ctools - Action #3319

Shorten ctools unit testing

08/09/2020 10:43 AM - Knödlseder Jürgen

Status:	In Progress	**Start date:**	08/09/2020
Priority:	Normal	**Due date:**	
Assigned To:	Knödlseder Jürgen	**% Done:**	20%
Category:		**Estimated time:**	0.00 hour
Target version:	2.1.0		

Description

The ctools unit test starts to last again for too long. Here a list of specific bottlenecks:

- Test cterror from Python (21 s)
- Test cterror on command line (7.2 s)
- Test ctobssim from Python (58 s)
- Test ctobssim on command line (14 s)
- Test cttsmap on command line (9.1 s)
- Test unbinned in-memory pipeline (14 s)
- Test unbinned pipeline with FITS file saving (14 s)
- Test csbkgmodel from Python (3m5s)
- Test csbkgmodel on command line (12s)
- Test csbkgmodel pickeling (12s)
- Test csiactobs from Python (9s)
- Test cslightcrv from Python (31s)
- Test cslightcrv on command line (10s)
- Test cslightcrv pickeling (9.6s)
- Test csmodelsois from Python (7.5s)
- Test csphagen from Python (11s)
- Test csphasecrv from Python (15s)
- Test csscs from Python (38s)
- Test csspec from Python (32s)
- Test csspec on command line (14s)
- Test csspec pickeling (14s)
- Test cstsdist from Python in unbinned mode (29s)
- Test cstsdist on command line (30s)
- Test cstsdist pickeling (29s)
- Test csworkflow from Python (29s)
- Test csworkflow on command line (15s)

History

#1 - 11/03/2020 05:04 PM - Knödlseder Jürgen

- Target version changed from 1.7.1 to 2.0.0

#2 - 03/15/2022 11:19 PM - Knödlseder Jürgen

- Status changed from New to In Progress

- Assigned To set to Knödlseder Jürgen

In addition, following the introduction of the carbon footprint computations the code has really become slow. This may be related to the determination of the host country, that eventually adds a few minutes before a tool terminates. Interestingly, the duration of the unit tests vary largely between systems:

centos6_64: 1 h 30 min
centos7_64: 2h 10 min
debian6_64: 23 min
fedora17_64: 1h 38 min
macosx10: 11 min
macosx11: 11 min
macosx12: 11 min

macosx13: 12 min
macosx14: 11 min
macosx15: 15 min
macosx16: 12 min
mandriva2011_64: 1h 40 min
opensolaris11_32: 36 min
opensuse12_64: 1h 25 min
sl6_64: 1h 33 min
ubuntu12_64: 20 min
ubuntu16_64: 23 min

In case that it turns out that the problem is the http query of the country code, the query could be done by the daemon, and the country code written into the $HOME/.gamma directory, so that it is sufficient to read back the country code from that directory, while the country is still updated on an hourly basis in case that the user moves around.

#3 - 03/16/2022 08:05 AM - Knödlseder Jürgen

- % Done changed from 0 to 10

After replacing host_country() in the GApplication by the string "FR" the following homogenous results were obtained. This implies that the issue is indeed the slowness of the host_country() function.

centos6_64: 26 min
centos7_64: 23 min
debian6_64: 23 min
fedora17_64: 22 min
macosx10: 14 min
macosx11: 13 min
macosx12: 14 min
macosx13: 14 min
macosx14: 13 min
macosx15: 16 min
macosx16: 13 min
mandriva2011_64: 21 min
opensolaris11_32: 33 min
opensuse12_64: 24 min
sl6_64: 25 min
ubuntu12_64: 20 min
ubuntu16_64: 23 min

#4 - 03/16/2022 11:39 AM - Knödlseder Jürgen

- % Done changed from 10 to 20

I now implemented some code change where the host country code is written in the file $HOME/.gamma/host-country, so that the host country code can be determined by simply reading back the code in the file. The daemon regularly updates the country code in the file, so if a user moves around with his laptop, the relevant country code will be used.

centos6_64: 28 min
centos7_64: 23 min
debian6_64: 27 min
fedora17_64: 25 min
macosx10: 14 min
macosx11: 13 min
macosx12: 13 min
macosx13: 14 min
macosx14: 13 min
macosx15: 17 min
macosx16: 13 min
mandriva2011_64: 23 min
opensolaris11_32: 40 min
opensuse12_64: 27 min
sl6_64: 28 min
ubuntu12_64: 23 min
ubuntu16_64: 25 min

#5 - 04/21/2022 02:27 PM - Knödlseder Jürgen

- Target version changed from 2.0.0 to 2.1.0