XPages CheatSheet v1

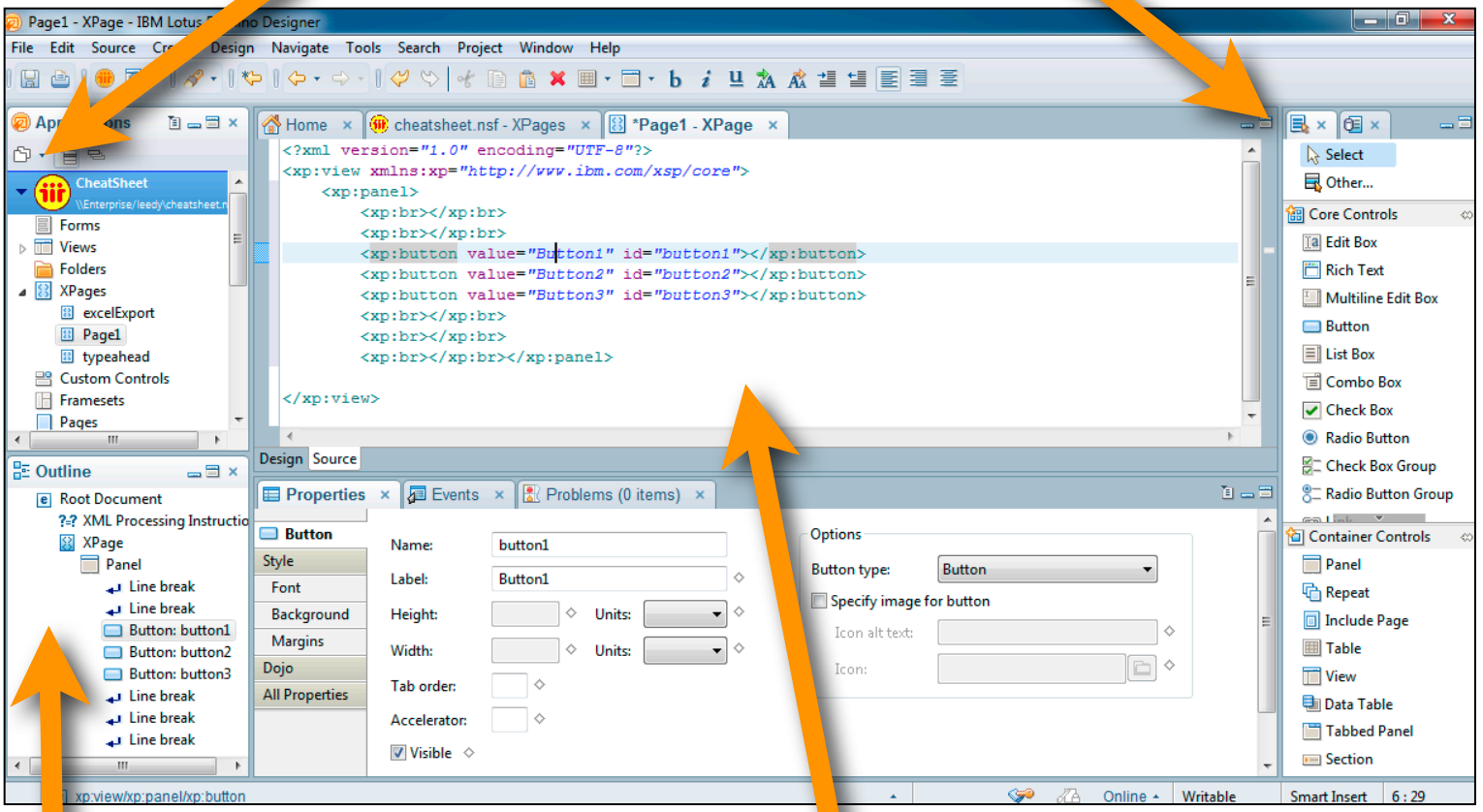
A Lotus Technical Information and Education Resource
Updated as of Jan 29 2011 www.lotus.com/ldd/lotustechinfo
Design Created and Info Compiled by David Leedy - NotesIn9.com ScreenCast

Working Sets

Working sets are very handy to group databases together. You can create a Working set and easily add and remove Notes applications. You can display multiple working sets at a time.

