CHIUW 2023:
STATE OF THE CHAPEL PROJECT

Brad Chamberlain
June 2, 2023
WELCOME TO THE 10TH ANNUAL CHIUW WORKSHOP!

Can a single parallel language be ...
... as productive as Python?
... as fast as Fortran?
... as scalable as MPI?

Attend CHIUW on Friday and hear about how the Chapel community is working to make this vision a reality!

CHIUW: Chapel Implementers and Users Workshop
http://chapel.cray.com/CHIUW.html
Friday, May 23rd, 2014
Advance Program

The Chapel Implementers and Users Workshop, to be held in conjunction with IPDPS 2014, will be the first in what is anticipated to be an annual series of workshops designed to bring developers and users of the Chapel language (http://chapel.cray.com) together to report on work being done with the language across the broad open-source community. Attendance is open to anyone interested in Chapel, from the most seasoned Chapel user or developer to someone simply curious to learn more. On behalf of the Chapel community and CHIUW steering committee, we hope to see you at CHIUW!
WELCOME TO THE 10TH ANNUAL CHIUW WORKSHOP!

Can a single parallel language...
... as productive as Python?
... as fast as Fortran?
... as scalable as MPI?

Attend CHIUW on Friday and hear about how the Chapel community is working to make this vision a reality!

<table>
<thead>
<tr>
<th>Time</th>
<th>Session</th>
<th>Speaker(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>8:30</td>
<td>Introduction to Chapel, State of the Project</td>
<td>Brad Chamberlain (Cray Inc.)</td>
</tr>
<tr>
<td>9:00</td>
<td>Technical Talks: Application Studies</td>
<td>Claudia Fohry (Universität Kassel, Germany), Jens Breitbart (Technische Universität München, Germany)</td>
</tr>
<tr>
<td>10:00</td>
<td>Technical Talks: Compiler Optimizations</td>
<td>David E. Bernholdt, Wael R. Elwasif, Christos Kartsaklis, Seyong Lee, Tiffany M. Mintz (Oak Ridge National Laboratory)</td>
</tr>
<tr>
<td>10:30</td>
<td>Technical Talks: Compiler Optimizations (continued)</td>
<td>Aroon Sharma, Rajeev Barua (University of Maryland), Michael Ferguson (Laboratory for Telecommunication Sciences)</td>
</tr>
<tr>
<td>11:00</td>
<td>Technical Talks: Runtime Improvements</td>
<td>Chris Wailes, Jeremy G. Siek (Indiana University)</td>
</tr>
<tr>
<td>11:30</td>
<td>Technical Talks: Compiler Optimizations (continued)</td>
<td>Akihiro Hayashi, Rishi Surendran, Jisheng Zhao, Vivek Sarkar (Rice University), Michael Ferguson (Laboratory for Telecommunication Sciences)</td>
</tr>
<tr>
<td>12:00</td>
<td>Invited Talk: Walking to the Chapel</td>
<td>Robert Harrison (Stony Brook University/Brookhaven National Laboratory)</td>
</tr>
<tr>
<td>1:00</td>
<td>Invited Talk: Walking to the Chapel</td>
<td>Robert Harrison (Stony Brook University/Brookhaven National Laboratory)</td>
</tr>
<tr>
<td>1:45</td>
<td>Technical Talks: Compiler Optimizations (continued)</td>
<td>Akihiro Hayashi, Rishi Surendran, Jisheng Zhao, Vivek Sarkar (Rice University), Michael Ferguson (Laboratory for Telecommunication Sciences)</td>
</tr>
<tr>
<td>2:15</td>
<td>Technical Talks: Runtime Improvements</td>
<td>Dylan T. Stark, Brian W. Barrett (Sandia National Laboratories)</td>
</tr>
<tr>
<td>2:45</td>
<td>Technical Talks: Runtime Improvements</td>
<td>Dylan T. Stark, Brian W. Barrett (Sandia National Laboratories)</td>
</tr>
<tr>
<td>3:15</td>
<td>Technical Talks: Runtime Improvements</td>
<td>Michael Ferguson (Laboratory for Telecommunication Sciences)</td>
</tr>
<tr>
<td>3:30</td>
<td>Break</td>
<td></td>
</tr>
</tbody>
</table>
| 3:30   | Community/Panel Discussion          | Anyone who is interested is welcome to join the panel discussion.

