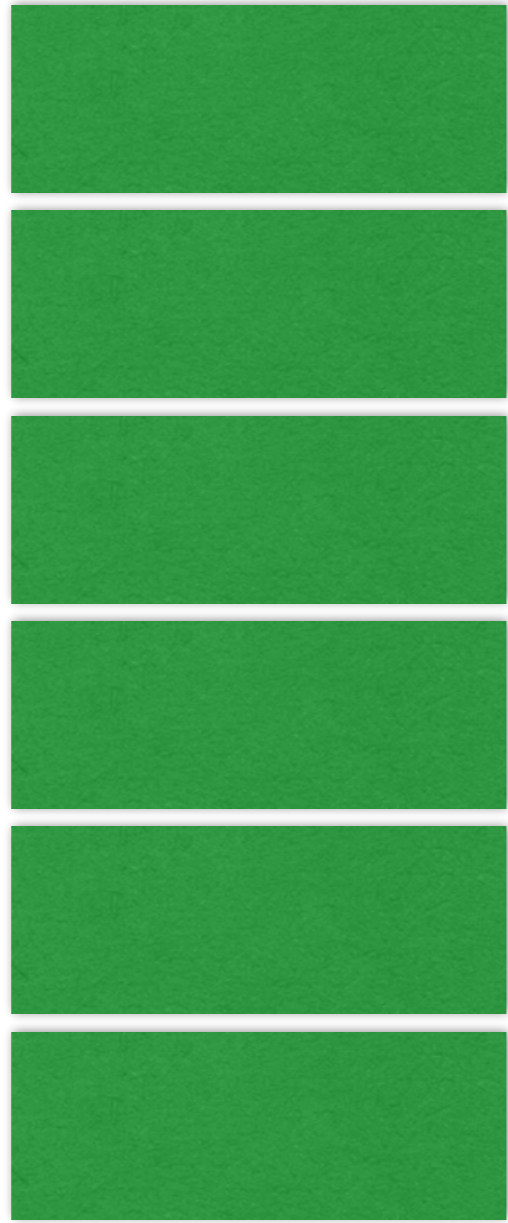


Jupyter

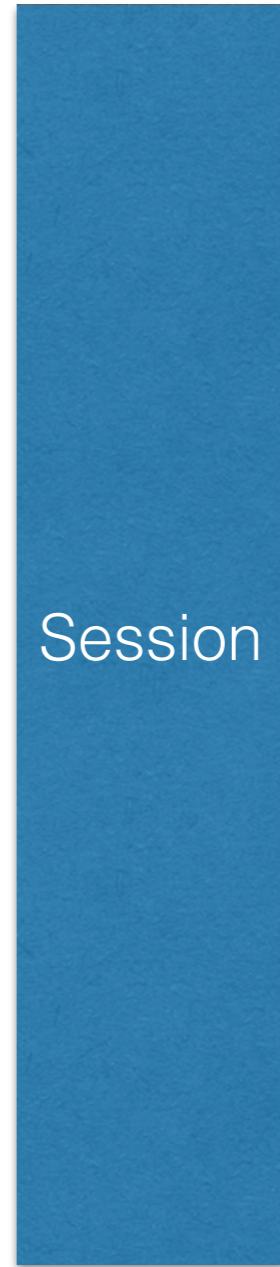
 jupyter



Chunks



→
Magic
←



Session

Files

Running

Clusters

Duplicate

Rename



Upload

New ▾



1



/ NBN



..



NBN.ipynb

Upload

New ▾



Text File

Folder

Terminal

Notebooks

Python 2

R



File Edit View Insert Cell Kernel Help

✎ | R O

          Code   CellToolbar

In []:

          Code   CellToolbar

```
In [3]: sprintf("This is an R session on version %s.%s", version$major, version$minor)
```

```
Out[3]: 'This is an R session on version 3.2.2'
```

```
In [ ]:
```

          Code   CellToolbar

```
In [8]: x <- 10
```

```
In [9]: x
```

```
Out[9]: 10
```

Demo time

Sharing

Out[11]:

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21	6	160	110	3.9	2.62	16.46	0	1	4	4
Mazda RX4 Wag	21	6	160	110	3.9	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108	93	3.85	2.32	18.61	1	1	4	1
Hornet 4 Drive	21.4	6	258	1							
Hornet Sportabout	18.7	8	360	1							
Valiant	18.1	6	225	1							

```
In [52]: data.plot + geom_point(data=coefficients, colour="blue", shape=3, size=10)
```



notebook files **include the output** so users don't need R

Sharing



nbviewer

A simple way to share Jupyter Notebooks

URL | GitHub username | GitHub username/repo | Gist ID

Go!

nbviewer.org



JUPYTER FAQ



OddsScraper / EU Referendum Odds.ipynb

EU Referendum Odds

I scraped the SkyBet odds on the EU referendum for a while in the run up to see how the odds changed. This notebook shows how I scraped it & what the data look like.

