

Automatically changing colours in XPage view objects

Author: Graham Dodge


Date: September 2010

Credit: Thanks to Declan Lynch and Keith Strickland for their code examples.

Foreign Language Help for all you Americanised Googlers: color ... color ... color

The Objective here is to colour a cell in a View based on a picklist in a form. I want each task to have a colour which can be individually assigned. Of course, I might also desire the colours to represent tasks assigned to particular individuals (eg all of Tom's tasks are colourcoded 'YellowOnBlack') but I'll leave that exercise as an Extra Credit option if you finish this exercise early.

Here is the desired result:



Name	Assigned To:	Participant	Text
Get Pizza	Tom		BlackOnYellow
Invite the gang	Harriet	Adam Apple,Billy Banana,Carol Carrot,David Darian	YellowOnBlack
Organise transport	Dick		BlackOnLightBlue
Rent Movie	Tom,Harriet	Carol Carrot,Edward Eggplant	PurpleOnWhite

STEP 1: Create a CSS with a list of colours. I used W3C web standard colour names but in a production application I would be tempted to define the colours using #RRGGBB codes:

ProjectColours.CSS

```
.LightRedOnWhite{
    color: LightRed;
    background-color: White;
}
.DarkRedOnWhite{
    color: DarkRed;
    background-color: White;
}
.LightBlueOnWhite{
    color: LightBlue;
    background-color: White;
}
.DarkBlueOnWhite{
    color: DarkBlue;
    background-color: White;
}
etc
etc
```

STEP 2: Put the CSS colour names into a multivalue keyword document e.g.

Edit Document	
KEYWORD	
Name:	Colour
Values:	LightRedOnWhite DarkRedOnWhite LightBlueOnWhite DarkBlueOnWhite LightGreenOnWhite DarkGreenOnWhite YellowOnWhite PurpleOnWhite AquaOnWhite BlackOnWhite

STEP 3: Create a Custom Control with a combobox using Labels and Values populated via a dblookup eg
'@DbLookup (@DbName (), "Keyword" , "Colour" , 2)

STEP 4: Name the project document and the colour combobox on the Custom Control for future reference. I set the document name = 'ProjectDoc' and the combobox field = 'ProjectColourText' .

STEP 5: Select the Style tab for the Custom Control and add the 'ProjectColours' style sheet. Set the Style Class of the combobox to the Expression Language value 'ProjectDoc.ProjectColourText' . This will make the combobox display using the colour values defined in the CSS

The screenshot displays a software interface with a design view and a script editor. In the design view, a form titled "EDIT PROJECT" contains a "Project name:" field and a "Project colour:" combobox. The combobox is highlighted with a red box and labeled "ProjectColourText". Below the form is a table with columns "AssignedTo", "Participant", and "Comment".

The "Properties" window shows the "Style" tab selected. The "Class" is set to "{Computed}" and the "Styles" list includes "ProjectColours.css". The "Script Editor" window shows the "Language" set to "Expression Language (EL)" and the "Condition" set to "ProjectDoc.ProjectColourText".

Red lines connect the "ProjectColourText" label in the design view to the "Style" tab in the Properties window and the "ProjectDoc.ProjectColourText" condition in the Script Editor.

STEP 6: You can now start creating Project documents and assigning them colours. When you assign a colour the display of the Project Colour field will change to match your selection:

Project name:

Project colour:

STEP 7: We now need to have the selected colours displayed on the top level view of the documents. Set the Data Var for the View object to 'SelectedProjDoc'. (Note that this is the 'Data Var' setting ... **not** the 'Data\Data Var' setting).

The screenshot shows a view design grid with columns: Name, AssignedTo, Participant, and Text. Below the grid is the Properties window. The 'View' tab is selected, and the 'data' section is expanded. The 'var' field is set to 'SelectedProjDoc'.

Property	Value
refreshId	
rendered	
rendererType	
summary	
title	
data	
+ data	xp:dominoView
first	
indexVar	
pageName	
rows	
value	
var	SelectedProjDoc

STEP 8: The final step is to tell the selected columns within the View object that they should use the value retrieved from the ProjectColourText field within each document as the CSS colourvalues for that particular document.

The screenshot shows a view design grid with columns: Name, AssignedTo, Participant, and Text. Below the grid is the Properties window. The 'Style' property is selected. The Script Editor is open, showing the 'Compute Dynamically' option selected and the expression 'SelectedProjDoc.ProjectColourText' entered.

Script Editor Configuration:

- Language: Expression Language (EL)
- Condition: Compute Dynamically Compute on Page Load
- Expression: SelectedProjDoc.ProjectColourText