Abstract:

MADNESS is a general-purpose numerical environment that sits upon a scalable runtime that consciously includes elements borrowed from other projects including Charm++ and the HPCS programming languages, including Chapel; it is also designed to be interoperable with "legacy" code. But as I have said to Chapel's architects several times, maintaining our own runtime is just an unpleasant transitional phase and we are looking for a permanent home — is that home Chapel? I'll give you some flavor of what MADNESS does and how, with the objective of starting a conversation and seeding collaborations.

On this day 9 years ago:

Brad Chamberlain is with Lydia Duncan.
May 23, 2014 •

still had a solid crowd at the end of a workshop on Chapel at the end of a week-long conference. Not a bad way to kick off what we intend to be an annual event.

http://chapel.cray.com/CHIUW.html
Cray first expressed its intention of developing new language(s) as part of HPCS in January 2003.

<table>
<thead>
<tr>
<th>Cray Response on Software Productivity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Brad Chamberlain</td>
</tr>
<tr>
<td>Thomas Sterling</td>
</tr>
<tr>
<td>Terry Grezczk</td>
</tr>
<tr>
<td>The Cray Cascade Project</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Language Wishlist</th>
</tr>
</thead>
<tbody>
<tr>
<td>◆ Graceful ramp for computation:</td>
</tr>
<tr>
<td>◆ Portable</td>
</tr>
<tr>
<td>◆ Performance</td>
</tr>
<tr>
<td>◆ Interoperable with other languages</td>
</tr>
<tr>
<td>◆ Allows compilation</td>
</tr>
<tr>
<td>◆ Open source</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>What else?</th>
</tr>
</thead>
<tbody>
<tr>
<td>◆ Need parallel language abstractions for:</td>
</tr>
<tr>
<td>◆ Decoupled Organization:</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Cray PL Strategy</th>
</tr>
</thead>
<tbody>
<tr>
<td>◆ Legacy Codes:</td>
</tr>
<tr>
<td>◆ New Codes:</td>
</tr>
</tbody>
</table>

Cray is interested in novel languages

- we should continue to strive for increased abstraction
- yet we’re nervous about the acceptance problem
WHAT IS CHAPEL?

**Chapel:** A modern parallel programming language
- portable & scalable
- open-source & collaborative

**Goals:**
- Support general parallel programming
- Make parallel programming at scale far more productive
HPC BENCHMARKS: CONVENTIONAL APPROACHES VS. CHAPEL

STREAM TRIAD: C + MPI + OPENMP

use BlockDist;
config const n = 1_000_000,
alpha = 0.01;
const Dom = Block.createDomain({1..n});
var A, B, C: [Dom] real;
B = 2.0;
C = 1.0;
A = B + alpha * C;

HPCC RA: MPI KERNEL

forall (_, r) in zip(Updates, RAStream()) do
T[r & indexMask].xor(r);

STREAM Performance (GB/s)

RA Performance (GUPS)
APPLICATIONS OF CHAPEL

CHAMPS: 3D Unstructured CFD
Laurendeau, Bourgault-Côté, Parenteau, Plante, et al.
École Polytechnique Montréal

Arkouda: Interactive Data Science at Massive Scale
Mike Merrill, Bill Reus, et al.
U.S. DoD

ChOp: Chapel-based Optimization
INRIA, IMEC, et al.

ChplUltra: Simulating Ultralight Dark Matter
Nikhil Padmanabhan, J. Luna Zagorac, et al.
Yale University et al.

Lattice-Symmetries: a Quantum Many-Body Toolbox
Tom Westerhout
Radboud University

Nelson Luis Dias
The Federal University of Paraná, Brazil

RapidQ: Mapping Coral Biodiversity
Rebecca Green, Helen Fox, Scott Bachman, et al.
The Coral Reef Alliance

ChapQG: Layered Quasigeostrophic CFD
Ian Grooms and Scott Bachman
University of Colorado, Boulder et al.

