

PREPARING YOUR MODEL FOR TAKE OFF!

Jonathan Stott

jonathan.stott@magairports.com



PRODUCTION?



PRE-FLIGHT CHECKS



?



PREPARING FOR TAKE OFF: COMMAND LINE ARGUMENTS

```
# my_script.R
# shows how to use command line arguments

doc <- "Usage: my_script.R [options] [--] <input_file> <output_file>

Options:
-h, --help          this help
-l LEVEL, --log LEVEL  logging level [default: INFO]
"

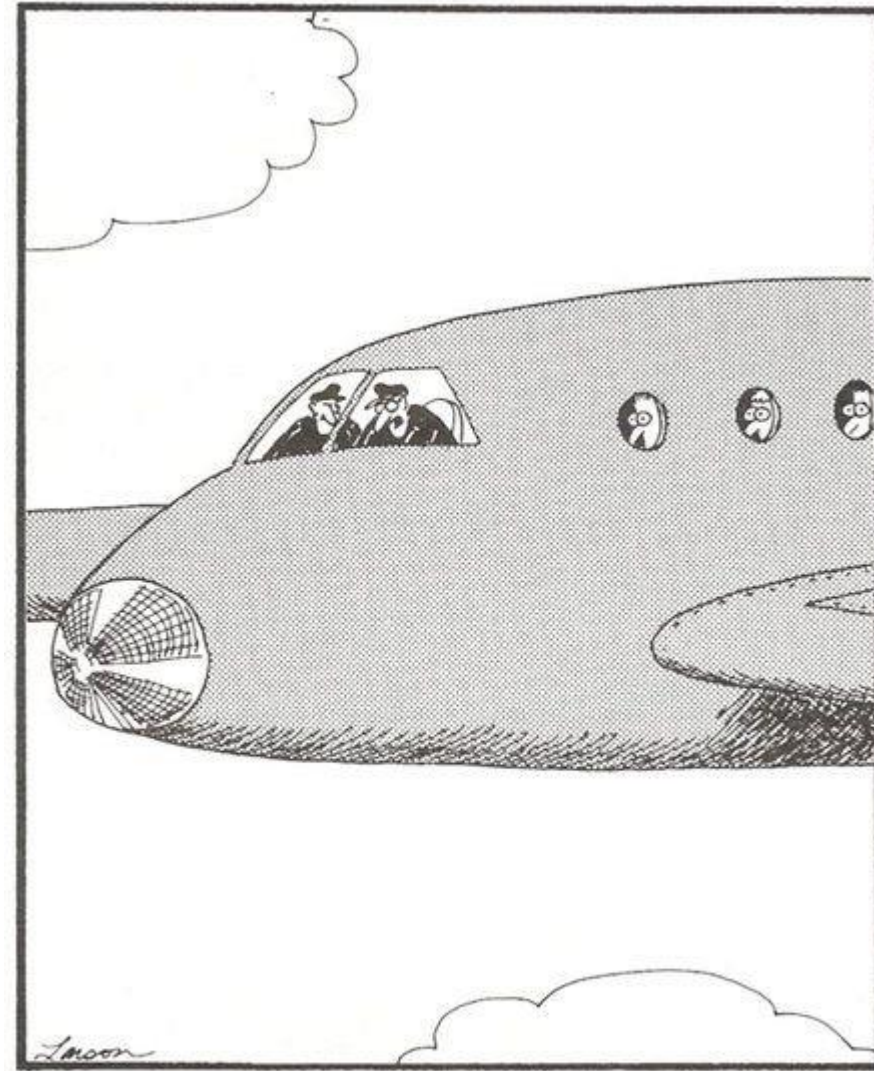
args <- docopt::docopt(doc)
print(args)

# Do things with the arguments
```

```
$ Rscript my_script.R input.csv output.rds
```

PREPARE FOR DISASTER

... or errors at least



"The fuel light's on, Frank! We're all going to die! ...
We're all going to die! ... Wait, wait. ... Oh, my
mistake—that's the intercom light."

