

Nozzle: Firehose Report Library

Nils Gehlenborg with Lihua Zou, Douglas Voet and Michael Noble

GDAC Team Meeting / 15 February 2011

What are the goals for this meeting?

1. Introduction and overview of Nozzle.
2. Collection of feedback to address potential issues before first release.

What is “Nozzle”?

- "A device attached to the end of a fire hose that **directs, shapes** and **regulates** the flow of the water or fire fighting agent pumped into the hose. (Wikipedia)



- Our R Package to generate Firehose reports based on HTML, CSS (cascading style sheets), JavaScript and semantic markup.

Why are we doing this?

- **Content is difficult to interpret.**
 - Because important explanations are missing.
- **Organization is difficult to navigate.**
 - Because every report has a different structure.
 - Because every report has a different layout.
 - Because some reports contain a lot of detail.

What are the goals for Nozzle?

- All reports should have the same **structure**.
- All reports should have the same **layout**.
- All reports should have **advanced features** such as folding sections and subsections, zoomable figures, etc.
- All reports should be created with a **simple** set of instructions and without any knowledge of the technologies used to render the reports.
- **Developers focus on the content!**

Example: Text Formatting

without Nozzle (from an existing report):

```
write(paste("<p><b>Figure 1:</b> Consensus NMF clustering of ", genenumber, " variably expressed genes and ",  
samplenum, " samples.</p>", sep=""), file="report.html", append=TRUE);
```

with Nozzle:

```
p <- newParagraph( asStrong( "Figure 1: "), "Consensus NMF clustering of ", genenumber, " variably expressed genes  
and ", samplenum, " samples." );  
  
# insert p into a section, ..., write report
```

Example: Figure

without Nozzle (from an existing report):

```
write( paste( "<br>", sep=" "),
       file="report.html", append=TRUE);
```

with Nozzle:

```
imageFile <- "files/histogram.png";
imageFileHighRes <- "files/histogram.pdf"; # could also be a high-res pixel-based image file

f <- newFigure( imageFile, imageFileHighRes, "Histogram of Correlation Values. ..." );

# insert figure into a section, ..., write report
```

Example: Table

without Nozzle (from an existing report):

```
top = "<h3> Top 25 Most Negatively Correlated</h3><table><tr>";

title = dimnames(ov_final)[[2]];
for (t1 in 1:length(title)){
  top = paste(top, "<th>", title[t1], "</th>", sep="");
}
top = paste(top, "</tr>", sep="");

### TOP 25 ###
for (a in 1:25){
  top = paste(top, "<tr>", sep="");
  for (b in 1:ncol(ov_final)){
    top = paste(top, "<td>", ov_final[a,b], "</td>", sep="");
  }
  top = paste(top, "</tr>", sep="");
}
top = paste(top, "</table><br>", sep="");
write(top, file="report.html", append=TRUE);
```

with Nozzle:

```
corFile <- "files/correlations_gdac_spearman.txt";
corTable <- read.table( corFile, sep="\t", nrow=25, stringsAsFactors=FALSE, header=TRUE );

t <- newTable( corTable, corFile, "Top 25 most negatively correlated methylation probe/gene pairs. ..." );

# insert table into a section, ..., write report
```


Example: Table (cont'd)

```
newTable( table, tableFilename, ..., protection=PROTECTION.PUBLIC )
```

This function creates a Table element.

Arguments:

- **table** - A matrix or data.frame R data structure containing the table that should be included in the report. The table header will be created from the column names of the table data structure. The number of lines should be limited to no more than 30 unless there are good reasons to include more.
- **tableFilename** - Path and filename to a file containing the full table. Ideally, this should be a tab-delimited text file with a single line header.
- ... - Zero or more strings and scalar variables that will be concatenated and used as caption for the table. This may also contain references and Result elements.
- *protection* - Protection level for the new element.

Value:

A Table element.