Much more about Applications of Chapel throughout the day
(images provided by their respective teams and used with permission)
CHAPEL ON
HPE CRAY EX / SLINGSHOT-11
ARKOUDA ARGSORT PERFORMANCE: CHIUW 2022

HPE Apollo (May 2021)
- HDR-100 Infiniband network (100 Gb/s)
- 73,728 cores / 576 nodes
- 72 TiB of 8-byte values
- ~480 GiB/s (~150 seconds)

Arkouda Argsort Performance

HPE Cray EX (April 2023)
- Slingshot-11 network (200 Gb/s)
- 114,688 cores / 896 nodes
- 28 TiB of 8-byte values
- ~1200 GiB/s (~24 seconds)

HPE Cray EX (May 2023)
- Slingshot-11 network (200 Gb/s)
- 1,048,576 cores / 8192 nodes
- 256 TiB of 8-byte values
- ~8500 GiB/s (~31 seconds)

A notable performance achievement in ~100 lines of Chapel
ARKOUDA ARGSORT PERFORMANCE: TODAY

HPE Apollo (May 2021)
- HDR-100 Infiniband network (100 Gb/s)
- 73,728 cores / 576 nodes
- 72 TiB of 8-byte values
- ~480 GiB/s (~150 seconds)

HPE Cray EX (April 2023)
- Slingshot-11 network (200 Gb/s)
- 114,688 cores / 896 nodes
- 28 TiB of 8-byte values
- ~1200 GiB/s (~24 seconds)

A notable performance achievement in ~100 lines of Chapel
ARKOUDA ARGSORT PERFORMANCE: TODAY

HPE Apollo (May 2021)
- HDR-100 Infiniband network (100 Gb/s)
- 73,728 cores / 576 nodes
- 72 TiB of 8-byte values
- ~480 GiB/s (~150 seconds)

HPE Cray EX (April 2023)
- Slingshot-11 network (200 Gb/s)
- 114,688 cores / 896 nodes
- 28 TiB of 8-byte values
- ~1200 GiB/s (~24 seconds)

HPE Cray EX (May 2023)
- Slingshot-11 network (200 Gb/s)
- 1,048,576 cores / 8192 nodes
- 256 TiB of 8-byte values
- ~8500 GiB/s (~31 seconds)

A notable performance achievement in ~100 lines of Chapel
HPE CRAY EX IMPROVEMENTS

Other New Features on HPE Cray EX:

- ability to run multiple locales per compute node
  - locale per NIC
  - locale per socket
- ability to devote a core to handling communication

What’s Next?

- Extend the above features to other platforms
  - most notably GASNet over InfiniBand
- Perform additional benchmarking studies at scale (HPCC, Bale, PRK, ...)
  - comparing to reference versions in MPI and SHMEM
**DYNO IN A NUTSHELL**

**Motivation:**
- The Chapel compiler was originally written quickly, by a small team, as a research project
- As a result, it tends to be...
  - slow
  - difficult to understand when there are errors
  - not terribly well-architected: inflexible, challenging to get started with

**This Effort:**
- Last year, we kicked off an effort to massively rearchitect it, to address these lacks:
  - better user experience in terms of speed and errors
  - easier for developers to start contributing to
  - more flexible and capable:
    - separate compilation / incremental recompilation
    - better support for tools
    - dynamic evaluation of code
    - ...
DYNO HIGHLIGHTS SINCE CHIUW 2022

- Key elements of ‘dyno’ are now used in the Chapel release:
  - **1.27.0**: its parser and AST became the default
  - **1.28.0**: ‘chpldoc’ was rewritten to use its front-end library
  - **1.29.0**: its framework for improved error messages came online
  - **‘main’**: its scope resolver is now used by default (and will be in this month’s 1.31.0 release)

- Other key elements are well underway:
  - **type/call resolver**: the major components exist in draft form
  - **AST save/restore**: a key step toward separate compilation
  - **Chapel language server**: in support of modern IDE features
COMPILING CHAPEL TO GPUS
STREAM TRIAD USING GPUS AND CPUS: CHIUW 2022 (SINGLE-LOCALE)

stream-ep.chpl

config const n = 1_000_000,
alpha = 0.01;

cobegin {
    coforall gpuid in 1..numGPUs do on here.getChild(gpuid) {
        var A, B, C: [1..n] real;
        A = B + alpha * C;
    }
    {
        var A, B, C: [1..n] real;
        A = B + alpha * C;
    }
}

This program uses all CPUs and GPUs on a single compute node.

