

Basics of ReportingTools

Jason A. Hackney and Jessica L. Larson

October 1, 2012

Contents

1	Introduction	2
2	Basics of Reporting	2
3	Adding plots or text to a report	4
4	Styled Tables	4
5	More advanced reporting	4

1 Introduction

Frequently, when performing an analysis, it is helpful to be able to share these results in several formats at once: as HTML tables, csv files or even as R data packages. `ReportingTools` attempts to make this as painless as possible. At its heart, `ReportingTools` is nothing more than a set of S4 classes that describe what kinds of reports to generate, and a set of S4 methods that describe how to publish something in that format. In this vignette we will highlight the fundamentals of `ReportingTools`. `ReportingTools` has several methods for displaying microarray and RNA-seq results; for more details, please refer to the corresponding vignettes.

2 Basics of Reporting

The easiest type of report to generate is a csv file. This is done using the `CSVFile` class and the `publish` method. To start we'll create a `data.frame` that we'll use throughout the vignette.

```
> my.df <- data.frame(EGID = c("103", "104", "105", "106", "107"),
+                    RPKM = c(4, 5, 3, 100, 75),
+                    DE = c("Yes", "Yes", "No", "No", "No"))
> my.df
  EGID RPKM DE
1  103    4 Yes
2  104    5 Yes
3  105    3 No
4  106   100 No
5  107    75 No
```

Next, we'll create the `CSVFile` object to which we'll publish our results.

```
> library(ReportingTools)
> csvFile <- CSVFile(shortName = "my_csv_file",
+                   reportDirectory = "./reports/")
> publish(my.df, csvFile)
```

Obviously, this isn't much less work than just calling `write.csv` on the `data.frame` itself, but this is really just a toy example. We can also publish the `data.frame` as an HTML report.

```
> htmlRep <- HTMLReport(shortName = "my_html_file",
+                      reportDirectory = "./reports/")
> publish(my.df, htmlRep)
> finish(htmlRep)
```

It's necessary to call `finish` on the `HTMLReport`, to allow the contents to be written to the file.

It's also possible to publish the same object in two separate formats at once.

```
> csvFile2 <- CSVFile(shortName = "my_csv_file2",
+                   reportDirectory = "./reports/")
> htmlRep2 <- HTMLReport(shortName = 'my_html_file2',
+                      reportDirectory = "./reports/")
> publish(my.df, list(csvFile2, htmlRep2))
> finish(htmlRep2)
```

The same few lines of code could be used to publish, for example, the results of a `limma` differential expression analysis, or the results of a Gene Ontology analysis, all without worrying about coercing the objects to a tabular format ourselves. For more information, see the microarray and RNA-seq vignettes.

my_html_file

Search all columns: <input type="text"/>			Show	10	entries
			From	to	
EGID	RPKM	DE			
103	4	Yes			
104	5	Yes			
105	3	No			
106	100	No			
107	75	No			

Showing 1 to 5 of 5 entries First Previous 1 Next Last

Figure 1: Resulting page created by publish for my.df.

my_html_file2

Search all columns: <input type="text"/>			Show	10	entries
			From	to	
EGID	RPKM	DE			
103	4	Yes			
104	5	Yes			
105	3	No			
106	100	No			
107	75	No			

Showing 1 to 5 of 5 entries First Previous 1 Next Last

Figure 2: Resulting page created by calling publish for my.df to htmlRep2.

3 Adding plots or text to a report

To add additional text or plots to a report, simply open the report with `HTMLReport`, write to it (e.g., with `hwriter` functions), and then call `publish` on the original data frame and `finish` the report. Below we make a simple plot and then add it and some descriptive text to our report.

```
> png(filename="reports/barplot.png")
> barplot(my.df$RPKM, names.arg=my.df$EGID, xlab="EGID",
+         ylab="RPKM", main="Bar plot of RPKMs", col="blue")
> dev.off()
> library(hwriter)
> htmlRep3 <- HTMLReport(shortName = "my_html_file3",
+ reportDirectory = "./reports/")
> hwrite("Bar chart of results", p=page(htmlRep3), heading=2)
> himg<-hwriteImage("barplot.png", link="barplot.png")
> hwrite(himg, page(htmlRep3), br=TRUE)
> hwrite("Results Table", p=page(htmlRep3), heading=2)
> publish(my.df, htmlRep3)
> finish(htmlRep3)
```

4 Styled Tables

If you have a preferred table style, defined in `css`, it's easy to style your tables consistently using the `link.css` parameter when creating an `HTMLReport`. Below we use a simpler table format.

