

23 - Criando um pipeline no portal II

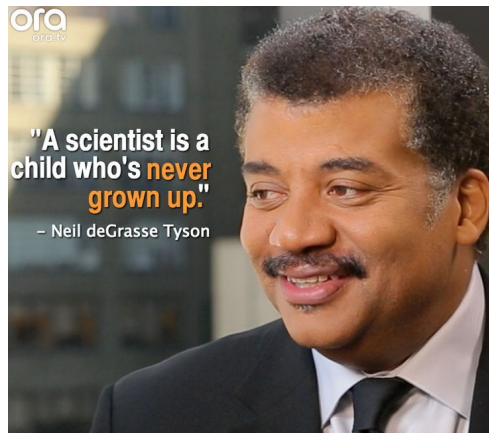
Tutorial

Julia Gschwend

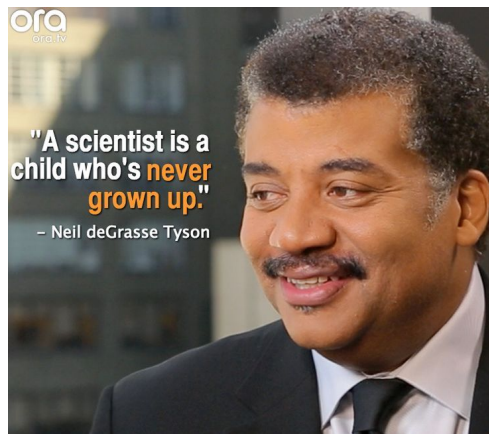
Momento mensagem motivacional

Motivos para desenvolver seu science workflow no DES Science Portal (1.0):

- Mais 2 anos de DES (no mínimo)
- Componentes poderão ser aproveitados no portal LSST
- Apoio da equipe LIneA (IT team + cientistas)
- Agilidade no desenvolvimento (E2E VS. Science)



- Mais 2 anos de DES (no mínimo)
- Componentes poderão ser aproveitados no portal LSST
- Apoio da equipe LInEA (IT team + cientistas)
- Agilidade no desenvolvimento (E2E VS. Science)



1. Desenhar workflow - criar doc pipeline description
2. Criar versão inicial do código no Jupyter (opcional)
3. Criar repositório para o novo pipeline contendo os arquivos:
 - pipeline.xml
 - VERSION (V00_00_00)
 - ChangeLog
4. Criar repositório para o componente 1 contendo os arquivos:
 - __init__.py
 - config/component_config.xml
 - input/component_input.xml
 - VERSION (V00_00_00)
 - ChangeLog
5. Criar repositório para o componente 2 contendo os arquivos:
 - __init__.py
 - output/component_output.xml
 - etc/ingest_table.rd
 - VERSION (V00_00_00)
 - ChangeLog
6. Atualizar repositório
7. Rodar no portal do desenvolvedor
8. Debugar/fine tuning
9. Versionar pipeline e componentes
10. Atualizar testing
11. Testar em testing
12. Fast deploy

1. Desenhar workflow - criar doc pipeline description
2. Criar versão inicial do código no Jupyter (opcional)
3. Criar repositório para o novo pipeline contendo os arquivos:
 - pipeline.xml
 - VERSION (V00_00_00)
 - ChangeLog
4. Criar repositório para o componente 1 contendo os arquivos:
 - __init__.py
 - config/component_config.xml
 - input/component_input.xml
 - VERSION (V00_00_00)
 - ChangeLog
5. Criar repositório para o componente 2 contendo os arquivos:
 - __init__.py
 - output/component_output.xml
 - etc/ingest_table.rd
 - VERSION (V00_00_00)
 - ChangeLog
6. Atualizar repositório
7. Rodar no portal do desenvolvedor
8. Debugar/fine tuning
9. Versionar pipeline e componentes
10. Atualizar testing
11. Testar em testing
12. Fast deploy

Exemplo: estimando massas de galáxias a partir das cores (Bell et al. 2003 / Taylor et al. 2011).

$$\log(\mathcal{M}(\mathcal{M}_{\odot})) = 1.15 + 0.7 \cdot (g - i)_0 - 0.40 \cdot M_i$$

Monthly Notices

of the
ROYAL ASTRONOMICAL SOCIETY

Mon. Not. R. Astron. Soc. **418**, 1587–1620 (2011)

doi:10.1111/j.1365-2966.2011.19536.x

Galaxy And Mass Assembly (GAMA): stellar mass estimates

Edward N. Taylor,^{1,2*} Andrew M. Hopkins,³ Ivan K. Baldry,⁴ Michael J. I. Brown,⁵
Simon P. Driver,⁶ Lee S. Kelvin,⁶ David T. Hill,⁶ Aaron S. G. Robotham,⁶
Joss Bland-Hawthorn,¹ D. H. Jones,⁵ R. G. Sharp,⁷ Daniel Thomas,⁸ Jochen Liske,⁹
Jon Loveday,¹⁰ Peder Norberg,¹¹ J. A. Peacock,¹¹ Steven P. Bamford,¹² Sarah Brough,³
Matthew Colless,³ Ewan Cameron,¹³ Christopher J. Conselice,¹² Scott M. Croom,¹
C. S. Frenk,¹⁴ Madusha Gunawardhana,¹ Konrad Kuijken,¹⁵ R. C. Nichol,⁸
H. R. Parkinson,¹¹ S. Phillipps,¹⁶ K. A. Pimbblet,⁵ C. C. Popescu,¹⁷ Matthew Prescott,⁴
W. J. Sutherland,¹⁸ R. J. Tuffs,¹⁹ Eelco van Kampen⁹ and D. Wijesinghe¹

