Library Improvements

Chapel versions 1.21 / 1.22

April 9 / 16, 2020


chapel_info@cray.com


chapel-lang.org


@ChapelLanguage

CRAY

a Hewlett Packard Enterprise company

Outline

- HDF5 Parallel I/O
- Map Improvements

HDF5 Parallel I/O

HDF5 I/O: Background

- The HDF5 module had functions for…
  - parallel read/write of distributed arrays to one file per locale
  - parallel reads of distributed arrays from a single file

- But didn't yet include parallel writes of distributed arrays to a single file

HDF5 I/O: This Effort

- Added a new function to write a distributed array in parallel to a single file
- Moved the single file read/write functions to a new submodule of HDF5

```
use BlockDist, HDF5, HDF5.IOusingMPI;
var A = newBlockArr({1..100, 1..100}, int(32));
… A[i] = …
hdf5WriteDistributedArray(A, "dataFile.h5", "dataset");
```

HDF5 I/O: Impact, Status, Next Steps

CRAY
a Hewlett Packard Enterprise company

Impact:

- Parallel I/O routines supported for both reading and writing distributed arrays
- Requires using the nested 'HDF5.IOusingMPI' module

Next Steps:

- Continue to generalize
  - Support more distributed array types
- Tune for performance

Map Improvements

Map: Background

- 'map' is intended to map from keys of any type to values of any type
  - But not all flavors of classes worked as values
    - Some resulted in unhelpful error messages
- 'map' had the following iterators:
  - 'these': yields keys
  - 'values': yields values
  - 'items': yields key-value pairs
- 'map' declarations always required 'parSafe' as the first argument

```
var m: map(false, keyType, valType);
```


Map: This Effort

- Updated 'map' to support additional class flavors
  - Nilable owned classes

```
var m = new map(string, owned C?);
```

  - Non-nilable shared classes

```
var m = new map(string, shared C);
```

- Disabled ‘ref’-based accessors for maps of non-nilable classes
  - Rationale: such maps use nilable internally, so can’t return refs to non-nilable
  - Added new methods for accessing non-nilable fields

```
proc getValue(key) const
proc getAndRemove(key)
proc getBorrowed(key)
```

Map: This Effort

- Added 'keys' iterator
  - The default iterator 'these' is currently a synonym for 'keys'

```
var m: map(string,int); m.add("one", 1); m.add("two", 2);
for k in m.keys() do writeln(k);    // two, one
for k in m do writeln(k);           // also two, one
```

- Rearranged 'map' fields to allow more concise usage

```
var m: map(keyType, valType);
```


Map: Impact, Status, Next Steps

Impact: The 'map' type is generally more useful and consistent

Next Steps:

- Extend 'map' to work with additional class flavors
- Finalize definition of these() iterator
  - should it return values or (key, value) pairs (issue #14718)

For More Information

For a more complete list of library-related changes in the 1.21 and 1.22 releases, refer to the following sections of the CHANGES.md file:

- Deprecated / Removed Library Features
- Standard Library Modules
- Package Modules
- Bug Fixes

FORWARD LOOKING STATEMENTS

This presentation may contain forward-looking statements that involve risks, uncertainties and assumptions. If the risks or uncertainties ever materialize or the assumptions prove incorrect, the results of Hewlett Packard Enterprise Company and its consolidated subsidiaries ("Hewlett Packard Enterprise") may differ materially from those expressed or implied by such forward-looking statements and assumptions. All statements other than statements of historical fact are statements that could be deemed forward-looking statements, including but not limited to any statements regarding the expected benefits and costs of the transaction contemplated by this presentation; the expected timing of the completion of the transaction; the ability of HPE, its subsidiaries and Cray to complete the transaction considering the various conditions to the transaction, some of which are outside the parties' control, including those conditions related to regulatory approvals; projections of revenue, margins, expenses, net earnings, net earnings per share, cash flows, or other financial items; any statements concerning the expected development, performance, market share or competitive performance relating to products or services; any statements regarding current or future macroeconomic trends or events and the impact of those trends and events on Hewlett Packard Enterprise and its financial performance; any statements of expectation or belief; and any statements of assumptions underlying any of the foregoing. Risks, uncertainties and assumptions include the possibility that expected benefits of the transaction described in this presentation may not materialize as expected; that the transaction may not be timely completed, if at all; that, prior to the completion of the transaction, Cray's business may not perform as expected due to transaction-related uncertainty or other factors; that the parties are unable to successfully implement integration strategies; the need to address the many challenges facing Hewlett Packard Enterprise's businesses; the competitive pressures faced by Hewlett Packard Enterprise's businesses; risks associated with executing Hewlett Packard Enterprise's strategy; the impact of macroeconomic and geopolitical trends and events; the development and transition of new products and services and the enhancement of existing products and services to meet customer needs and respond to emerging technological trends; and other risks that are described in our Fiscal Year 2018 Annual Report on Form 10-K, and that are otherwise described or updated from time to time in Hewlett Packard Enterprise's other filings with the Securities and Exchange Commission, including but not limited to our subsequent Quarterly Reports on Form 10-Q. Hewlett Packard Enterprise assumes no obligation and does not intend to update these forward-looking statements.

THANK YOU

QUESTIONS?


chapel_info@cray.com


@ChapelLanguage


chapel-lang.org

cray.com


@cray_inc


linkedin.com/company/cray-inc-/