LaunchPad Tiles

Overview

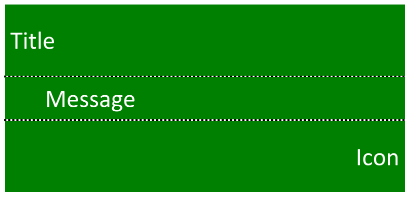
A tile view is often in a LaunchPad dashboard. This document will explain the basic use, layout, and customization capabilities including the use of expressions.

Basic Usage

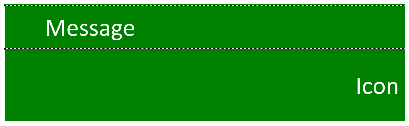
- Run a standard IDI which shows one or more numeric columns.
- From the grid toolbar, choose View As → Tile.
- A tile is shown which includes the values from the grand totals of all numeric columns.
- Save the view, if required, in the normal way.

Tile Layout

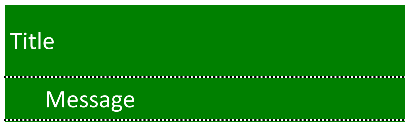
A tile is made up of three areas stacked vertically - the Title, Message and Icon.



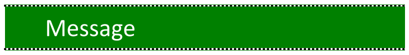
- when the title is undefined then the Title area is suppressed.



- when the icon is undefined then the icon area is suppressed.



- when both are undefined then both are suppressed.



This then allows the Message to define more/all of the tile, which allows for greater customization.

Customization

When a tile is presented choose Tools/Tile Settings.

Title

The title to show at the top of the tile.

- Setting the title to blank suppresses the Title area of the tile.

Tile Color

The default background color of the tile.

- The background color of text in the Message part can be different colors.

Text Color

The default font forecolor to use for all text in the tile.

- Text in the Message part can be different colors.

Icon

The icon to show in the bottom right of the tile.

  - Setting the Icon to None suppresses the Icon area of the tile.

Message

The text to appear in the Message area of the tile.

- The system generates a message based on the currently presented numeric columns in the view. For example it might generate:

```
{Total Balance}
Total Balance
```

- Text within single curly braces, e.g. {Total Balance}, refer to the grand total of column with that title.
- It is also possible to refer to the underlying field names if known e.g. {Total Balance} may refer to the underlying field name of BALANCE and hence could also be referred to as ${BALANCE} (internally single curly brace tokens are simply replaced with ${<fieldname>} prior to onward processing).
- When the message is generated it is indented. Highlight the text and click the Decrease Indent button on the message toolbar to remove this indent.
- Expressions may be used in the message – refer Expressions.
- To show the total number of rows in the grid ie the rowcount use ${TOTALROWS}

Expressions

Expressions are a string which define a computation. Expressions are evaluated purely in the browser - i.e. they are not evaluated at the server.

An expression is defined by enclosing it within double braces.

```
{{ <expression text> }}
```

Column values are referred to within the expression text in the normal way i.e. within single curly braces.

An expression can perform arithmetic e.g. to add 10% to the Total Balance.

```
{{ {Total Balance} * 1.1 }}
```

The javascript Math object is exposed using MATH e.g. to get the minimum of two values:

```
{{ MATH.min( {Value1}, {Value2} ) }}
```

Expressions can have an arbitrary number of operations e.g.

```
{{ {Total Balance} * 1.1 / MATH.min( {Value1}, {Value2} ) + 10000 }}
```

Built In Functions

A number of built in functions exist:

Function	Description	Usage
ICON	Show specified icon	ICON("/custom/images/flag.png")
GREATERICON	Show an icon if a value is over a specified amount	GREATERICON({value1},{value2},"/custom/images/flag.png")
FORMAT	Format a numeric with comma grouping	FORMAT({value1}) *note remember to enclose function in {{ FORMAT({value1}) }} Optionally the decimal places can be specified as well ie FORMAT({value1},3) displays 3 decimal places, with the second argument omitted no decimal places is assumed
MATH.<function>	Any valid javascript Math object <function>	e.g. MATH.min({Value1}, {Value2})

Custom Functions

Custom functions can be added to the system. These functions are simply javascript functions.

To do this create or edit the file /web/custom/local.js and add a javascript function to the SW.userFunctions object.

For example the following defines a function named GR which shows a value in a specified colour when it exceeds a limit:

```
SW.userFunctions["GR"] = function( value, limit, color ) {
 if ( value > limit )
   return "<font color='" + color + "'>" + value + "</font>";

 return value;
}
```

After adding a function the page must be reloaded to make the function available.