¹Sydney Institute for Astronomy, School of Physics, University of Sydney, NSW 2006, Australia

²School of Physics, The University of Melbourne, Parkville, VIC 3010, Australia

³Australian Astronomical Observatory, PO Box 296, Epping, NSW 1710, Australia

⁴Astrophysics Research Institute, Liverpool John Moores University, Twelve Quays House, Egerton Wharf, Birkenhead CH41 1LD

⁵School of Physics, Monash University, Clayton, VIC 3800, Australia

⁶School of Physics & Astronomy, University of St Andrews, North Haugh, St Andrews KY16 9SS

⁷Research School of Astronomy & Astrophysics, Mount Stromlo Observatory, Weston Creek, ACT 2611, Australia

⁸Institute of Cosmology and Gravitation (ICG), University of Portsmouth, Portsmouth PO1 3FX

⁹European Southern Observatory, Karl-Schwarzschild-Str. 2, 85748 Garching, Germany

¹⁰Astronomy Centre, University of Sussex, Falmer, Brighton BN1 9QH

¹¹Institute for Astronomy, University of Edinburgh, Royal Observatory, Blackford Hill, Edinburgh EH9 3HJ

¹²Centre for Astronomy and Particle Theory, University of Nottingham, University Park, Nottingham NG7 2RD

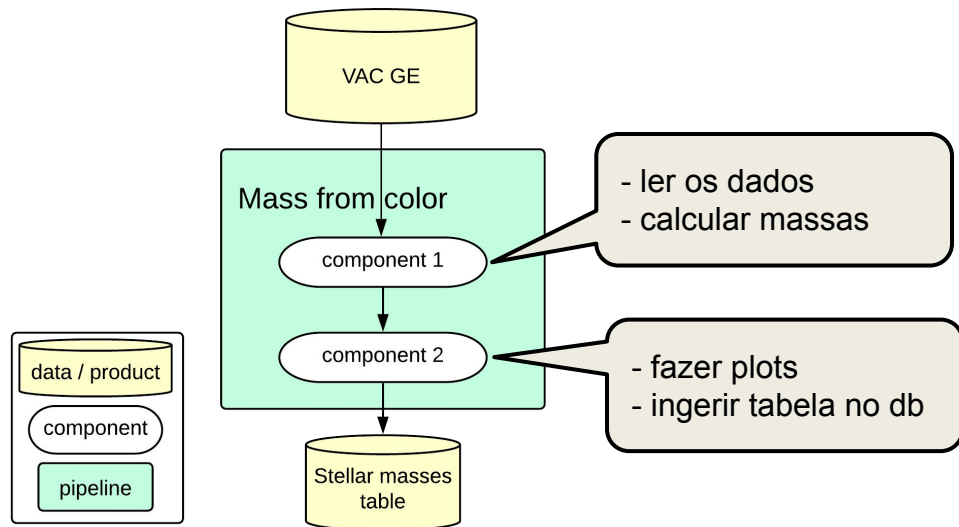
¹³Department of Physics, Swiss Federal Institute of Technology (ETH-Zürich), 8093 Zürich, Switzerland

¹⁴Institute for Computational Cosmology, Department of Physics, Durham University, Durham DH1 3LE

1- Desenho do Workflow

Exemplo: estimando massas de galáxias a partir das cores (Bell et al. 2003 / Taylor et al. 2011).

$$\log(\mathcal{M}(\mathcal{M}_{\odot})) = 1.15 + 0.7 \cdot (g - i)_0 - 0.40 \cdot M_i$$



Monthly Notices

of the
ROYAL ASTRONOMICAL SOCIETY

Mon. Not. R. Astron. Soc. **418**, 1587–1620 (2011)

doi:10.1111/j.1365-2966.2011.19536.x

Galaxy And Mass Assembly (GAMA): stellar mass estimates

Edward N. Taylor,^{1,2*} Andrew M. Hopkins,³ Ivan K. Baldry,⁴ Michael J. I. Brown,⁵ Simon P. Driver,⁶ Lee S. Kelvin,⁶ David T. Hill,⁶ Aaron S. G. Robotham,⁶ Joss Bland-Hawthorn,¹ D. H. Jones,⁵ R. G. Sharp,⁷ Daniel Thomas,⁸ Jochen Liske,⁹ Jon Loveday,¹⁰ Peder Norberg,¹¹ J. A. Peacock,¹¹ Steven P. Bamford,¹² Sarah Brough,³ Matthew Colless,³ Ewan Cameron,¹³ Christopher J. Conselice,¹² Scott M. Croom,¹ C. S. Frenk,¹⁴ Madusha Gunawardhana,¹ Konrad Kuijken,¹⁵ R. C. Nichol,⁸ H. R. Parkinson,¹¹ S. Phillipps,¹⁶ K. A. Pimbblet,⁵ C. C. Popescu,¹⁷ Matthew Prescott,⁴ W. J. Sutherland,¹⁸ R. J. Tuffs,¹⁹ Elco van Kampen⁹ and D. Wijesinghe¹

