Automatic builds of the ADQL Standard
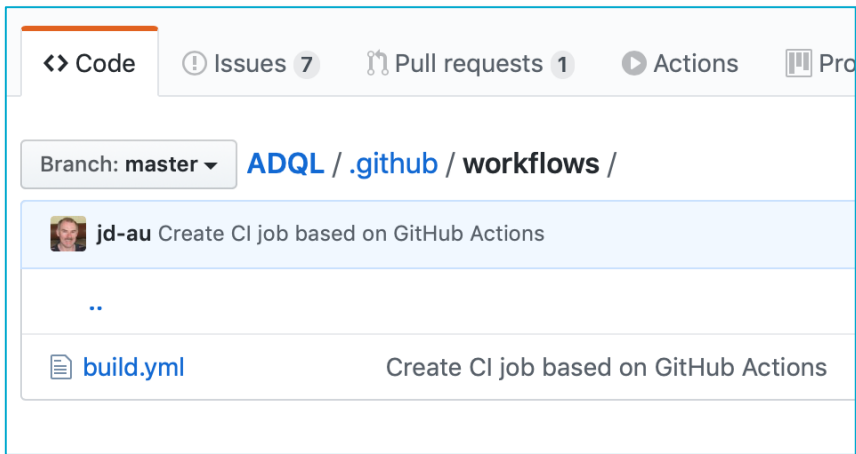
James Dempsey | 7 May 2020

Why do automatic builds?

- Check every pull request automatically
- Has the change broken the document?
- What does the result of the change look like?

- Benefits:
  - Quick feedback for the authors
  - Helps reviewers see what has changed
  - Identify bugs (tex is code) quickly!

How we set it up

- GitHub Actions
  - https://github.com/ivoa-std/ADQL/blob/master/.github/workflows/build.yml
- Build is part of the repository
- Run as part of the GitHub ecosystem
- Another good alternative is Travis

What the build does

- Steps
  1. Checkout the branch
  2. Prepare a build environment
  3. Make the document
  4. Check that the pdf has been built
  5. Publish the PDF with the PR

```
1   name: CI build of the ADQL standard
2
3   on:
4     pull_request:
5       types: [opened, synchronize, reopened]
6
7   jobs:
8     build:
9
10      runs-on: ubuntu-latest
11
12      steps:
13      - uses: actions/checkout@v1
14        with:
15          submodules: true
16      - name: Setup dependencies
17        run: |
18          sudo apt update
19          sudo apt install texlive-latex-base texlive-latex-recommended t
20
21      - name: Build the document
22        run: make biblio forcetex
23
24      - name: Check the output
25        run: |
26          test -f ADQL.pdf
27          test -f ADQL.bbl
28
29      - uses: actions/upload-artifact@v1
30        with:
31          name: PDF Preview
32          path: ADQL.pdf
```

CSIRO

Where to find the document

How does it affect the pull request

Whe

- Ever
- We'

Who else was involved

- Thanks to
  - Ole Streicher – original Travis build template
  - Pat Dowler – Trigger recommendations
  - DAL group – debugging and feedback on the implementation

Thank you

CSIRO Information Management & Technology
James Dempsey
Senior Developer, Information Services Applications

+61 2 6214 2912
James.Dempsey@csiro.au