Controls

XPages Provides a variety of basic controls to get you started on your web page. A large number of additional controls are available on OpenNTF.org in the Extension Library project.



OutLine

The Outline is handy to quickly select objects. Selected objects will even be highlighted in the Source Pane. You can even move the objects by clicking and dragging from the Outline.

Note: Some Event properties, like onComplete seem to be ONLY available by clicking on the outline and drilling down from there.

The Source

In addition to using the IDE, XPage developers now have Access to the XML source. This is not the TRUE source as everything gets compiled down to Java Objects, but that's not something you want to modify. The XML Source pane allows you to avoid the IDE, and input code, like HTML for instance, directly. For example, there is no control for HTML's "fieldset". So to create a fieldset, just go into the source and enter the necessary HTML manually.

Note: That even when working in the source you can still make use of the properties settings if your cursor is inside the tags for an XPages control.

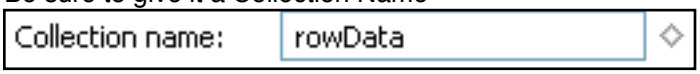
Community Websites		Search PlanetLotus.org for:					
XPages.info	XPagesWiki.com	Jeremy Hodge	Mark Hughes	Declan Lynch	Mike McGarel	Julian Buss	Tommy Valand
XPages.TV	XPagesBlog.com	Paul Withers	Patrick Picard	Stephan Wissel	Paul Hannan	Matt White	Per Henrik Lausten
theXCast.net	XPag.es/?Search	Tim Tripcony	Chris Toohey	Thomas Adrian	Paul Calhoun	Sean Cull	**MANY OTHERS...**

Working with URLs

Redirect to a URL
facesContext.getExternalContext().redirect("http://abc.com")
Redirect to another XPage
context.redirectToPage("Page.xsp")
Current Path
facesContext.getExternalContext().getRequest().getContextPath()
Full Path Including page name
facesContext.getExternalContext().getRequest().getRequestURI()
Full URL
facesContext.getExternalContext().getRequest().getRequestURL()
Get URL Parameter
param.get("parameter")
Accessing /domino/data/html
/.ibmxspres/domino/
/.ibmxspres/domino/icons -> <data>\domino\icons
/.ibmxspres/dojoroot -> <data>\domino\js\dojo-1.x.x
/.ibmxspres/domjs -> <data>\domino\js
/.ibmxspres/domino/oneuiv2/images -> <data>\domino\html\oneuiv2\images
/.ibmxspres/global -> <data>\domino\java\xsp

Repeat Controls

Repeat controls will repeat any list type data - Views, collections, Arrays, Vectors, etc. They do NOT need to look like a table.
Be sure to give it a Collection Name



You can often use Simple Data Binding by setting your data source to "rowData". "rowData" is like a NotesViewEntry. You don't have immediate access to the document but you can use getDocument(). However getColumnsValues() will perform better.
rowData.getColumnValues()[1]
rowData.getColumnValues()[2].toString()
rowData.getDocument().getItemValueString("createdBy")
To get the common name add formula language
@Name("[CN]",rowData.getColumnValue("createdBy"))

Document Looping

SSJS uses the Java, not LotusScript, versions of the Domino Object Model. XPages will recycle Domino Objects when the request to the server completes, but loops need to be handled manually.
Here's an example

```
var doc:NotesDocument = collect.getFirstDocument();
var tmpDoc:NotesDocument
while (doc != null) {
tmpDoc = collect.getNextDocument();
// Insert Code here.  Work with the "doc" object.
doc.recycle();
doc = tmpDoc;
}
```

Scoped Objects

applicationScope - Available to all
sessionScope - Current User
viewScope - Current Page
requestScope - 1 Trip to server

```
// Creating a Scoped Variable
viewScope.put("varName","value")
// Retrieving a Scoped Variable
var temp = viewScope.get("varName")
```

Scoped vars can not hold Domino objects but can hold Java/ JavaScript Objects. A Java Hashmap is similar to an array or list. It's an object of matching key/value pairs.

```
sessionScope.basket = (sessionScope.basket || new java.util.HashMap());
sessionScope.basket.put("key,"value");
```