¹Sydney Institute for Astronomy, School of Physics, University of Sydney, NSW 2006, Australia

²School of Physics, The University of Melbourne, Parkville, VIC 3010, Australia

³Australian Astronomical Observatory, PO Box 296, Epping, NSW 1710, Australia

⁴Astrophysics Research Institute, Liverpool John Moores University, Twelve Quays House, Egerton Wharf, Birkenhead CH41 1LD

⁵School of Physics, Monash University, Clayton, VIC 3800, Australia

⁶School of Physics & Astronomy, University of St Andrews, North Haugh, St Andrews KY16 9SS

⁷Research School of Astronomy & Astrophysics, Mount Stromlo Observatory, Weston Creek, ACT 2611, Australia

⁸Institute of Cosmology and Gravitation (ICG), University of Portsmouth, Portsmouth PO1 3FX

⁹European Southern Observatory, Karl-Schwarzschild-Str. 2, 85748 Garching, Germany

¹⁰Astronomy Centre, University of Sussex, Falmer, Brighton BN1 9QH

¹¹Institute for Astronomy, University of Edinburgh, Royal Observatory, Blackford Hill, Edinburgh EH9 3HJ

¹²Centre for Astronomy and Particle Theory, University of Nottingham, University Park, Nottingham NG7 2RD

¹³Department of Physics, Swiss Federal Institute of Technology (ETH-Zürich), 8093 Zürich, Switzerland

¹⁴Institute for Computational Cosmology, Department of Physics, Durham University, Durham DH1 3LE

1. Desenhar workflow - criar doc pipeline description
2. Criar versão inicial do código no Jupyter (opcional)
3. Criar repositório para o novo pipeline contendo os arquivos:
 - pipeline.xml
 - VERSION (V00_00_00)
 - ChangeLog
4. Criar repositório para o componente 1 contendo os arquivos:
 - __init__.py
 - config/component_config.xml
 - input/component_input.xml
 - VERSION (V00_00_00)
 - ChangeLog
5. Criar repositório para o componente 2 contendo os arquivos:
 - __init__.py
 - output/component_output.xml
 - etc/ingest_table.rd
 - VERSION (V00_00_00)
 - ChangeLog
6. Atualizar repositório
7. Rodar no portal do desenvolvedor
8. Debugar/fine tuning
9. Versionar pipeline e componentes
10. Atualizar testing
11. Testar em testing
12. Fast deploy

Exemplo: estimando massas de galáxias a partir das cores (Bell et al. 2003 / Taylor et al. 2011).

$$\log(\mathcal{M}(\mathcal{M}_{\odot})) = 1.15 + 0.7 \cdot (g - i)_0 - 0.40 \cdot M_i$$

jupyter session_23



jupyter session_23 (autosaved)

File Edit View Insert Cell Kernel Widgets Help Trusted science

LineA Bootcamp 2019

Sessão 23 - Criando um pipeline II

Resumo

Este tutorial irá mostrar o passo a passo para criar um novo pipeline no DES Science Portal. O código abaixo será adaptado para rodar no portal.

Os slides desta sessão estão disponíveis [neste link](#).

Exemplo: estimando massas de galáxias a partir das cores

Para ilustrar a criação de um novo pipeline, vamos utilizar o método de estimativa de massas estelares de galáxias (\mathcal{M}) proposto por [Bell et al. \(2003\)](#) e revisado por [Taylor et al. \(2011\)](#), a partir da magnitude absoluta (M_i) e a cor no ref. de repouso $((g - r)_0)$.

$$\log(\mathcal{M}(\mathcal{M}_{\odot})) = 1.15 + 0.7 \cdot (g - i)_0 - 0.40 \cdot M_i$$

```
In [ ]: %matplotlib inline
import numpy as np
import pylab as pl
import pandas as pd
import seaborn as sns
import sqlalchemy
```

1. Desenhar workflow - criar doc pipeline description
2. Criar versão inicial do código no Jupyter (opcional)
3. Criar repositório para o novo pipeline contendo os arquivos:
 - pipeline.xml
 - VERSION (V00_00_00)
 - ChangeLog
4. Criar repositório para o componente 1 contendo os arquivos:
 - __init__.py
 - config/component_config.xml
 - input/component_input.xml
 - VERSION (V00_00_00)
 - ChangeLog
5. Criar repositório para o componente 2 contendo os arquivos:
 - __init__.py
 - output/component_output.xml
 - etc/ingest_table.rd
 - VERSION (V00_00_00)
 - ChangeLog
6. Atualizar repositório
7. Rodar no portal do desenvolvedor
8. Debugar/fine tuning
9. Versionar pipeline e componentes
10. Atualizar testing
11. Testar em testing
12. Fast deploy

3. Criar repositório para o novo pipeline

```
$ mkdir new_pipeline.tmp
$ cd new_pipeline.tmp
$ git init

$ vim pipeline.xml
$ vim VERSION
$ vim ChangeLog

$ git add .
$ git commit -m "Initial commit for a new submodule"
$ git tag `cat VERSION`
```