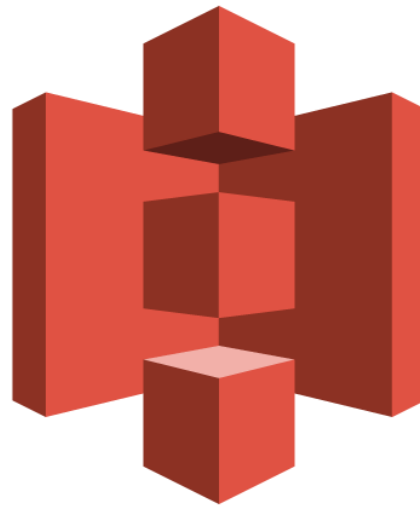
Fred Larson, Far Side

LOGS: YOUR FLIGHT RECORDER

LOG ALL THE THINGS



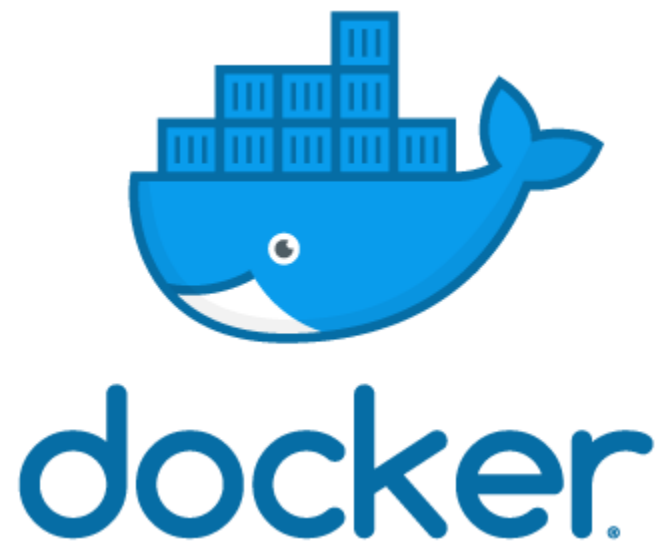
MAPS: WHERE DOES YOUR DATA LIVE?



Amazon
S3



SHIPPING CONTAINERS



```
FROM rocker/r-ver:latest
```

```
# install a package
```

```
RUN install2.r tidyverse
```

```
# copy code in
```

```
COPY . /app
```

```
# set working directory (important for R  
scripts!)
```

```
WORKDIR /app
```


SCHEDULED FLIGHTS



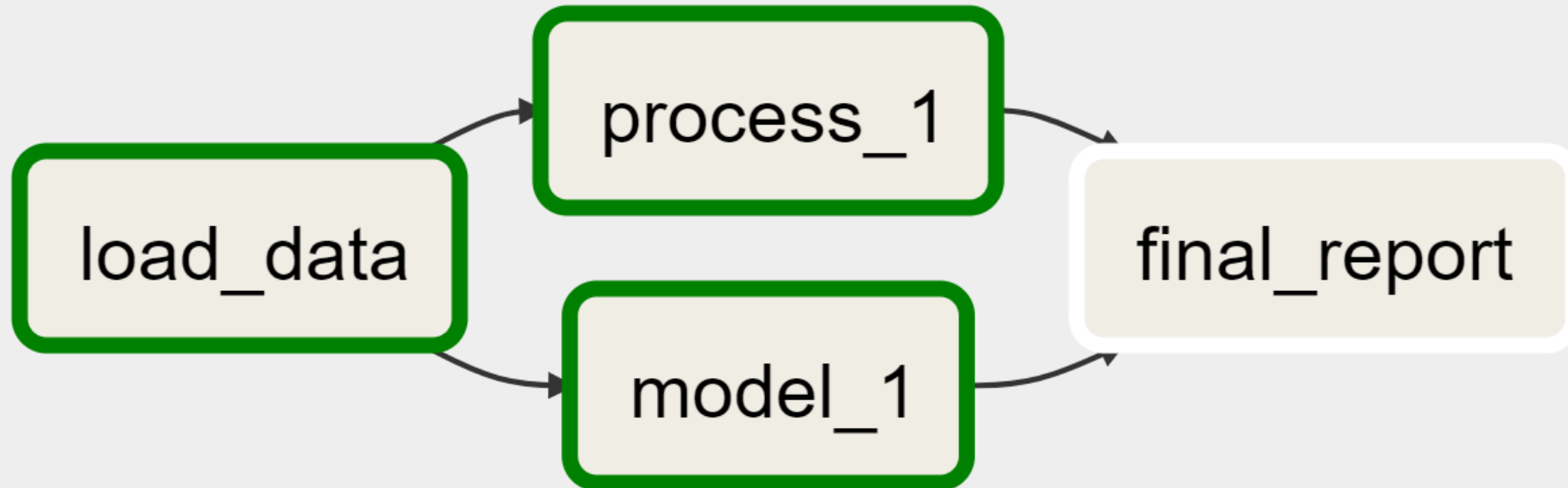
Photo by [Amin Salehi](#) on [Unsplash](#)