DBColumns and Lookups

Note that @DbColumn and @DbLookup still have a 64k limit.
Examples:

```
db = new Array(@DbName()[0], 'names.nsf');
@DbColumn(db, "(People)", 2)
```

Pass an Array in to specify a foreign Database

```
var serverName:NotesDatabase = database.getServer();
var dbName = new Array(serverName,"lookupdb.nsf");
var lRep = @DbLookup(dbName,"Viewname",lookupvalue,2);
```

Parameter Possibilities:

```
@DbLookup(dbName, viewName, key, colNumber )
@DbLookup(dbName, viewName, key, fieldName )
@DbLookup(dbName, viewName, key, colNumber, keywords )
@DbLookup(dbName, viewName, key, fieldName, keywords )
```

Note: That if you are using @DbColumn and @DbLookup's to populate a Repeat Control, they will not return an array if only a single value is found. To force an array to be returned do this:

```
var r = @DbLookup(...);
return ((r.constructor == Array) ? r : [r]);
```

Type Ahead Example

Pull Values from another database in the SAME directory

```
key = getComponent("jobSearch").getValue();
var path = database.getFilePath().split(database.getFileName())[0]
var dbInfo = new Array(@DbName()[0],path + "dbname.nsf");
return @DbLookup(dbInfo,"(lookupView)",key,1,"[PARTIALMATCH]");
```

Visible Property

Visible means "Rendered". If it's not "Visible" then it's not rendered to the browser and doesn't exist on the web page. To hide and show sections use nested panels, "outer" and "inner" . Use viewScope variable to hold true or false and read that value in to the visible/ rendered property. Then do a Partial Refresh on the OUTER panel which will then cause the server to resend the Inner Panel to the browser.
Visible/Rendered Property:

```
viewScope.get("showDetail")
```

Show Button (onClick):

```
viewScope.put("showDetail", true)
```

Hide Button (onClick)

```
viewScope.put("showDetail", false)
```

Programming Tips

JavaScript is CASE SENSITIVE
= sets values, == compares values
No dot notation when dealing with fields. Use
document.getItemValueString("fieldname")
document.replaceItemValue("fieldname")
@Functions use commas, not semi-colons
@Functions might have different parameters
@Functions without parameters still need parentheses, i.e @Now()

Themes

When creating your own theme it's important to always extend one of the 2 themes that ship with Domino; webStandard and oneUI. This is needed because some of the dojo components, like the date picker rely on various css files to work.
In the example below, the theme extends webStandard in order to load the needed dojo files but not the extra code from the oneUI framework. The screen.css and application.css files will always get loaded but the ie.css file will only be sent to the browser if the user is running Internet Explorer version 6.

```
<theme extends="webStandard">
   <resource>
     <content-type>text/css</content-type>
     <href>/screen.css</href>
   </resource>
   <resource rendered="#{javascript:context.getUserAgent().isIE(0,6) == true}">
     <content-type>text/css</content-type>
     <href>/ie.css</href>
   </resource>
   <resource>
     <content-type>text/css</content-type>
     <href>/application.css</href>
   </resource>
</theme>
```

It should be noted that while DDE will try and render css at design time, that's only if you load the css directly on the page. That does not work if you are solely using themes.
To extend oneUI you would use:
<theme extends="oneUI">

More Theme Tricks

Specify If A Resource Is Only Used When Dojo Controls Are detected.
`<resource dojoTheme="true">`
Browser / Client Detection
<resource rendered="#{javascript:context.getUserAgent().isIE(0,6) == true}">
<resource rendered="#{javascript:context.getUserAgent().isFirefox()}">
<resource rendered="#{javascript:context.getUserAgent().isSafari()}">
<resource rendered="#{javascript:context.isRunningContext('Notes')}">
Text Direction Detection
<resource rendered="#{javascript:context.isDirectionRTL()}">
<resource rendered="#{javascript:context.isDirectionLTR()}">
Not Just For CSS
<content-type>text/javascript</content-type>
<href>clientFunctions.js</href>
Reference Server Based / External Files
<href>/.ibmxspres/global/theme/oneui/iehacks.css</href>
<href>http://www.someserver.com/resources/application.css</href>
Programmatically Changing a Theme

```
var f = "/"+@RightBack(context.getUrl().getAddress(),"/");
context.setSessionProperty("xsp.theme","Company");
context.redirectToPage(f);
```