1. Desenhar workflow - criar doc pipeline description
2. Criar versão inicial do código no Jupyter (opcional)
3. Criar repositório para o novo pipeline contendo os arquivos:
 - pipeline.xml
 - VERSION (V00_00_00)
 - ChangeLog
4. Criar repositório para o componente 1 contendo os arquivos:
 - __init__.py
 - config/component_config.xml
 - input/component_input.xml
 - VERSION (V00_00_00)
 - ChangeLog
5. Criar repositório para o componente 2 contendo os arquivos:
 - __init__.py
 - output/component_output.xml
 - etc/ingest_table.rd
 - VERSION (V00_00_00)
 - ChangeLog
6. Atualizar repositório
7. Rodar no portal do desenvolvedor
8. Debugar/fine tuning
9. Versionar pipeline e componentes
10. Atualizar testing
11. Testar em testing
12. Fast deploy

4,5. Criar repositório para o novo component

```
$ mkdir new_component.tmp
$ cd new_component.tmp
$ git init

$ vim __init__.py
$ vim VERSION
$ vim ChangeLog

$ git add .
$ git commit -m "Initial commit for a new submodule"
$ git tag `cat VERSION`
```

3,4,5. Criar repositório para o novo pipeline/component

helpdesk@linea.gov.br

Email subject: product creation: PRODUCTNAME VXX_YY_ZZ

(e.g.: pipeline-cluster_halo_matching V00_00_00, component-subsets_separation V00_00_00)

Email content:

Please could you deploy this new product?

Product Type: (pipeline/component)

Product Location: full path to newmodule.tmp

Does it involve a new data class? (y/n)

Does it involve a new version of package in Tawala/EUPS? (y/n)

Documentação: [NT 010-2018 - Release Procedure for Developers](#)

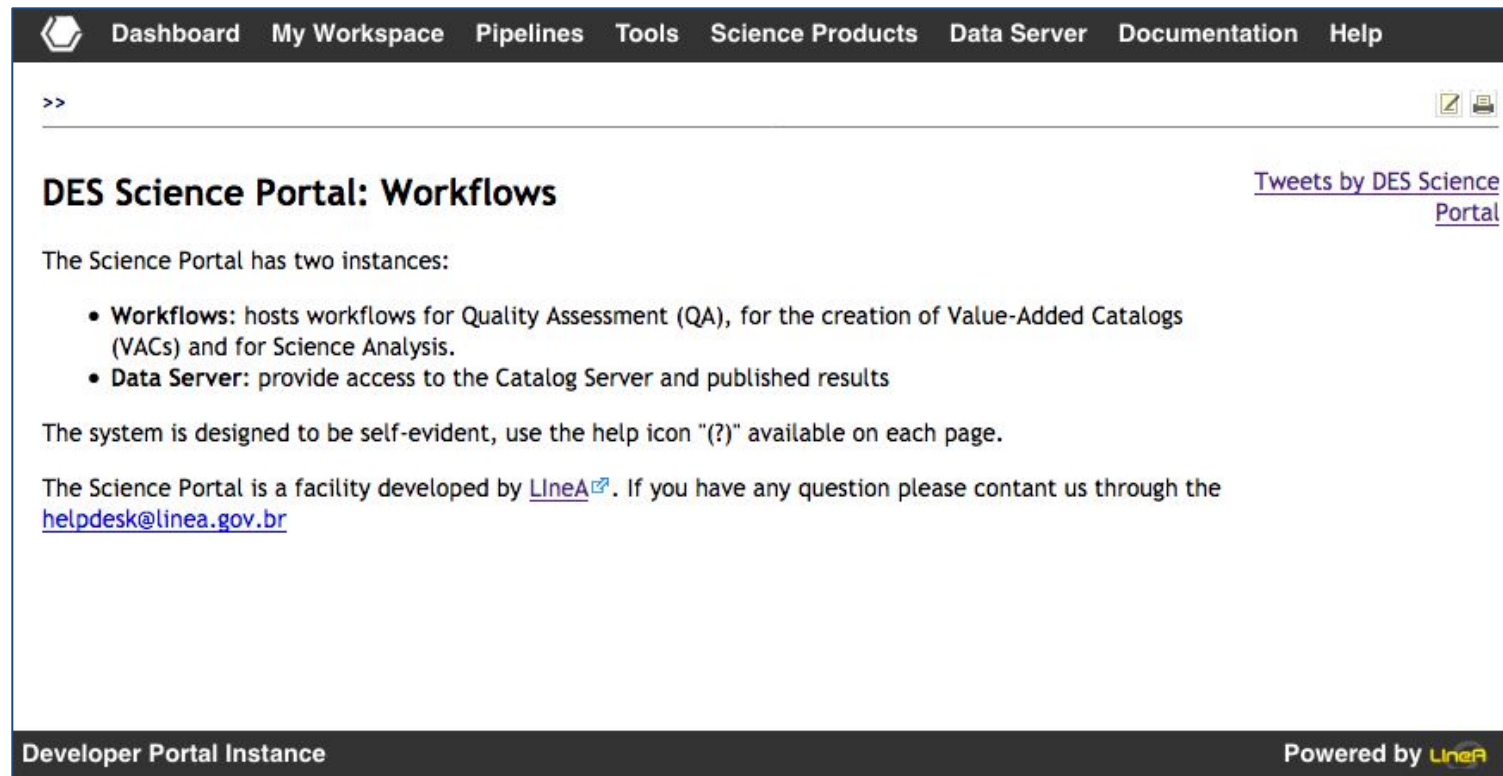
1. Desenhar workflow - criar doc pipeline description
2. Criar versão inicial do código no Jupyter (opcional)
3. Criar repositório para o novo pipeline contendo os arquivos:
 - pipeline.xml
 - VERSION (V00_00_00)
 - ChangeLog
4. Criar repositório para o componente 1 contendo os arquivos:
 - __init__.py
 - config/component_config.xml
 - input/component_input.xml
 - VERSION (V00_00_00)
 - ChangeLog
5. Criar repositório para o componente 2 contendo os arquivos:
 - __init__.py
 - output/component_output.xml
 - etc/ingest_table.rd
 - VERSION (V00_00_00)
 - ChangeLog
6. **Atualizar repositório**
7. Rodar no portal do desenvolvedor
8. Debugar/fine tuning
9. Versionar pipeline e componentes
10. Atualizar testing
11. Testar em testing
12. Fast deploy

