Styling of HTML Reports

Overview

opReports versions 3.0.8 and newer provide various mechanisms that let you adjust the styling of HTML reports. These mechanisms are primarily aimed at a report's "standalone" display, i.e. when the report is saved as ZIP file or emailed to somebody, and do not apply when the report is viewed in "embedded mode" within the application.

Custom Banner

To add a custom banner before the report content, you can use the report option `homelink`: if set to a string of HTML that html will be inserted before the report.

Please note that this option is available only for scheduled reports, and affects the report only if viewed in standalone fashion.

Custom CSS and JavaScript

opReports automatically includes or attaches a CSS style sheet and a Javascript file with any HTML report.

The following configuration options define which files get attached:

```
'opreports_default_css' => '<omk_public>/omk/css/opReports_report.css',
'opreports_default_js' => '<omk_public>/omk/js/opReports_report.js',
```

If you want to make changes it's recommended that you copy these files, edit the copies and update the configuration; the installer will overwrite the default files on upgrade, and any changes to those files would therefore be lost.

Custom Templates

Since version 3.0.8 opReports' HTML reports are templated which simplifies customisation by end-users.

Each opReports HTML report involves two templates:

1. `templates/reports/reports/report_wrapper.html.ep` is shared by all reports and produces the outer, common, structure.
   The wrapper template is only *partially* in control when a report is show in embedded mode within the application (i.e. the HTML headers and everything before and after a set of markers is ignored). The wrapper template is fully interpreted if the report is viewed in "standalone mode" (i.e. viewing an emailed report, or a ZIPped report or when you click on a report's HTML link).
2. `templates/reports/reports/report_<type>.html.ep` is specific to the report type in question, and produces the report body.

Please refer to the provided standard report templates for details regarding the data interface.

Limitations and Caveats

- This functionality is meant for advanced users only!
  You need to learn a little bit about Mojo Templates and you'll likely have to experiment a little before achieving full success.
- The data interface between opReports and the templates is subject to change between versions.
- Template modifications do cause installer warnings on upgrades, and by default the installer will **overwrite all templates**!
  It is vital that you keep backups of your custom templates and manually merge or restore your desired customisations after an opReport upgrade.
- The `report_wrapper` template contains two crucial markers that control its dual functionality:
  Everything before '`<!-- report body start 7fd66b21f83244e6ca07cf27dbac3b0c -->`' and the corresponding 'body end' marker is omitted when the report is shown in embedded mode.
  It is therefore crucial that you put your changes in the appropriate place, i.e. custom headings or logos should go on the *in*side of this marker set.

Custom Logos, other Custom Attachments

In opReports 3.0.10 we've added the ability to attach and associate arbitrary files with a report instance. This can be used together with a custom template to show your logo in standalone mode.
opReports will include all custom files when a report is emailed or saved as ZIP file.

To make use of this facility you have to list all custom files you want to include in configuration option `opreports_custom_files`, e.g.

```
'opreports_custom_files' =>  [ '/path/to/my/logo.png', '/some/other/file.jpg' ],
```

Please note that the files must be present on the server, readable by the webserver's user (usually `apache` or `www-data`), and the given path must be absolute.

When opReports creates a report in HTML format, all custom files defined at the time of report creation are copied and attached to the report. Every HTML report will have copies of all defined custom files attached, and it's inadvisable to use very large files in this situation.

All custom files are copied and saved separately with every report instance, and the per-report copies are saved with a dynamically generated globally unique file name.

Access to Custom Files for Visualisation

Custom files **cannot** be referenced by their original file names, as only the per-report copy is included with the report, and that has a dynamically generated file name. This means that referencing such files from a CSS stylesheet is likely not possible. However, from a report template this information is easy to access, and may very well be saved in the HTML as a Javascript map object or HTML attribute.

The report's two templates can refer to the report's copies of any custom files by accessing the data interface variable `$report->{meta}->{moduleurls}`, which is a hash datastructure (key: the full "original" path given in `opreports_custom_files`, value: the unique file name for that file and this report, which works fine as a safe relative url for it).

The standard wrapper template contains this usage example, which produces an HTML comment for every custom file involved:

```
% for my $extra (grep(!/^(css|js)$/, keys %{$report->{meta}->{moduleurls}})) {
<!-- attached file: <%= $extra %> as <%= $report->{meta}->{moduleurls}->{$extra} %> -->
% }
```