‘cobegin { ... }’ creates a task per child statement

one task creates GPU tasks

the other runs the multi-CPU triad

This program uses all CPUs and GPUs on a single compute node.
STREAM TRIAD USING GPUS AND CPUS: TODAY (SINGLE-LOCALE)

```
stream-ep.chpl
config const n = 1_000_000,
    alpha = 0.01;

cobegin {
    coforall gpu in here.gpus do on gpu {
        var A, B, C: [1..n] real;
        A = B + alpha * C;
    }
    { 
        var A, B, C: [1..n] real;
        A = B + alpha * C;
    }
}
```

Improved syntax for GPU sublocales

This program uses all CPUs and GPUs on a single compute node
STREAM TRIAD USING GPUS AND CPUS: TODAY (MULTI-LOCALE)

stream-ep.chpl

```chpl
config const n = 1_000_000,
      alpha = 0.01;

coforall loc in Locales do on loc {
  cobegin {
    coforall gpu in here.gpus do on gpu {
      var A, B, C: [1..n] real;
      A = B + alpha * C;
    }
    {
      var A, B, C: [1..n] real;
      A = B + alpha * C;
    }
  }
}
```

Support for multi-locale GPU programs

This program uses all CPUs and GPUs across all of our compute nodes
Lots of progress in the past year:

- Mar 2022 (v1.26): first multi-GPU NVIDIA runs
- CHIUW 2022
- Jun 2022 (v1.27): first multi-locale, multi-GPU NVIDIA runs
- Dec 2023 (v1.29): first heroic AMD runs
- Mar 2023 (v1.30): first multi-GPU AMD runs
- Jun 2023 (v1.31): first multi-locale, multi-GPU AMD runs

Throughout this time, ongoing improvements in generality, features, and performance
Performance vs. reference versions has become increasingly competitive over the past 6 months.

More about GPU Computing with Chapel in Session 5 today (1:35 PDT)
LANGUAGE AND LIBRARIES
CHAPEL 2.0

Background:
- For the past several years, we have been working toward a forthcoming Chapel 2.0 release
- Intent: stop making backward-breaking changes to core language and library features

Status as of CHIUW 2022:
- Major language-related changes were considered to have largely wound down (ha!)
- Primary remaining effort was on stabilizing the standard libraries
### CHAPEL 2.0 LIBRARY STABILIZATION