6. Atualizar repositório (depois da resposta do ticket)

```
$ cd $PORTAL_ROOT/des  
$ git pull  
$ git submodule init  
$ git submodule update
```

1. Desenhar workflow - criar doc pipeline description
2. Criar versão inicial do código no Jupyter (opcional)
3. Criar repositório para o novo pipeline contendo os arquivos:
 - pipeline.xml
 - VERSION (V00_00_00)
 - ChangeLog
4. Criar repositório para o componente 1 contendo os arquivos:
 - __init__.py
 - config/component_config.xml
 - input/component_input.xml
 - VERSION (V00_00_00)
 - ChangeLog
5. Criar repositório para o componente 2 contendo os arquivos:
 - __init__.py
 - output/component_output.xml
 - etc/ingest_table.rd
 - VERSION (V00_00_00)
 - ChangeLog
6. Atualizar repositório
7. Rodar no portal do desenvolvedor
8. Debugar/fine tuning
9. Versionar pipeline e componentes
10. Atualizar testing
11. Testar em testing
12. Fast deploy

7. Rodar no portal do desenvolvedor



The screenshot shows the 'Developer Portal Instance' of the DES Science Portal. The top navigation bar includes links for Dashboard, My Workspace, Pipelines, Tools, Science Products, Data Server, Documentation, and Help. The main content area has a header 'DES Science Portal: Workflows' and a sub-header 'Tweets by DES Science Portal'. The text explains that the Science Portal has two instances: Workflows (for Quality Assessment and Value-Added Catalogs) and Data Server (for access to the Catalog Server and published results). It also mentions that the system is self-evident and provides a help icon. The footer indicates it is powered by LineA.

>>

DES Science Portal: Workflows

[Tweets by DES Science Portal](#)


The Science Portal has two instances:

- **Workflows:** hosts workflows for Quality Assessment (QA), for the creation of Value-Added Catalogs (VACs) and for Science Analysis.
- **Data Server:** provide access to the Catalog Server and published results

The system is designed to be self-evident, use the help icon "(?)" available on each page.

The Science Portal is a facility developed by [LineA](#). If you have any question please contact us through the helpdesk@linea.gov.br

Developer Portal Instance

Powered by 

1. Desenhar workflow - criar doc pipeline description
2. Criar versão inicial do código no Jupyter (opcional)
3. Criar repositório para o novo pipeline contendo os arquivos:
 - pipeline.xml
 - VERSION (V00_00_00)
 - ChangeLog
4. Criar repositório para o componente 1 contendo os arquivos:
 - __init__.py
 - config/component_config.xml
 - input/component_input.xml
 - VERSION (V00_00_00)
 - ChangeLog
5. Criar repositório para o componente 2 contendo os arquivos:
 - __init__.py
 - output/component_output.xml
 - etc/ingest_table.rd
 - VERSION (V00_00_00)
 - ChangeLog
6. Atualizar repositório
7. Rodar no portal do desenvolvedor
8. **Debugar/fine tuning**
9. Versionar pipeline e componentes
10. Atualizar testing
11. Testar em testing
12. Fast deploy

8. Debug /fine tuning

```
$ cd /mnt/scratch/users/<username>/master_des/0000<process number>  
$ cd <task name>_1_sandbox  
  
$ # change code
```

8. Debug /fine tuning

```
$ cd /mnt/scratch/users/<username>/mast  
$ cd <task name>_1_sandbox  
  
$ # change code
```

```
$ cd $PORTAL_ROOT/<path to submodule>  
$ git checkout master  
$ git pull  
  
$ vim __init__.py # make changes  
$ git status  
$ git add __init__.py  
$ git commit -m "<commit message>"
```

Repetir quantas
vezes for
necessário

8. Debug /fine tuning

```
$ cd /mnt/scratch/users/<username>/master_des/0000<process number>
$ cd <task name>_1_sandbox

$ # change code

$ # run in sandbox (debug mode)
--- local: ---
$ python -c "import components.<component name> as alias; alias.run()"
--- using the cluster: ---
$ condor_run 'python -c "import components.<component name> as alias; alias.run()" ' &

--- if changes in XML: ---
$ ingestWorkflow -W <pipeline name> --status enabled
```