Getting data

I scraped the SkyBet odds using PhantomJS. I don't have a server at home so I borrowed one from Amazon & used Ansible to provision it to run a Docker container. The Docker container repeatedly loads the SkyBet page and dumps out the odds by pulling them out of the page using JQuery.

On the odds of an event are within an `td` tag & have the CSS class `oc-odds-desc`. Each has an attribute for the numerator and denominator of the odds. The PhantomJS script in `app/grab.js` hoovers up every odds entry on the page and dumps it as a timestamped JSON object.

The restart policy of the Docker container is set to `always` which means the script will poll the page as quickly as possible.

Transforming it

The `Makefile` in the root of the repository shows one quick post-processing step to make it easier to parse; as the JSON objects include commas, the comma was a bad choice of delimiter in the log file. To get around that I convert the first comma to a pipe with `sed` and use that as a delimiter instead.

Loading the data into R

Getting the data into R is quick with `read.table`, but there are some transformations that need to be done on the date and the odds JSON objects.

```
In [28]: data <- read.table('logs.tab', sep="|", stringsAsFactors = F, quote = '')
library(dplyr)
library(tidyrr)
library(jsonlite)
```

Fixing the date

The date needs to be parsed first. The datestams include two different timezones (with the same offset) which mean the offset can't

Allows anyone to view a notebook online

GitHub

The screenshot displays the GitHub interface for the repository 'mathew-hall / OddsScraper'. At the top, there are navigation links for 'Pull requests', 'Issues', and 'Gist'. Below the repository name, there are buttons for 'Unwatch', 'Star', and 'Fork'. A navigation bar includes 'Code', 'Issues', 'Pull requests', 'Projects', 'Wiki', 'Pulse', 'Graphs', and 'Settings'. The current branch is 'master', and the file path is 'OddsScraper / EU Referendum Odds.ipynb'. A commit by 'mathew-hall' is shown with the message 'Add some more plots' and a timestamp of '2cc4558 on Jul 6'. The file size is '4.85 MB'. The preview of the Jupyter Notebook shows the following content:

EU Referendum Odds

I scraped the SkyBet odds on the EU referendum for a while in the run up to see how the odds changed. This notebook shows how I scraped it & what the data look like.

Getting data

I scraped the SkyBet odds using PhantomJS. I don't have a server at home so I borrowed one from Amazon & used Ansible to provision it to run a Docker container. The Docker container repeatedly loads the SkyBet page and dumps out the odds by pulling them out of the page using JQuery.

On the odds of an event are within an a tag & have the CSS class `oc-odds-desc`. Each has an attribute for the numerator and denominator of the odds. The PhantomJS script in `app/gzab.js` hoovers up every odds entry on the page and dumps it as a timestamped JSON object.

The restart policy of the Docker container is set to `always` which means the script will poll the page as quickly as possible.

Transforming it

notebooks in a GitHub repo show nice previews

	RStudio + RMarkdown	Jupyter Notebooks	RStudio + Source Files
Mix code & data	Yup	Yup	Not really
Generate publication-ready docs	Yup	Not really	Not really
Share repeatable, remixable analyses	Yup	Yup	Not really
Write testable code	Not really	Not really	Yup
Super interactive	Yup*	Yup	Not really

Use cases

Prototyping analyses

Automatic data
cleaning workflows
(with bonus reports)

Generating data for use
in reports

Dialogs with
collaborators
(exploratory analyses)

Should I use it?