AIR TRAFFIC CONTROL WITH AIRFLOW



```
$ pip install apache-airflow
```



































































ROUTES...



LAYOUT OF THE CONTROL TOWER

DAGs

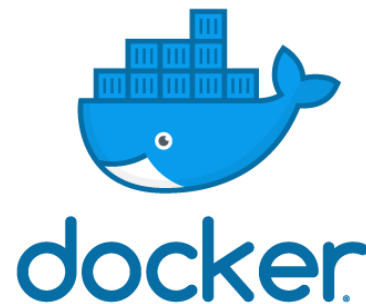
Search:

		DAG	Schedule	Owner	Recent Tasks 	Last Run 	DAG Runs 	Links
	On	RMS__budget_trackr	11 7***	rms	<div><div>3</div><div></div><div></div><div></div><div></div><div></div><div></div></div>	2018-11-13 07:11 	<div><div>76</div><div></div><div>2</div></div>	      
	On	RMS_db_load_v3	0 6***	rms	<div><div>39</div><div></div><div></div><div></div><div></div><div></div><div></div></div>	2018-11-13 06:00 	<div><div>64</div><div></div><div></div></div>	      
	On	RMS_v2_10_EMA	0 5***	rms	<div><div>41</div><div></div><div></div><div></div><div></div><div></div><div></div></div>	2018-11-13 05:00 	<div><div>64</div><div></div><div></div></div>	      
	On	RMS_v2_10_MAN	0 5***	rms	<div><div>41</div><div></div><div></div><div></div><div></div><div></div><div></div></div>	2018-11-13 05:00 	<div><div>64</div><div></div><div></div></div>	      
	On	RMS_v2_10_STN	0 5***	rms	<div><div>41</div><div></div><div></div><div></div><div></div><div></div><div></div></div>	2018-11-13 05:00 	<div><div>64</div><div></div><div></div></div>	      
	On	airflow-db-cleanup	@daily	operations	<div><div>8</div><div></div><div></div><div></div><div></div><div></div><div></div></div>	2018-11-13 00:00 	<div><div>48</div><div></div><div></div></div>	      
	On	customer_classification__full_v1	None	salesforce	<div><div></div><div></div><div></div><div></div><div></div><div></div><div></div></div>		<div><div>4</div><div></div><div>2</div></div>	      

DIFFERENT PLANES (OR OPERATORS) FOR DIFFERENT JOBS



Amazon
S3



WATCHING OUT FOR INCOMING DATA



Photo by [Nicolas Prieto](#) on [Unsplash](#)