1. Desenhar workflow - criar doc pipeline description
2. Criar versão inicial do código no Jupyter (opcional)
3. Criar repositório para o novo pipeline contendo os arquivos:
 - pipeline.xml
 - VERSION (V00_00_00)
 - ChangeLog
4. Criar repositório para o componente 1 contendo os arquivos:
 - __init__.py
 - config/component_config.xml
 - input/component_input.xml
 - VERSION (V00_00_00)
 - ChangeLog
5. Criar repositório para o componente 2 contendo os arquivos:
 - __init__.py
 - output/component_output.xml
 - etc/ingest_table.rd
 - VERSION (V00_00_00)
 - ChangeLog
6. Atualizar repositório
7. Rodar no portal do desenvolvedor
8. Debugar/fine tuning
9. **Versionar pipeline e componentes**
10. Atualizar testing
11. Testar em testing
12. Fast deploy

9. Versionar component/pipeline

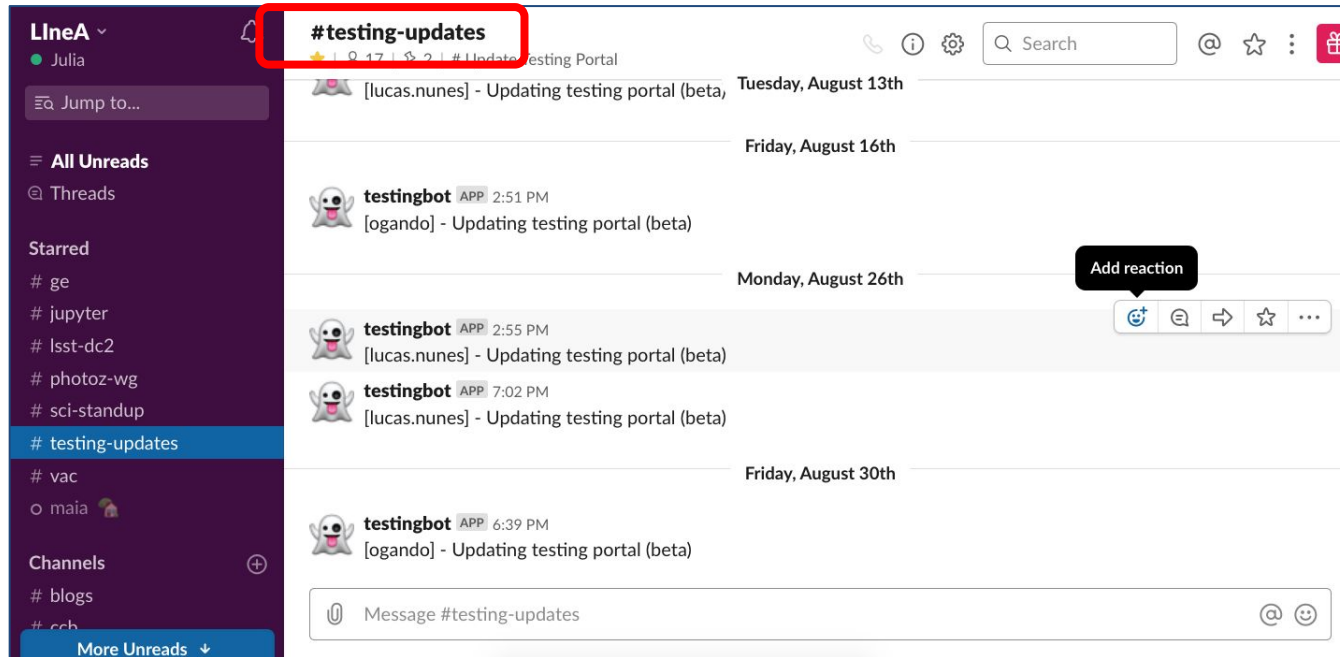
```
$ vim VERSION # change version number
$ vim ChangeLog # add high level description of changes
$ git add VERSION ChangeLog
$ git commit -m "Updating submodule with version `cat VERSION`."

$ git tag `cat VERSION`
$ git push origin
$ git push --tag
```

1. Desenhar workflow - criar doc pipeline description
2. Criar versão inicial do código no Jupyter (opcional)
3. Criar repositório para o novo pipeline contendo os arquivos:
 - pipeline.xml
 - VERSION (V00_00_00)
 - ChangeLog
4. Criar repositório para o componente 1 contendo os arquivos:
 - __init__.py
 - config/component_config.xml
 - input/component_input.xml
 - VERSION (V00_00_00)
 - ChangeLog
5. Criar repositório para o componente 2 contendo os arquivos:
 - __init__.py
 - output/component_output.xml
 - etc/ingest_table.rd
 - VERSION (V00_00_00)
 - ChangeLog
6. Atualizar repositório
7. Rodar no portal do desenvolvedor
8. Debugar/fine tuning
9. Versionar pipeline e componentes
10. **Atualizar testing**
11. Testar em testing
12. Fast deploy