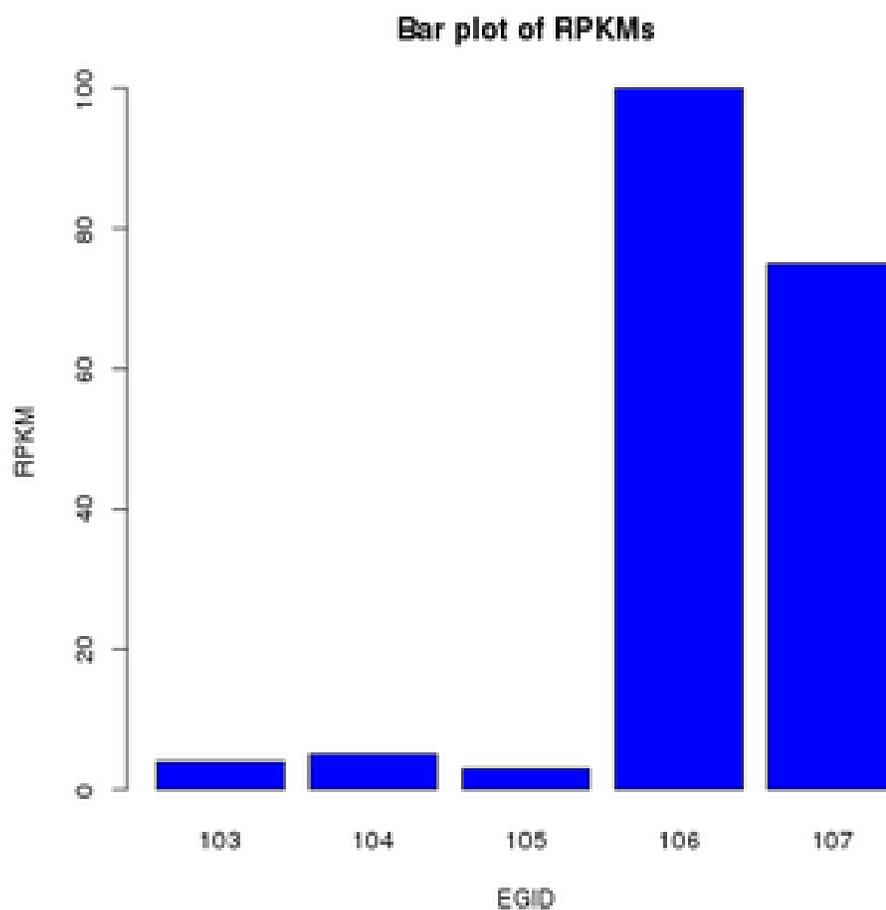
```
> css.file <- system.file('extdata/table.css', package='ReportingTools')
> htmlRep4 <- HTMLReport(shortName = 'my_html_file4',
+ reportDirectory = "./reports/", link.css=css.file)
> publish(my.df, htmlRep4)
> finish(htmlRep4)
```

5 More advanced reporting

For publishing experimental results, including how to publish a `limma`-based linear model and an `edgeR` RNA-seq analysis, please see the relevant vignette. Example output is shown below.

my_html_file3

Bar chart of results



Results Table

Show entries

Search all columns:

	From <input type="text"/>	to <input type="text"/>	
EGID	RPKM	DE	
103	4	Yes	
104	5	Yes	
105	3	No	
106	100	No	
107	75	No	5

Showing 1 to 5 of 5 entries [First](#) [Previous](#) [Next](#) [Last](#)

Figure 3: Resulting page created after adding additional figures and text.

my_html_file4

Show entries

Search all columns:

EGID From to DE

EGID	RPKM	DE
103	4	Yes
104	5	Yes
105	3	No
106	100	No
107	75	No

Showing 1 to 5 of 5 entries

First Previous | Next Last

EGID RPKM DE

Figure 4: Resulting page created by calling `publish` for `my.df` with specific styling.

Analysis of BCR/ABL translocation differential expression

Search all columns: Show entries

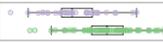
Probeld	Entrezid	Symbol	GeneName	Image	mol.bio/BCR/ABL logFC	mol.bio/BCR/ABL p-Value
1134_at	10188	TNK2	tyrosine kinase, non-receptor, 2		0.422	3.32e-05
1140_at	3682	ITGAE	integrin, alpha E (antigen CD103, human mucosal lymphocyte antigen 1; alpha polypeptide)		-0.820	2.71e-05
1249_at	2549	GAB1	GRB2-associated binding protein 1		1.080	1.95e-07

Figure 5: Resulting page created for analysis of a microarray study with `limma@`.