Export to Excel

Create an XPage with this code and save it. This is a NON-Rendered XPage. So the user will NEVER see this particular page. When it's invoked it will trigger the export of the view data to Excel.

```
<?xml version="1.0" encoding="UTF-8"?>
<xp:view xmlns:xp="http://www.ibm.com/xsp/core" rendered="false">
<xp:this.afterRenderResponse><![CDATA[#{javascript:
var exCon = facesContext.getExternalContext();
var writer = facesContext.getResponseWriter();
var response = exCon.getResponse();
var myView:NotesView = database.getView('VIEWNAME');
var viewNav:NotesViewNavigator = myView.createViewNav();
var viewEnt:NotesViewEntry = viewNav.getFirst();
print("2")
response.setContentType("application/vnd.ms-excel");
response.setHeader("Cache-Control", "no-cache");
writer.write("<table>");
writer.write("<thead><tr>");
writer.write("<td><b>Column1Header</b></td>");
writer.write("<td><b>Column2Header</b></td>");
writer.write("<td><b>Column3Header</b></td>");
writer.write("</tr></thead>");

while (viewEnt != null) {
  writer.write("<tr>");
  writer.write("<td>" + viewEnt.getColumnValues()[0] + "</td>");
  writer.write("<td>" + viewEnt.getColumnValues()[1] + "</td>");
  writer.write("<td>" + viewEnt.getColumnValues()[2] + "</td>");
  writer.write("</tr>");
  viewEnt = viewNav.getNext(viewEnt);
}
writer.write("</table>");
writer.endDocument()}]]></xp:this.afterRenderResponse>
</xp:view>
```

Be sure to update the viewName with the actual name of your view.
Then on an XPage that the user would see, simply add a button to call the non-rendered XPage.
More information can be found here:
http://support.microsoft.com/kb/260239

Data Binding

Bind to an External Database
@Name("[CN]",@Subset(@DbName(),1))+"!!dbname.nsf"
Get the PATH of the current database
database.getFilePath().split(database.getFileName())[0]
or
@LeftBack(database.getFilePath(), "\\")

oneUI

The basic layout needed for working with oneUI.

```
<xp:panel styleClass="lotusFrame">
  <xp:panel styleClass="lotusBanner">Banner Info</xp:panel>
  <xp:panel styleClass="lotusTitleBar">Title Bar Info</xp:panel>
  <xp:panel styleClass="lotusPlaceBar">Place Bar Info</xp:panel>
  <xp:panel styleClass="lotusMain">
    <xp:panel styleClass="lotusLeftCol">Left Side Bar</xp:panel>
    <xp:panel styleClass="lotusRightCol">Right Side Bar</xp:panel>
    <xp:panel styleClass="lotusContent">Main Page Content</xp:panel>
  </xp:panel>
  <xp:panel styleClass="lotusFooter">Footer Information</xp:panel>
  <xp:panel styleClass="lotusLegal">Legal Information</xp:panel>
</xp:panel>
```

Data Relationships

Keep in mind that since you can now do a lookup from inside a repeat control, you can effectively create SQL type "JOINS". This allows you to structure your data a little more like a relational database.
See http://XPag.es/?JOINS

About the CheatSheet

Most of this information has been compiled from **XagesWiki.com** and other Lotus Community Resources like the bloggers listed on the cover. Unfortunately there wasn't space to list everybody.