**Status: Visualized**

<table>
<thead>
<tr>
<th>1.26</th>
<th>1.27</th>
<th>1.28</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Builtins</strong></td>
<td><img src="images/tick.png" alt="✓" /></td>
<td><img src="images/tick.png" alt="✓" /></td>
</tr>
<tr>
<td><strong>ChplConfig</strong></td>
<td><img src="images/tick.png" alt="✓" /></td>
<td><img src="images/tick.png" alt="✓" /></td>
</tr>
<tr>
<td><strong>List</strong></td>
<td><img src="images/tick.png" alt="✓" /></td>
<td><img src="images/tick.png" alt="✓" /></td>
</tr>
<tr>
<td><strong>Map</strong></td>
<td><img src="images/tick.png" alt="✓" /></td>
<td><img src="images/tick.png" alt="✓" /></td>
</tr>
<tr>
<td><strong>Set</strong></td>
<td><img src="images/tick.png" alt="✓" /></td>
<td><img src="images/tick.png" alt="✓" /></td>
</tr>
<tr>
<td><strong>FileSystem</strong></td>
<td><img src="images/tick.png" alt="✓" /></td>
<td><img src="images/tick.png" alt="✓" /></td>
</tr>
<tr>
<td><strong>IO</strong></td>
<td><img src="images/tick.png" alt="✓" /></td>
<td><img src="images/tick.png" alt="✓" /></td>
</tr>
<tr>
<td><strong>Path</strong></td>
<td><img src="images/tick.png" alt="✓" /></td>
<td><img src="images/tick.png" alt="✓" /></td>
</tr>
<tr>
<td><strong>Reflection</strong></td>
<td><img src="images/tick.png" alt="✓" /></td>
<td><img src="images/tick.png" alt="✓" /></td>
</tr>
<tr>
<td><strong>Types</strong></td>
<td><img src="images/tick.png" alt="✓" /></td>
<td><img src="images/tick.png" alt="✓" /></td>
</tr>
<tr>
<td><strong>BigInteger</strong></td>
<td><img src="images/tick.png" alt="✓" /></td>
<td><img src="images/tick.png" alt="✓" /></td>
</tr>
<tr>
<td><strong>Math/AutoMath</strong></td>
<td><img src="images/tick.png" alt="✓" /></td>
<td><img src="images/tick.png" alt="✓" /></td>
</tr>
<tr>
<td><strong>Random</strong></td>
<td><img src="images/tick.png" alt="✓" /></td>
<td><img src="images/tick.png" alt="✓" /></td>
</tr>
<tr>
<td><strong>Barriers</strong></td>
<td><img src="images/tick.png" alt="✓" /></td>
<td><img src="images/tick.png" alt="✓" /></td>
</tr>
<tr>
<td><strong>CTypes</strong></td>
<td><img src="images/tick.png" alt="✓" /></td>
<td><img src="images/tick.png" alt="✓" /></td>
</tr>
<tr>
<td><strong>Subprocess</strong></td>
<td><img src="images/tick.png" alt="✓" /></td>
<td><img src="images/tick.png" alt="✓" /></td>
</tr>
<tr>
<td><strong>Sys</strong></td>
<td><img src="images/tick.png" alt="✓" /></td>
<td><img src="images/tick.png" alt="✓" /></td>
</tr>
<tr>
<td><strong>SysBasic</strong></td>
<td><img src="images/tick.png" alt="✓" /></td>
<td><img src="images/tick.png" alt="✓" /></td>
</tr>
<tr>
<td><strong>SysError</strong></td>
<td><img src="images/tick.png" alt="✓" /></td>
<td><img src="images/tick.png" alt="✓" /></td>
</tr>
<tr>
<td><strong>DateTime</strong></td>
<td><img src="images/tick.png" alt="✓" /></td>
<td><img src="images/tick.png" alt="✓" /></td>
</tr>
<tr>
<td><strong>Regex</strong></td>
<td><img src="images/tick.png" alt="✓" /></td>
<td><img src="images/tick.png" alt="✓" /></td>
</tr>
<tr>
<td><strong>Time</strong></td>
<td><img src="images/tick.png" alt="✓" /></td>
<td><img src="images/tick.png" alt="✓" /></td>
</tr>
<tr>
<td><strong>Version</strong></td>
<td><img src="images/tick.png" alt="✓" /></td>
<td><img src="images/tick.png" alt="✓" /></td>
</tr>
<tr>
<td><strong>String / Bytes</strong></td>
<td><img src="images/tick.png" alt="✓" /></td>
<td><img src="images/tick.png" alt="✓" /></td>
</tr>
<tr>
<td><strong>Ranges</strong></td>
<td><img src="images/tick.png" alt="✓" /></td>
<td><img src="images/tick.png" alt="✓" /></td>
</tr>
<tr>
<td><strong>Domains</strong></td>
<td><img src="images/tick.png" alt="✓" /></td>
<td><img src="images/tick.png" alt="✓" /></td>
</tr>
<tr>
<td><strong>Arrays</strong></td>
<td><img src="images/tick.png" alt="✓" /></td>
<td><img src="images/tick.png" alt="✓" /></td>
</tr>
<tr>
<td><strong>Shared / Owned</strong></td>
<td><img src="images/tick.png" alt="✓" /></td>
<td><img src="images/tick.png" alt="✓" /></td>
</tr>
<tr>
<td><strong>Errors</strong></td>
<td><img src="images/tick.png" alt="✓" /></td>
<td><img src="images/tick.png" alt="✓" /></td>
</tr>
<tr>
<td><strong>Memory.MoveInitialization</strong></td>
<td><img src="images/tick.png" alt="✓" /></td>
<td><img src="images/tick.png" alt="✓" /></td>
</tr>
<tr>
<td><strong>Locales</strong></td>
<td><img src="images/tick.png" alt="✓" /></td>
<td><img src="images/tick.png" alt="✓" /></td>
</tr>
<tr>
<td><strong>SyncVar</strong></td>
<td><img src="images/tick.png" alt="✓" /></td>
<td><img src="images/tick.png" alt="✓" /></td>
</tr>
<tr>
<td><strong>Atomics</strong></td>
<td><img src="images/tick.png" alt="✓" /></td>
<td><img src="images/tick.png" alt="✓" /></td>
</tr>
</tbody>
</table>