BUILDING OUR OWN

```
class ROperator(BaseOperator):
    template_fields = ('script', 'arguments', 'cwd')

    @property
    def script(self):
        return self._script

    @property
    def arguments(self):
        return self._arguments

    @property
    def cwd(self):
        return self._cwd

    def __init__(self, script, arguments = [], cwd = None, **kwargs):
        self._script = script
        self._arguments = arguments
        self._cwd = cwd
        super(ROperator, self).__init__(**kwargs)

    def execute(self, context):
        r_proc = subprocess.Popen(
            ['/usr/bin/Rscript', self.script] + self.arguments,
            stdout=subprocess.PIPE, stderr=subprocess.STDOUT, close_fds=True,
            cwd=self.cwd)
        r_stdoutdata, _ = r_proc.communicate()
        self.log.info("Rscript output %s", escape_string(r_stdoutdata))

        if r_proc.returncode != 0:
            raise AirflowException("Rscript {} failed with return code {}".format(self.script, r_proc.returncode))

        return escape_string(r_stdoutdata)
```

BUILDING OUR OWN

```
class ROperator(BaseOperator):
    template_fields = ('script', 'arguments', 'cwd')

    @property
    def arguments(self):
        return self._arguments

    def __init__(self, script, arguments = [], cwd = None, **kwargs):
        self.script = script
        self.arguments = arguments
        self.cwd = cwd
        super(ROperator, self).__init__(**kwargs)

    def execute(self, context):
        r_proc = subprocess.Popen(
            ['/usr/bin/Rscript', self.script] + self.arguments,
            stdout=subprocess.PIPE, stderr=subprocess.STDOUT, close_fds=True,
            cwd=self.cwd)
        r_stdoutdata, _ = r_proc.communicate()
        self.log.info("Rscript output %s", escape_string(r_stdoutdata))

        if r_proc.returncode != 0:
            raise AirflowException("Rscript {} failed with return code {}".format(self.script, r_proc.returncode))

        return escape_string(r_stdoutdata)
```


BUILDING OUR OWN

```
class ROperator(BaseOperator):
    template_fields = ('script', 'arguments', 'cwd')

    def __init__(self, script, arguments = [], cwd = None, **kwargs):
        self.script = script
        self.arguments = arguments
        self.cwd = cwd
        super(ROperator, self).__init__(**kwargs)

    def execute(self, context):
        r_proc = subprocess.Popen(
            ['/usr/bin/Rscript', self.script] + self.arguments,
            stdout=subprocess.PIPE, stderr=subprocess.STDOUT, close_fds=True,
            cwd=self.cwd)
        r_stdoutdata, _ = r_proc.communicate()
        self.log.info("Rscript output %s", escape_string(r_stdoutdata))

        if r_proc.returncode != 0:
            raise AirflowException("Rscript {} failed with return code {}".format(self.script, r_proc.returncode))

        return escape_string(r_stdoutdata)
```

BUILDING OUR OWN

```
class ROperator(BaseOperator):
    template_fields = ('script', 'arguments', 'cwd')

    @apply_defaults
    def __init__(self, script, arguments=[], cwd=None, **kwargs):
        self.script = script
        self.arguments = arguments
        self.cwd = cwd
        super(ROperator, self).__init__(**kwargs)

    def execute(self, context):
        r_proc = subprocess.Popen(
            ['/usr/bin/Rscript', self.script] + self.arguments,
            stdout=subprocess.PIPE, stderr=subprocess.STDOUT, close_fds=True,
            cwd=self.cwd)
        r_stdoutdata, _ = r_proc.communicate()
        self.log.info("Rscript output %s", escape_string(r_stdoutdata))

        if r_proc.returncode != 0:
            raise AirflowException("Rscript {} failed with return code {}".format(self.script, r_proc.returncode))

        return escape_string(r_stdoutdata)
```