Working with Domino Objects

Note: session, database, and any bound data sources are global objects and are always available.

var doc:NotesDocument - datasource.getDocument()

Working with a Field:

```
var doc:NotesDocument = datasourcename.getDocument();
doc.replaceItemValue("somefield", "somevalue");
```

More Examples

Create NotesDocument Object

```
var newDoc:NotesDocument = database.createDocument()
```

Assign date to NotesDocument Object

```
var newDate:NotesDateTime = session.createDateTime(“Today”);
newDate.setNow()
newDoc.replaceItemValue(“dateField”, newDate);
```

Save a document

```
newDoc.save()
```

Get the String value of a field

```
var fname = document1.getItemValueString(“firstname”)
```

Get a doc by UNID

```
var doc:NotesDocument = database.getDocumentByUNID(unid)
```

Note: that when creating an in memory object, the “:ObjectName” is not be required, but by including it, the Domino Designer SSJS Editor will be able to recognize the type of object and will provide you typeahead for the methods and properties. And really, even though it’s optional, REAL programmers TYPE their variables. :-)

Dojo Example: AccordionContainer

XPages contain the full Dojo JavaScript Framework. Even if a particular feature of Dojo has not been exposed in Domino Designer, you can still easily use it in your applications.

First be sure to set dojoParseOnLoad and dojoTheme to “true” and add any resources.

```
<xp:view xmlns:xp="http://www.ibm.com/xsp/core" dojoParseOnLoad="true"
dojoTheme="true">
<xp:this.resources>
  <xp:dojoModule name="dijit.layout.AccordionContainer">
  </xp:dojoModule>
  <xp:dojoModule name="dijit.layout.ContentPane"></xp:dojoModule>
</xp:this.resources>
```

Setup your Div

```
<div dojoType="dijit.layout.AccordionContainer" style="height: 300px;">
   <div dojoType="dijit.layout.ContentPane" title="Projects">
     //content for panel 1
   </div>
   <div dojoType="dijit.layout.ContentPane" title="Tasks">
     //content for panel 2
   </div>
   <div dojoType="dijit.layout.ContentPane" title="Other">
     //Content for panel 3
   </div>
</div>
```

XPages in the Media

Be sure to check out “The XCast” podcast. Hosted by Tim Clark, this is a discussion about all things XPages. http://thexcast.net

Working with Controls on an XPage

SSJS

```
getComponent("elementName").getValue();
getComponent("elementName").setValue("something");
```

Client Side JavaScript

```
dojo.byId("#{id:elementID}").value;
```

Mixing @Formulas and SSJS

```
var month = @Month(@Today())
if (month == 8) {
  //Sets the value by binding to the viewScope Value
  viewScope.put("value","August" );
  //Sets the value by updating the control on the XPage
  getComponent("edtResult").setValue("It Really is August");
}
else {
  viewScope.put("value", "Not August");
  getComponent("edtResult").setValue("It's still Not August");
}
```

Many but not all @Formulas have been re-written in SSJS and there is no performance penalty to using them. However, some behave differently then what you might expect. For example in classic Notes code @Unique returns a value based on username and time. In SSJS @Unique returns a pure random number. If you want to get the classic @Unique results into SSJS, use:

```
session.evaluate(“@Unique”)
```

Multi-Value Fields

There are at least 2 ways to display the contents of a multi-value field. One way is to use a Repeat Control. Repeat controls can iterate over any list based data like a collection, array or multi-value field.

Another method is this SSJS snippet:

```
var values = viewEntry.getColumnValues(“ColumnName”);
values.join('<br>')
```

Custom Controls

Custom Controls are much more powerful then traditional sub-forms. One common design pattern for XPages is to do very little programming on the XPage itself, but create Custom Controls for your content. So a web page would be made up of several different custom controls. These controls would then be simply dropped onto an actual XPage. In effect the XPage is then used as a container to hold the custom controls.

You can also define parameters for a custom control in the Property Definition tab. Then when you add the custom control to an XPage or another custom control, you can define the parameter and access the value by using:

```
compositeData.paramaterName
```

Further Information

There are many great articles and sample code on the Lotus Notes and Domino Application Wiki: http://www-10.lotus.com/ldd/ddwiki.nsf Not to be missed are the articles by IBMer Robert Perron. http://www-10.lotus.com/ldd/ddwiki.nsf/xpViewRecent.xsp?searchValue=robert%20perron There are many articles with examples of using Domino Objects inside Server Side JavaScript. And of course check out openNTF.org for sample applications.

For suggestions and comments on this document please see http://xpagescheatsheet.com