- **✓** Stable
- **Progress**
- **Review Started**
# CHAPEL 2.0 LIBRARY STABILIZATION

Status: Visualized

<table>
<thead>
<tr>
<th>Builtins</th>
<th>ChplConfig</th>
<th>List</th>
<th>Map</th>
<th>Set</th>
<th>FileSystem</th>
<th>IO</th>
<th>Path</th>
<th>Reflection</th>
<th>Types</th>
<th>BigInteger</th>
<th>Math/AutoMath</th>
<th>Random</th>
<th>Collectives</th>
<th>CTYPES</th>
<th>subprocess</th>
<th>Sys</th>
<th>SysBasic</th>
<th>SysError</th>
<th>DateTime</th>
<th>Regex</th>
<th>Time</th>
<th>Version</th>
<th>String / Bytes</th>
<th>Ranges</th>
<th>Domains</th>
<th>Arrays</th>
<th>Shared / Owned</th>
<th>Errors</th>
<th>MemMove</th>
<th>Locales</th>
<th>SyncVar</th>
<th>Atomics</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>1.28</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>✔️</td>
<td>✔️</td>
<td></td>
<td></td>
<td></td>
<td>✔️</td>
<td></td>
<td>✔️</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>1.29</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>✔️</td>
<td>✔️</td>
<td></td>
<td></td>
<td></td>
<td>✔️</td>
<td></td>
<td>✔️</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>1.30</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>✔️</td>
<td>✔️</td>
<td></td>
<td></td>
<td></td>
<td>✔️</td>
<td></td>
<td>✔️</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **Stable**
- **Progress**
- **Review Started**
NEW LANGUAGE AND LIBRARY FEATURES

- **@attributes**: for embedding information in code outside the language
- **‘Communication’ module**: for single-sided puts/gets across locales
- **new first-class function syntax**: more aligned with Chapel’s procedures
- **weak class pointers**: for use with ‘shared’-based classes
- **throwing initializers**: for initializers whose post-field-init bodies may hit errors
- ...
PERFORMANCE OPTIMIZATIONS
In December, we did a soft-launch of the Chapel Language Blog: https://chapel-lang.org/blog/

**Featured Articles**

**Welcome to the Chapel blog!**
Posted on November 30, 2022

An introduction to the Chapel blog, and our intentions and plans for it.

**Advent of Code 2022: Twelve Days of Chapel**
Posted on November 30, 2022

The Chapel team’s plan for blogging during Advent of Code 2022.

**Announcing Chapel 1.30.0!**
Posted on March 23, 2023

A summary of highlights from the March 2023 release of Chapel 1.30.0

**NetCDF in Chapel, Part 1: Interfacing with the C Library**
Posted on April 26, 2023

An introduction to C interoperability in Chapel using the NetCDF library.