BUILDING OUR OWN

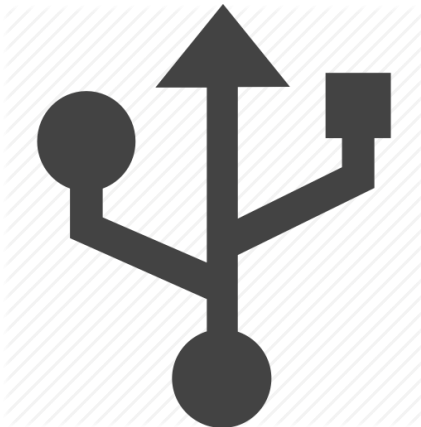
```
import subprocess
import re

from airflow.models import BaseOperator
from airflow.utils.decorators import apply_defaults
from airflow.exceptions import AirflowException

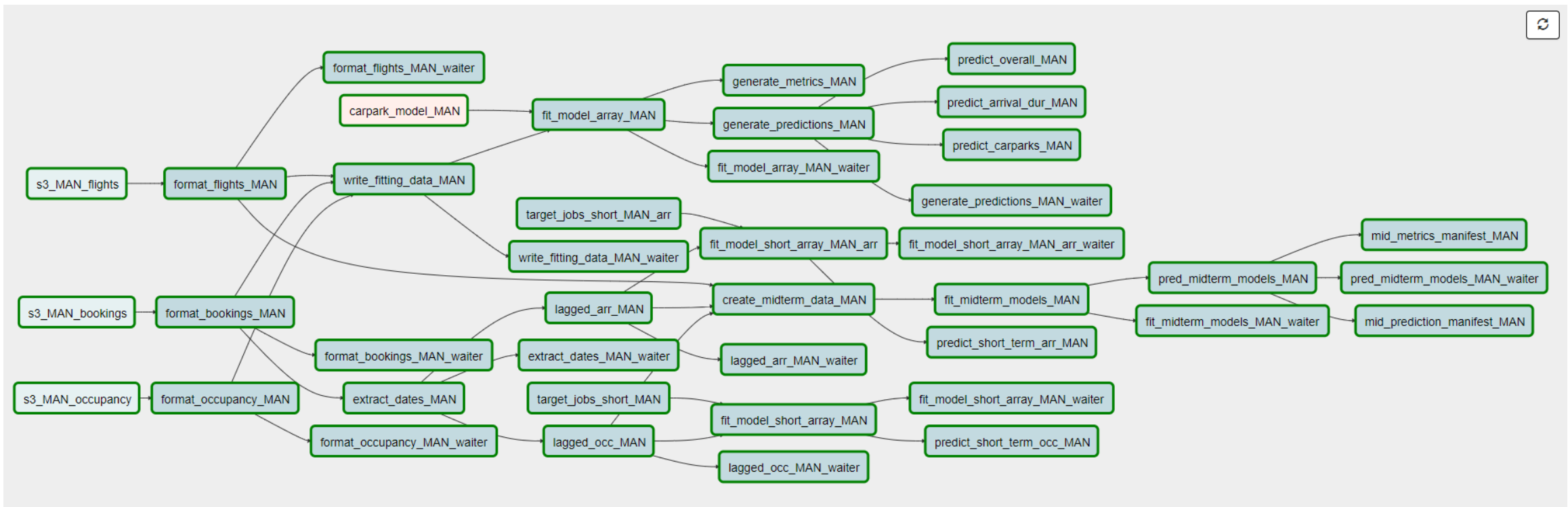
def escape_string(string):
    """escape the weird characters R sometimes outputs"""
    string = repr(string) # easy escape
    string = re.sub(r"\\n", "\n", string) # unescape new lines
    string = re.sub(r"\\t", "\t", string)
    return string

class ROperator(BaseOperator):
    # class definition ...
```

OTHER FEATURES IN THE CONTROL TOWER



COMPLICATED ROUTES ...



EXPLORING FURTHER



Jenkins



AWS / Azure Batch



LANDING


Converting a well architected R project into production is not that hard


Apache Airflow can help you to manage complex workflows

There's a lot more out there too

THANKS!

jonathan.stott@magairports.com

 namelessjon

 namelessjon

<https://github.com/namelessjon/Preparing-your-model-for-takeoff>

<https://speakerdeck.com/namelessjon/preparing-your-model-for-take-off>

<https://pixabay.com/en/night-flight-plane-airport-2307018/>