10. Atualizar testing - Slack #testing-updates



```
$ sudo -H -u testing -i /home/testing/update_portal.sh
```



1. Desenhar workflow - criar doc pipeline description
2. Criar versão inicial do código no Jupyter (opcional)
3. Criar repositório para o novo pipeline contendo os arquivos:
 - pipeline.xml
 - VERSION (V00_00_00)
 - ChangeLog
4. Criar repositório para o componente 1 contendo os arquivos:
 - __init__.py
 - config/component_config.xml
 - input/component_input.xml
 - VERSION (V00_00_00)
 - ChangeLog
5. Criar repositório para o componente 2 contendo os arquivos:
 - __init__.py
 - output/component_output.xml
 - etc/ingest_table.rd
 - VERSION (V00_00_00)
 - ChangeLog
6. Atualizar repositório
7. Rodar no portal do desenvolvedor
8. Debugar/fine tuning
9. Versionar pipeline e componentes
10. Atualizar testing
11. Testar em testing
12. Fast deploy

7. Rodar no portal Testing

 Dashboard My Workspace Pipelines Tools Science Products Data Server Documentation Help

>>  


DES Science Portal: Workflows

The Science Portal has two instances:

- **Workflows:** hosts workflows for Quality Assessment (QA), for the creation of Value-Added Catalogs (VACs) and for Science Analysis.
- **Data Server:** provide access to the Catalog Server and published results

The system is designed to be self-evident, use the help icon "(?)" available on each page.

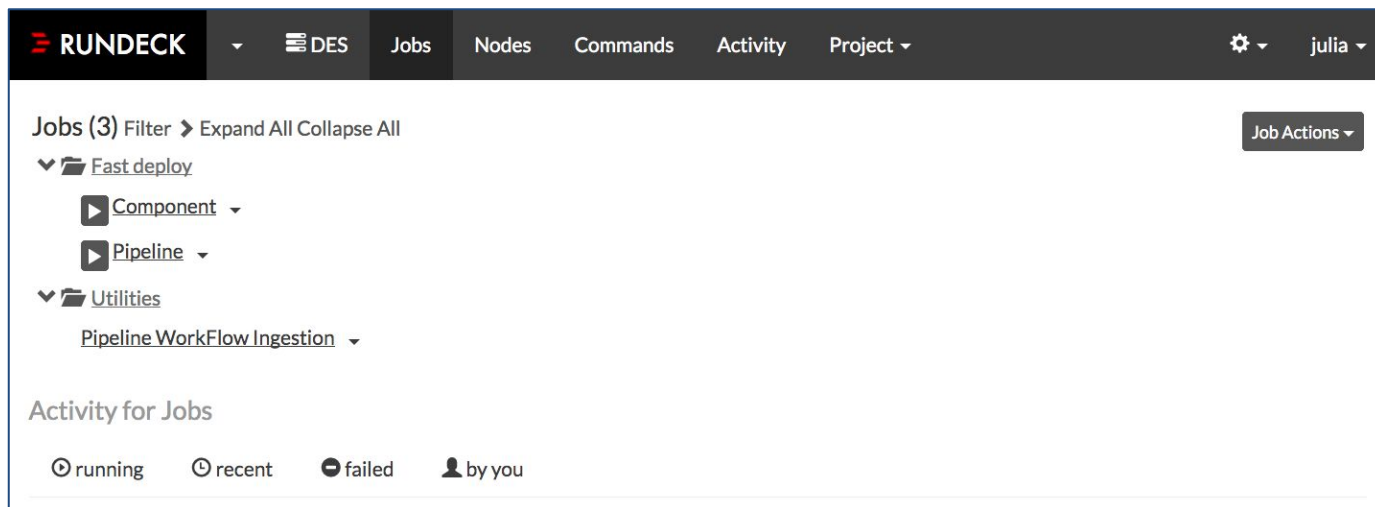
The Science Portal is a facility developed by [LineA](#). If you have any question please contact us through the helpdesk@linea.gov.br

Developer Portal Instance Powered by 

1. Desenhar workflow - criar doc pipeline description
2. Criar versão inicial do código no Jupyter (opcional)
3. Criar repositório para o novo pipeline contendo os arquivos:
 - pipeline.xml
 - VERSION (V00_00_00)
 - ChangeLog
4. Criar repositório para o componente 1 contendo os arquivos:
 - __init__.py
 - config/component_config.xml
 - input/component_input.xml
 - VERSION (V00_00_00)
 - ChangeLog
5. Criar repositório para o componente 2 contendo os arquivos:
 - __init__.py
 - output/component_output.xml
 - etc/ingest_table.rd
 - VERSION (V00_00_00)
 - ChangeLog
6. Atualizar repositório
7. Rodar no portal do desenvolvedor
8. Debugar/fine tuning
9. Versionar pipeline e componentes
10. Atualizar testing
11. Testar em testing
12. Fast deploy

12. Deploy em production

- E2E → update pipeline history (google drive) + email to change control board (ccb@linea.gov.br)
- Science Workflow → update pipeline history (google drive) + fast deploy (<https://fox.linea.gov.br/project/DES/jobs>)



Dúvidas?

