art - Bug #4452

Misbehavior of ParameterSet::put(...) and insert(...) functions when key exists

07/30/2013 09:24 AM - Christopher Green

Status:	Closed	**Start date:**	07/30/2013
Priority:	Normal	**Due date:**	09/30/2013
Assignee:	Christopher Green	**% Done:**	100%
Category:	Infrastructure	**Estimated time:**	4.00 hours
Target version:	1.13.00	**Spent time:**	5.00 hours
Occurs In:		**Experiment:**	Mu2e
Scope:	Internal	**SSI Package:**	fhicl-cpp

Description

If the specified key for an insert or put exists, the operation will silently fail. It is also seriously non-trivial to ascertain whether a parameter exists (ParameterSet::get_keys() followed by std::vector<std::string>::find()).

At least, insert() and put() should replace any existing value. However, I believe it is also desirable to provide a bool has_key() method to allow the user to check for presence first.

History

#1 - 07/30/2013 09:27 AM - Marc Paterno

The documentation for FHiCL, and common use, refers to the "keys" as names, so I would propose any new functions use the term 'name' rather than 'key'.

#2 - 07/30/2013 09:39 AM - Rob Kutschke

Here is my two cents...

I like the addition of a "hasName" method.

I prefer that put/insert have an optional, argument that specifies what to do if the name already exists: the two options should be to replace the existing definition or throw an exception. I prefer that the default be to throw an exception. We could also consider a third option: to allow replacement if and only if the "type" of the old and new values match; that is, we can replace a sequence with another sequence but not with a table or an atomic value.

Perhaps it also makes sense to have a "replace" method that just forwards to "put" but which sets the optional argument to allow replacement.

#3 - 02/17/2014 12:27 PM - Christopher Green

- Target version changed from 1.09.00 to 521

#4 - 09/02/2014 02:58 PM - Christopher Green

- Target version changed from 521 to 1.13.00

#5 - 02/03/2015 02:16 PM - Christopher Green

- Status changed from Accepted to Feedback

- Assignee set to Christopher Green

- % Done changed from 0 to 40

has_key() was added with fhicl-cpp:7ff98d3fdba1c9ea319eff5aa7fdd45c1eaef2cc.

We propose that insert() be taken private, because the ability to decode the encoded value relies on the correct construction of the boost::any, which cannot be left safely to the non-expert.

We further propose that put() throw if insert is not achieved, and that we provide the alternative function put_or_replace with the replace-if-existing behavior based on key only. Type checking is, unfortunately, not practical in the general case.

Does this sound reasonable?

#6 - 02/04/2015 03:58 PM - Rob Kutschke

Please go ahead with the proposal.

I don't want to give up on the limited notion of type checking that we asked for but it's important to get the agreed upon functionality out the door. Should I open a new issue for the stuff we would still like - or is it so hard that it is unrealistic to ever expect it?

#7 - 02/04/2015 04:08 PM - Christopher Green

The problem is that the only way to test whether a parameter is of a particular C++ type (or compatible with same) after encoding is to attempt the conversion and deal with the exception once thrown. It is generally not advisable to use exceptions as standard flow control as this (among other things) complicates debugging. There are several functions that aid one in distinguishing whether the key represents an atom, sequence or table, but once you have an atom determining its type requires an attempted any_cast(), because it is stored simply as a string.

#8 - 02/04/2015 04:24 PM - Rob Kutschke

Sorry. I was not clear.

I was not requesting anything to do with C++ type. I only meant FHiCL type: if a FHiCL name is already defined as an atom, I would like an option to say that redefining it as a non-atom is an error. (With the corner case that @nil is no type and can be redefined as anything). I imagine that I only get this choice once per .fcl file - is that right?

#9 - 02/04/2015 05:14 PM - Christopher Green

Ah, I understand now, sorry.

For the checking behavior you actually say you want, we would provide a function (say), put_or_replace_compatible() which would do the necessary checking and throw on an incompatible replace attempt. So, to summarize, the user-visible interface would be:

- [redacted]
  ```
  void put (std::string const & key, T const & value);
  ```
  Throw on existing key.

- [redacted]
  ```
  void put_or_replace (std::string const & key, T const & value);
  ```
  Generally succeed.

- [redacted]
  ```
  void put_or_replace_compatible (std::string const & key, T const & value);
  ```
  Throw if replacing an item with a different major FHiCL type.

Does this seem reasonable?

#10 - 02/04/2015 05:30 PM - Rob Kutschke

Yes. That's what I am looking for.

Is it also possible to make it configurable per policy so that the Mu2e executable chooses put_or_replace_compatible when the run time configuration is parsed?
I need to check with Andrei and a few others before I decide that this is what we really want. But I would like to know if it is a realistic request.

#11 - 02/05/2015 08:43 AM - Christopher Green

Choosing the behavior in code is easy (whatever function the user decides to call); allowing the coherent behavior to be configured by the application at run time is hard, and not something we've thought about before. I'm loathe to do it the quick and dirty way (a global), but anything else will require thought and design. Can I suggest that we implement the suggestions I made earlier for the three different user-facing functions, and ask that you submit a different feature request for the run time configurability you desire?

#12 - 02/05/2015 08:54 AM - Rob Kutschke

I like the proposed features - please get them in now.

As I remember the start of this discussion (maybe not captured well in the ticket) the goal was the catch a class of errors at FHiCL parsing time and for this feature to be available when the main art configuration file was parsed. But we can make this a separate ticket if needed.

#13 - 02/05/2015 10:44 AM - Christopher Green

- Status changed from Feedback to Assigned

- Experiment CDF added

- Experiment deleted (-)

#14 - 02/06/2015 10:15 AM - Christopher Green

- Status changed from Assigned to Resolved

- % Done changed from 40 to 100

- Experiment Mu2e added

- Experiment deleted (CDF)

Implemented with fhicl-cpp:1a89727 and fhicl-cpp:208dbff.

#15 - 02/16/2015 10:32 AM - Christopher Green

- Status changed from Resolved to Closed