Example: References

```
simpleCitation <- newCitation( authors="Nils Gehlenborg", title="The Nozzle Report Package", year="2011",  
url="http://www.gehlenborg.com" );  
  
webCitation <- newCitation( title="The Cancer Genome Atlas Website", url="http://tcga.cancer.gov" );  
  
fullCitation <- newCitation( authors="Nils Gehlenborg", title="Nozzle: An R Package for Report Generation",  
publication="Yet Another Bioinformatics Journal", issue="14", number="44", pages="783-789", year="2011",  
url="http://www.wikipedia.org" );  
  
r <- addToReferences( r, simpleCitation, webCitation, fullCitation );
```

- References

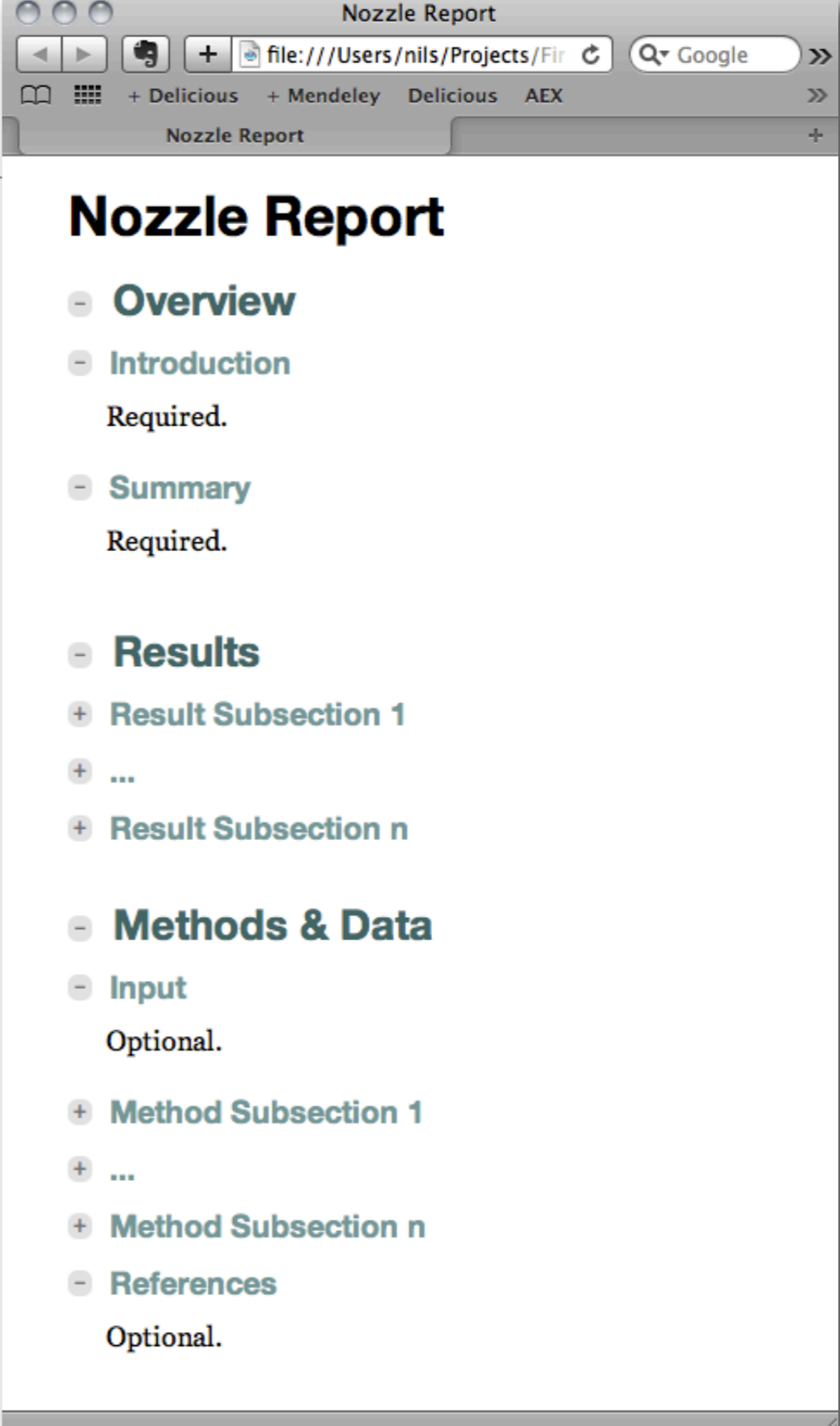
[1] Nils Gehlenborg, The Nozzle Report Package (2011)