**Announcing Chapel 1.30.0!**
Posted on March 23, 2023

The Chapel developer community is pleased to announce the release of Chapel version 1.30.0. To obtain a copy, please refer to the Downloading Chapel page on the Chapel website.

**Highlights of Chapel 1.30.0**

**@Attributes**

Chapel 1.30.0 makes good on a longstanding intention to add a generalized attribute capability to the language. These attributes are designed to convey information to the compiler—or other tools—in a way that is integrated with the source code, extensible, and independent of keyword-based language features.

At present, a small set of attributes is supported. In particular, there are a few attributes that can be used to characterize the stability of a feature, as well as a `chipdoc` attribute for suppressing the documentation for a particular declaration. As an example, the attributes on the following procedure will generate a deprecation warning for any calls to `foo()` while also ensuring that `chipdoc` does not generate documentation for `foo()`:

```chapel
1 #deprecated("foo() is deprecated; please use 'newfoo()' instead")
2 #chipdoc:nodoc
3 proc foo()
4     writeln("In foo()");
5 }
```

Future versions of Chapel will expand upon this initial set of attributes. To learn more about the current support, refer to the Attributes in Chapel technical note.
The 6th Annual Parallel Applications Workshop, Alternatives To MPI+X

Monday, November 13, 2023
Held in conjunction with SC23

Deadline: July 24, 2023
Submission Styles: Papers / Talks
WHAT’S NEXT?

- **Continue with Quarterly Releases:** June, September, December, March
  - September’s Chapel 1.32 will be a release candidate for Chapel 2.0
- **HPE Cray EX:** More Benchmarking and Tuning
- **Dyno:** Have it take over resolution of Calls and Types
- **GPUs:** More Features and Performance
- **Blog:** Hard-Launch
- **User Support and Outreach**
- **Performance and Feature Improvements**
- ...
IN MEMORIAM

• Mike Merrill passed away on November 8th
• Mike was the chief architect and developer of Arkouda, as well as a friend to many on the Chapel project

Arkouda: NumPy-like arrays at massive scale backed by Chapel

Michael Merrill*, William Reus†, and Timothy Neumann‡
U.S. Department of Defense Washington DC, USA
Email: *mhmerrill@mac.com, †reus@post.harvard.edu, ‡timothyneumann1@gmail.com

Mike’s obituary is online at: https://www.donaldsonclarksville.com/obituary/Michael-Merrill
IN MEMORIAM

- Mike Merrill passed away on November 8th
- Mike was the chief architect and developer of Arkouda, as well as a friend to many on the Chapel project

Arkouda: NumPy-like arrays at massive scale backed by Chapel

Michael Merrill*, William Reus†, and Timothy Neumann‡
U.S. Department of Defense Washington DC, USA
Email: *mhmerrill@mac.com, †reus@post.harvard.edu, ‡timothyneumann1@gmail.com

More about Arkouda in Session 3 today (11:45 PDT)

Mike’s obituary is online at: https://www.donaldsonclarksville.com/obituary/Michael-Merrill
CHAPEL RESOURCES

Chapel homepage: [https://chapel-lang.org](https://chapel-lang.org) • (points to all other resources)

Social Media:
- Blog: [https://chapel-lang.org/blog/](https://chapel-lang.org/blog/)
- Twitter: [@ChapelLanguage](https://twitter.com/ChapelLanguage)
- Facebook: [@ChapelLanguage](https://facebook.com/ChapelLanguage)
- YouTube: [https://www.youtube.com/c/ChapelParallelProgrammingLanguage](https://www.youtube.com/c/ChapelParallelProgrammingLanguage)

Community Discussion / Support:
- Discourse: [https://chapel.discourse.group/](https://chapel-discourse.group/)
- Gitter: [https://gitter.im/chapel-lang/chapel](https://gitter.im/chapel-lang/chapel)
- Stack Overflow: [https://stackoverflow.com/questions/tagged/chapel](https://stackoverflow.com/questions/tagged/chapel)
- GitHub Issues: [https://github.com/chapel-lang/chapel/issues](https://github.com/chapel-lang/chapel/issues)
THANK YOU

https://chapel-lang.org
@ChapelLanguage