      }
```

- GetElementByPointer is easiest way to read

```
lf & jsonNav.GetElementByPointer("/jokes/0/question").Value & _
lf & jsonNav.GetElementByPointer("/jokes/0/answer").Value & _
lf & jsonNav.GetElementByPointer("/jokes/1/question").Value & _
lf & jsonNav.GetElementByPointer("/jokes/1/answer").Value

MessageBox jokes
```

- NotesJSONNavigator, Element, Object, Array
  - https://www.ibm.com/support/knowledgecenter/en/SSVRGU_10.0.1/basic/H_NOTESJSONNAVIGATOR_CLASS.html
  - https://www.ibm.com/support/knowledgecenter/en/SSVRGU_10.0.1/basic/H_NOTESJSONELEMENT_CLASS.html
  - https://www.ibm.com/support/knowledgecenter/en/SSVRGU_10.0.1/basic/H_NOTESJSONOBJECT_CLASS.html
  - https://www.ibm.com/support/knowledgecenter/en/SSVRGU_10.0.1/basic/H_NOTESJSONARRAY_CLASS.html

- They can only **read** JSON into an object
  - as of Notes 10.0.1 FP3
- You can't generate your own from scratch
  - unless you've already got a JSON string
- This will change in the future!
  - Notes 11

- Problems fixed in Notes 10.0.1 FP2
  - error if incoming JSON string is **empty**
  - error if incoming JSON string has **UTF-8** characters
  - error if incoming JSON string is **> 64k**
  - error if incoming JSON string has **linefeeds**

https://support.hcltechsw.com/csm?id=kb_article&sys_id=9fbbeef31b27774883cb86e9cd4bcb40

- Use **GetElementByPointer()** if you can
  - you have to already know the structure of JSON
  - pointer syntax: https://tools.ietf.org/html/rfc6901
- If not, **GetNextElement()** or **GetNthElement()**
  - problems with GetNthElement() in early versions
  - a little tricky to step through the elements

Stepping Through JSON Elements

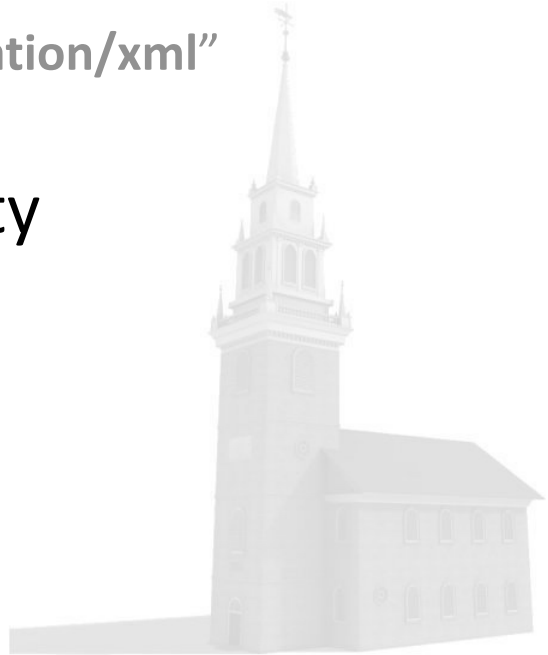

```
Function JsonToString (jsonNav As NotesJSONNavigator) As
String
  Dim json As String
  Dim sep As String
  Dim jsonElem As NotesJSONElement

  Set jsonElem = jsonNav.getFirstElement()
  While Not (jsonElem Is Nothing)
    json = json & sep & ElementToString(jsonElem)
    sep = ", "
    Set jsonElem = jsonNav.GetNextElement()
  Wend

  JsonToString = "{ " & json & " }"
End Function
```

this will convert the contents of a NotesJSONNavigator to a String

```
Function ElementToString (jsonElem As NotesJSONElement) As String
     Dim json As String
     If (jsonElem.Type = JSONElem_Type_Array) Or _
        (jsonElem.Type = JSONElem_Type_Object) Then
           Dim child As NotesJSONElement
           Dim i As Integer, sep As String

           Set child = jsonElem.Value.GetFirstElement()
           While Not child Is Nothing
                 json = json & sep & ElementToString(child)
                 sep = ", "
                 Set child = jsonElem.Value.GetNextElement()
           Wend

           If (jsonElem.Type = JSONElem_Type_Array) Then
                 json = "[ " & json & " ]"
           Else
                 json = "{ " & json & " }"
           End If
     Else
           json = |"| & jsonElem.Value & |"|
     End If

     If (jsonElem.Name <> "") Then
           json = |"| & jsonElem.Name & |": | & json
     End If
     ElementToString = json
End Function
```

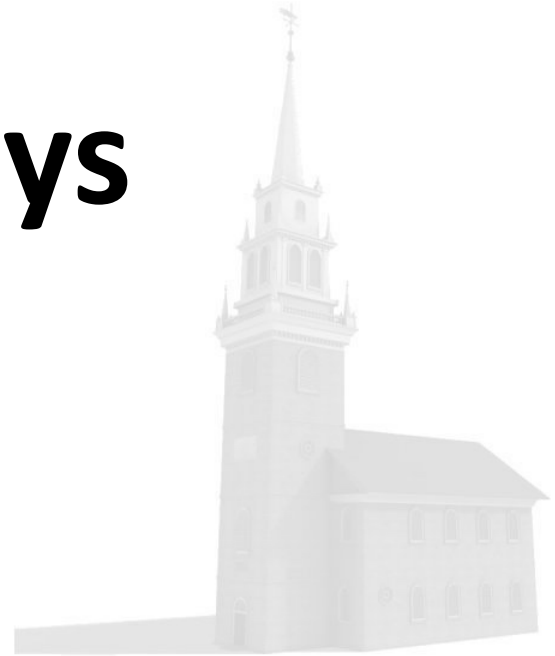
XML Data

- Ultimately HTTP requests with **XML** are just strings
  - we've already seen how to send/receive those
  - Posts might need Content-Type header of "**application/xml**"
- Use **native LotusScript classes** for safety
  - easy to generate broken XML by hand
  - hard to parse XML by hand
  - **NotesDOMParser** is probably what you want



- I wrote an XML wrapper a looooong time ago
- Uses NotesDOMParser under the hood
- Makes it easier to read XML in LotusScript
  - http://nsftools.com/tips/XmlNodeReader

Key Takeaways



- Always use **SSL for Basic Authorization**
- **No linefeeds** in your headers
- **Never re-use** a NotesHTTPRequest object
- Use **Debug_NotesHTTPRequest=1** for troubleshooting
  - or **Wireshark**, if you're hardcore

- Make sure you use **Notes 10.0.1 FP2** or higher

- **Download these slides** and look at them later!

THANK YOU!

Julian Robichaux :: @jrobichaux

panagenda
Senior Software Engineer
www.panagenda.com