[2] The Cancer Genome Atlas Website

[3] Nils Gehlenborg, Nozzle: An R Package for Report Generation, *Yet Another Bioinformatics Journal* **14**(44):783-789 (2011)

Structure of a Nozzle Report

```
r <- newReport( "Nozzle Report" );  
  
r <- addToIntroduction( r, newParagraph( "Required." ) );  
r <- addToSummary( r, newParagraph( "Required." ) );  
  
r <- addToResults( r, newSubSection( "Result Subsection 1" ) );  
r <- addToResults( r, newSubSection( "..." ) );  
r <- addToResults( r, newSubSection( "Result Subsection n" ) );  
  
r <- addToMethods( r, newSubSection( "Method Subsection 1" ) );  
r <- addToMethods( r, newSubSection( "..." ) );  
r <- addToMethods( r, newSubSection( "Method Subsection n" ) );  
  
r <- addToInput( r, newParagraph( "Optional." ) );  
r <- addToReferences( r, newParagraph( "Optional." ) );  
  
writeReport( r );
```



The screenshot shows a web browser window titled "Nozzle Report" displaying a document with a table of contents. The browser's address bar shows a file path: "file:///Users/nils/Projects/Fir". The document content is as follows:

Nozzle Report

- **Overview**
- **Introduction**
Required.
- **Summary**
Required.
- **Results**
 - + **Result Subsection 1**
 - + ...
 - + **Result Subsection n**
- **Methods & Data**
 - **Input**
Optional.
 - + **Method Subsection 1**
 - + ...
 - + **Method Subsection n**
- **References**
Optional.

Report Generation with Nozzle



- Conceptually a three stage process:
 1. **Generate** the elements that make up the report.
 - a. *Content elements* (paragraphs, figures, tables, etc.)
 - b. *Structural elements* (sections, lists, etc.)
 2. **Assemble** the elements into the structure of the report.
 3. **Write** the report to file.
- In practice it will be easier to combine parts of stage 1 and 2.

API Overview: Generation Stage

- **Structural elements:**

- newReport, newSection, newSubSection, newSubSubSection, newList, newResult

- **Content elements:**

- newParagraph, newTable, newFigure, newCitation, newParameterList

- **Content formatting:**

- asLink, asStrong, asEmphasis, asParameter, asValue, asFilename, asCode
- asReference, asSummary

Example

```
paragraph <- newParagraph( "Nozzle is an R package for report generation in ", asStrong( "Firehose" ) );  
subsection <- newSubSection( "About the Package" );  
report <- newReport( "The Nozzle Report" );
```

API Overview: Assembly Stage

- Reports are assembled in a bottom-up approach: generate content elements first and then attach them to structural elements and so on.
- General function to attach an element to a parent element:

```
addTo( parent, ..., row=NA, column=NA )
```

- Specialized functions to attach to predefined report elements:

```
addToResults( report, ... )  
addToMethods( report, ... )  
addToIntroduction( report, ... )  
addToSummary( report, ... )  
addToInput( report, ... )  
addToReferences( report, ... )
```

Example (cont'd)

```
paragraph <- newParagraph( "Nozzle is an R package for report generation in ", asStrong( "Firehose" ) );  
subsection <- newSubSection( "About the Package" );  
report <- newReport( "The Nozzle Report" );
```

```
subsection <- addTo( subsection, paragraph );  
report <- addToMethods( report, subsection );
```


API Overview: Writing Stage

```
writeReport( report, filename=DEFAULT.REPORT.FILENAME, debug=FALSE, level=PROTECTION.PUBLIC )
```

This function writes the report to a file as an HTML fragment for further processing by Firehose. Before the report is written, the optional, predefined “Methods & Data” subsections “Input” and “References” are removed from the report if no elements have been attached to them.

Arguments:

- **report** - The report element.
- *filename* - The filename for the report. The default is “report.html”.
- *debug* - If set to TRUE, a “stand-alone” version of the report will be generated that will work outside Firehose. This is for testing purposes only and this argument must be set to FALSE in production code.
- *level* - Determines which elements should be included in the report. If set to PROTECTION.PUBLIC, only elements marked with PROTECTION.PUBLIC will be included. If set to PROTECTION.TCGA, elements marked with PROTECTION.PUBLIC and PROTECTION.TCGA will be included. If set to PROTECTION.PRIVATE all elements will be included.