Extra

1. Desenhar workflow
2. Criar versão inicial do código no Jupyter (opcional)
3. Criar repositório para o novo pipeline contendo os arquivos:
 - pipeline.xml
 - VERSION (V00_00_00)
 - ChangeLog
4. Criar repositório para o componente 1 contendo os arquivos:
 - __init__.py
 - config/component_config.xml
 - input/component_input.xml
 - VERSION (V00_00_00)
 - ChangeLog
5. Criar repositório para o componente 2 contendo os arquivos:
 - __init__.py
 - output/component_output.xml
 - etc/ingest_table.rd
 - VERSION (V00_00_00)
 - ChangeLog
6. Atualizar repositório
7. Rodar no portal do desenvolvedor
8. Debugar/fine tuning
9. Versionar pipeline e componentes
10. Atualizar testing
11. Testar em testing
12. Fast deploy

Sessão 23 - Criando um pipeline no portal II



Calendário do Bootcamp					
	2	3	4	5	6
9:00 - 9:30	1- Introdução ao LineA L. Nicolaci - 45'	9- Introdução ao LineA II L. Nicolaci - 45'	17- Usando o Santos Dumont V. Lima - 30'	Colaborações Científicas	37- Documentação C. Silveira - 30'
9:30 - 10:00	2- LineA: Como se registrar G. Soares - 30'	10- Trabalhando no LineA A. Nunes - 45'	18- Usando o Santos Dumont: Casos de Uso H. Camacho, A. Pieres - 30'	25- DES - B. Santiago - 25' 26- DESI - F. Oliveira - 25' 27- SDSS - J. Schimoia - 25' 28- TON - J. Camargo - 25'	38- Transferência de Dados J. Souza - 20'
10:00 - 10:30	Coffee break 15'	Coffee break 15'	Foto e Coffee break 30'		39- Máquinas Virtuais, Containers e Kubernetes J. Souza - 20'
10:30 - 11:00	3- LineA: Infraestrutura C. Adean - 45'	Coffee break 15'	19- Uso Avançado da Infraestrutura C. Adean - 45'	Coffee break 20' (início 10:40)	40- Rede Nacional de Ocultação G. Rossi - 30'
11:00 - 11:30	4- Landing Page do LineA C. Silveira - 30'	11- Ferramentas Colaborativas F. Massena - 30'		Colaborações Científicas 29- LSST - L. Nicolaci - 30'	Coffee break 20' (início 10:40)
11:30 - 12:00		12- Ferramentas Astronômicas I C. Singulani - 30'	20- Portal Científico: Introdução J. Gschwend - 45'	30- Atividades Científicas A. Pieres - 30'	41- Ciberinfraestrutura para Apoio à e-Ciência A. Moura - 30'
12:00 - 12:30	Almoço 75'	Almoço 75'	Almoço 60'	Almoço 60'	42- Atividades de EPO E. Nogueira - 30'
12:30 - 13:00					Almoço 60'
13:00 - 13:30		13- Ferramentas Astronômicas II G. Costa - 60'	21- Criando um Pipeline no Portal I C. Singulani, L. Nunes - 60'	31- Analisando DES Y6 R. Gruendl - 60'	
13:30 - 14:00	5- Spotlight talks R. Bouffleur - 90'	14- Jupyter Notebooks: Criando J. Gschwend - 30'	22- Portal Científico: Workflows Científicos A. Pieres, M. Agüena - 45'	32- DES - Primary Products A. Pieres - 45'	43- Planejando o Futuro A. Nunes, A. Pieres - 150'
14:00 - 14:30		15- Jupyter Notebooks: Usando e Desenvolvendo A. Pieres, M. Agüena - 60'	23- Criando um Pipeline no Portal II J. Gschwend - 60'	33- DES Portal - Data Preparation J. Gschwend - 45'	
14:30 - 15:00	6- Serviços F. Massena - 30'			Coffee break 30'	44- Perspectivas do LineA L. Nicolaci - 30'
15:00 - 15:30	7- Jupyter Notebooks: Introdução G. Soares - 30'	Coffee break 30'	Coffee break 30'	34- Validando Imagens DES Y6 A. Pieres - 30'	Encerramento 30'
15:30 - 16:00		16- Jupyter Notebooks: Casos de Uso F. Oliveira, R. Bouffleur - 60'	24- Requisitos para o Portal do LSST M. Sanchez - 45'	35- Validando Catálogos DES Y6 M. Agüena - 30'	
16:00 - 16:30	8- Jupyter Notebooks: Exemplos A. Pieres - 60'			36- Explorando Dados DES Y6 M. Agüena - 30'	
16:30 - 17:00					
17:00 - 17:30	Q&A 30'	Q&A 30'	Q&A 30'		

Legenda

Apresentação

Tutorial

Hacksession/Hands on

ATUALIZAR TODOS OS PIPELINES E COMPONENTS

```
/home/aguena/bin_share/gitthem
```

CRIAR UMA COMPONENT NOVA

```
/home/aguena/bin_share/make_component
```

CRIAR UM PIPELINE NOVO

```
/home/aguena/bin_share/make_pipeline
```

DAR UM TAG E PUSH NO PIPELINE

```
/home/aguena/bin_share/tagueia
```

ADICIONAR COMMITS NO CHANGELOG

```
/home/aguena/bin_share/upchange
```