Just running some R
commands

R GUI/RStudio

Yeah

Prototyping an analysis

RMarkdown

Yeah

Preparing analysis for
publication

RMarkdown, Sweave

Maybe

Data cleaning

RMarkdown? R GUI/
RStudio

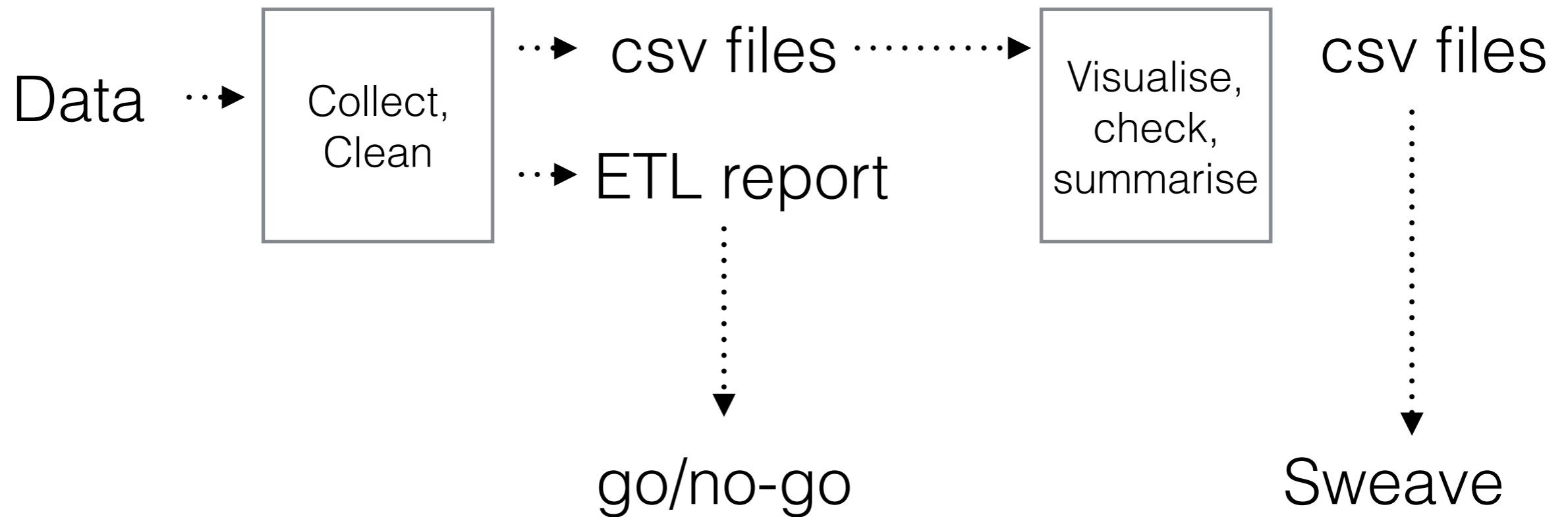
Yeah

Collaboration with a lot
of people using Git

RMarkdown ,R Package

Probably not

Example

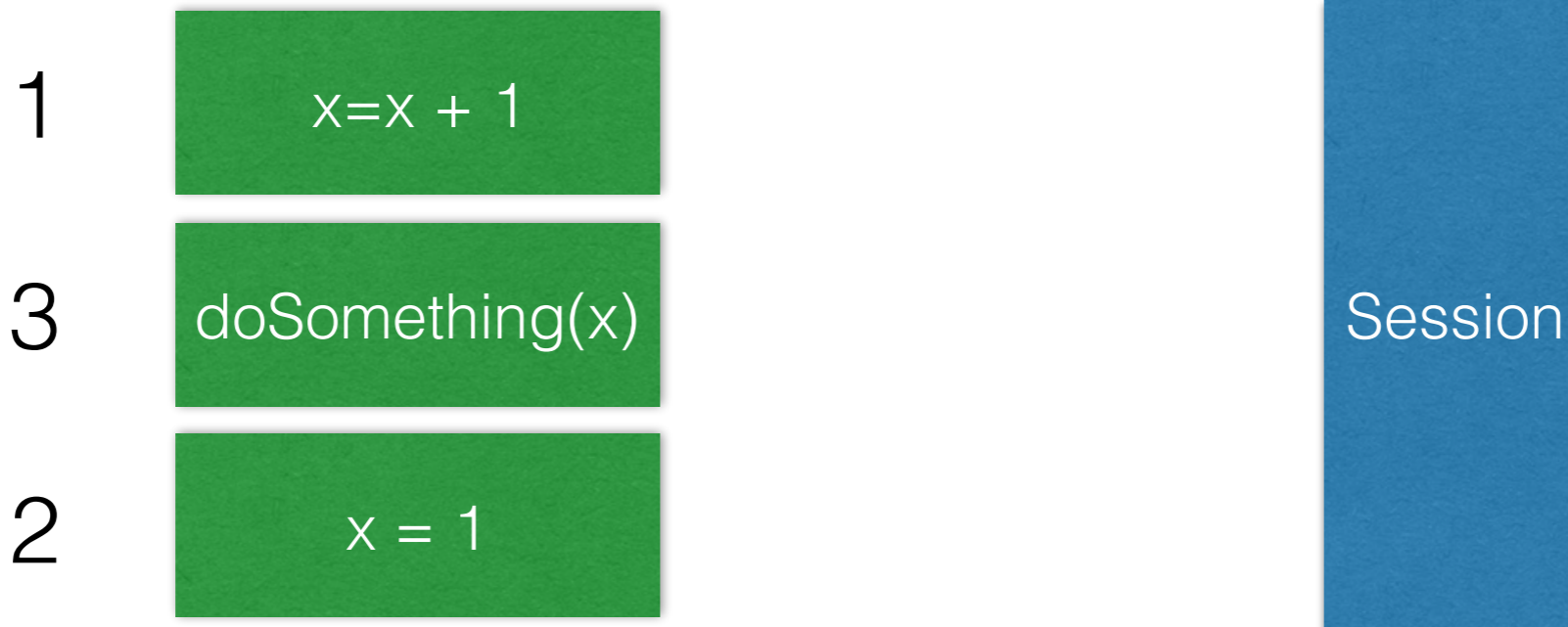


Used as a script with a nicer UI

Developed interactively & deployed as-is

Gotchas

 jupyter



State of the R Session depends on the order chunks run. It's easy to write non-linear notebooks by accident

Gotchas

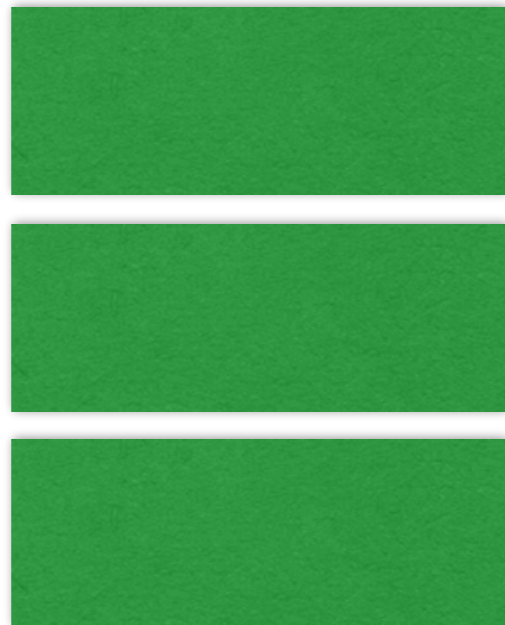
```
In [53]: library(superAwesomePackage)
```

```
Error in library(superAwesomePackage): there is no package called 'superAwesomePackage'
```

Need a mechanism to fetch dependencies

Gotchas

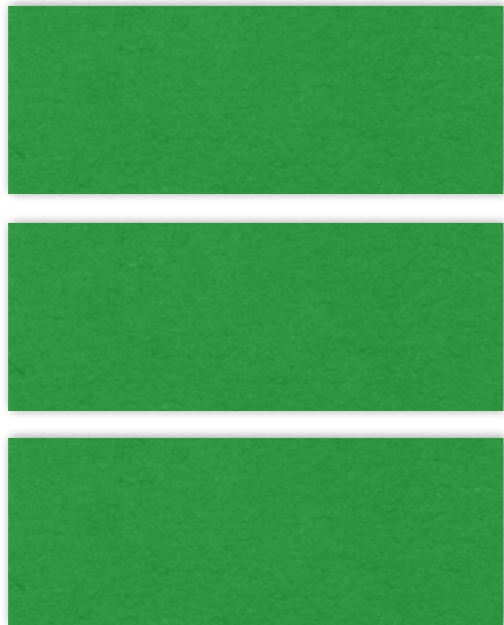
 jupyter



State of the R Session is not persistent,
only the stored outputs are kept in the
notebook file

Gotchas

 jupyter



 R Studio



Can turn a notebook into an R file
but much harder to go back

<http://stackoverflow.com/questions/32183164/best-practices-for-turning-jupyter-notebooks-into-python-scripts>

Tips

Don't get attached to
your R session

....▶

Routinely rerun **all**
chunks

Don't write enormous
notebooks

....▶

Migrate code to
packages when
sensible

Don't silently depend
on data/packages

....▶

Document dependencies/
automate fetching them

Resist temptation to
write spaghetti

....▶

Refactor as if the notebook
were a source file

Setup



1. Install Jupyter



2. Install IRKernel package



3. Register the kernel in Jupyter

Installing

- Recommended: use Anaconda (<https://www.continuum.io/downloads>) and R Essentials bundle (<http://anaconda.org/r/r-essentials>)
- Otherwise: pip install jupyter and follow instructions on (<https://irkernel.github.io>)
 - Will probably need to install zmq
- Option 3: `docker run -d -p 8888:8888 jupyter/r-notebook`