Value:

None.

Example (cont'd)

```
paragraph <- newParagraph( "Nozzle is an R package for report generation in ", asStrong( "Firehose" ) );  
subsection <- newSubSection( "About the Package" );  
report <- newReport( "The Nozzle Report" );
```

```
subsection <- addTo( subsection, paragraph );  
report <- addToMethods( report, subsection );
```

```
writeReport( report );
```

The Nozzle Report

- Overview
- + Introduction
- + Summary

- + Results

- Methods & Data
- About the Package

Nozzle is an R package for report generation in **Firehose**.

Application of Nozzle in Firehose

- Only at the very end of a pipeline in a separate GenePattern module.
- Best practice is to write all tables and figures to file before the report module is run.

Use of Nozzle in a GenePattern Module

- To ensure that always the latest version is used in a module:

```
install.packages( "nozzle-reports_v1.latest.tar.gz", "url/to/package" );  
library( "nozzle-reports_v1" );
```

- Changes that result in backwards incompatibilities will be reflected by an increase in the major release number:

```
install.packages( "nozzle-reports_v2.latest.tar.gz", "url/to/package" );  
library( "nozzle-reports_v2" );
```

- Developers will be required to update their modules to adopt a new major release. This is expected to be a rare event.

Tutorial: Assembling a Simple Report

- Correlation between methylation and gene expression levels.

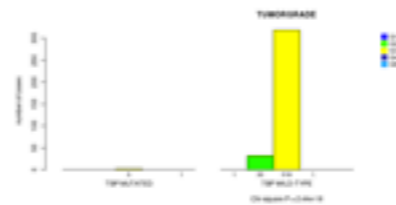
What are results?

CHST2	5	360	0.674	0.94	0.00011
GABRA6	6	359	0.966	4.46e-12	0.881
OR4F21	3	362	0.745	0.988	0.665
CYP11B1	7	358	0.292	0.99	0.0543
KCNJ12	5	360	0.674	0.968	0.897
TBP	4	361	0.892	2.44e-18	0.862
CDK12	9	356	0.571	0.998	0.392
LMF1	1	364	0.852	0.999	0.0926

Details for Association between TBP and Tumor Grade

GET HIGH-RES IMAGE

Figure 30. Distribution of samples for gene mutation status and clinical variable status.



Tutorial: Assembling a Report with Results

- A fictional extension of the previous report.

Questions?

- Are there any problems with the overall approach?
- Are features missing that are required for a particular report?
- Are features missing that would simplify certain aspects of the report generation?
 - Report initialization files (key-value pairs)?

Protecting Report Elements for Privacy

```
p1 <- newParagraph( "This paragraph is public.", protection=PROTECTION.PUBLIC );
p2 <- newParagraph( "This paragraph is TCGA-only.", protection=PROTECTION.TCGA );
subsection <- newSubSection( "Public Section" );
subsection <- addTo( subsection, p1, p2 );

# write a report (fragment) for public consumption: p2 will be stripped from the report before it is written to file
writeReport( addToResults( newReport( "Report: Public" ), subsection ), level=PROTECTION.PUBLIC );

# write a report (fragment) for consumption by TCGA members: both p1 and p2 will be included in the report
writeReport( addToResults( newReport( "Report: TCGA" ), subsection ), level=PROTECTION.TCGA );
```

public report:

- **Results**
- **Public Section**
This paragraph is public.

TCGA report:

- **Results**
- **Public Section**
This paragraph is public.
This paragraph is TCGA-only.

Behind the Scenes

- Every report element is a nested R “list” data structure and the report is assembled into a tree of nested lists (try `str(report)`).
- The `writeReport(...)` function preprocesses the report tree (e.g. it numbers figures, tables, etc. based on their position in the report and inserts the correct reference strings where requested) and then traverses the tree to produce the HTML code.
- The HTML of the report is only a fragment, not a full page. Firehose post-processes the fragments for consumption within and outside of Firehose.
- The CSS and JavaScript code required to display a report is embedded in the HTML fragment, making it a permanent, self-contained record once it is written to file.