Project Skara
Status Update

Robin Westberg & Erik Duveblad
Java Platform Group
May 7, 2020
Safe harbor statement

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, timing, and pricing of any features or functionality described for Oracle’s products may change and remains at the sole discretion of Oracle Corporation.
Erik Duveblad
Principal Member of Technical Staff
Oracle
May 7, 2020
Robin Westberg
Consulting Member of Technical Staff
Oracle
May 7, 2020
Agenda

1 Overview
2 Workflows
3 Status
4 Questions?
“Make all contributors more productive, both new and existing ones.”

Project Skara Mission Statement
Overview

The goal of Project Skara is to investigate alternative SCM and code review options for the OpenJDK source code, including options based upon Git rather than Mercurial, and including options hosted by third parties. It has produced the following JEPs:

- JEP 357: Migrate from Mercurial to Git
- JEP 369: Migrate to GitHub

The project has developed various tooling to assist with this effort:

- Server-side tooling compatible with GitHub and GitLab
- Client-side tooling compatible with Git and Mercurial

Skara provides at least read-only Git mirrors of all active and consolidated lines of development.
Agenda

1. Overview
2. Workflows
3. Status
4. Questions?
Workflows
Retain Existing - Enable Additional

Skara aims to retain existing workflows, while also enabling additional ways of interaction. However, a few new concepts have been introduced:

• Every contributor will have a personal fork of an OpenJDK repository they want to contribute to
• Every change will start out as a pull request*

To keep up with and review incoming changes there are many available options:

• Mailing lists
• Web UI
• IDE integrations

(* Projects can choose to allow direct pushes to a repository, bypassing most Skara tooling)
8244339: [iworld] JIT support for inline types with abstract class supertypes #33

TobiHartmann wants to merge 2 commits into openjdk:iworld from TobiHartmann:JDK-8244339

Closed

Contribution: 7 Commits: 2 Checks: 1 Files changed: 18

TobiHartmann commented 21 hours ago • edited by openjdk

C1 and C2 do not correctly handle and optimize inline types with abstract class supertypes and corresponding array covariance. I've fixed and refactored related code. Added lots of tests and IR verification of optimizations.

Progress

- Change must not contain extraneous whitespace

Issue

- JDK-8244339: [iworld] JIT support for inline types with abstract class supertypes

Download

$ git fetch https://git.openjdk.java.net/valhalla pull/33/HEAD:pull/33
$ git checkout pull/33

8244339: [iworld] JIT support for inline types with abstract class supertypes
[Iworld] RFR: 8244339: [Iworld] JIT support for inline types with abstract class supertypes

Tobias Hartmann thartmann@openjdk.java.net
Mon May 4 14:04:09 UTC 2020

- Previous message: [Iworld] [Rev 02] RFR: 8237072: [Iworld] Add support for denoting and deriving the reference projection
- Next message: [Iworld] RFR: 8244339: [Iworld] JIT support for inline types with abstract class supertypes
- Messages sorted by: [date] [thread] [subject] [author]

C1 and C2 do not correctly handle and optimize inline types with abstract class supertypes and corresponding array covariance. I’ve fixed and refactored related code. Added lots of tests and IR verification of optimizations.

-----------

Commit messages:
- 8244339: [Iworld] JIT support for inline types with abstract class supertypes

Changes: https://git.openjdk.java.net/valhalla/pull/33/files
Webrev: https://webrevs.openjdk.java.net/valhalla/33/webrev.00
Issue: https://bugs.openjdk.java.net/browse/JDK-8244339
Stats: 1011 lines in 17 files changed: 917 ins; 15 del; 79 mod
Patch: https://git.openjdk.java.net/valhalla/pull/33.diff
Fetch: git fetch https://git.openjdk.java.net/valhalla pull/33/head:pull/33

PR: https://git.openjdk.java.net/valhalla/pull/33

-----------

More information about the valhalla-dev mailing list
<table>
<thead>
<tr>
<th></th>
<th>Modified</th>
<th>Modified</th>
<th>New File</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Changes</strong></td>
<td><strong>src/hotspot/share/c1/c1/Instruction.cpp</strong>&lt;br&gt;f58a1388: 8244339: [1world] JIT support for inline types with abstract class supertypes&lt;br&gt;15 lines changed; 0 ins; 8 del; 7 mod; 1149 unchg</td>
<td><strong>src/hotspot/share/c1/c1/LIRGenerator.cpp</strong>&lt;br&gt;f58a1388: 8244339: [1world] JIT support for inline types with abstract class supertypes&lt;br&gt;4 lines changed; 0 ins; 3 del; 1 mod; 4148 unchg</td>
<td><strong>src/hotspot/share/c1/c1/InstanceKlass.hpp</strong>&lt;br&gt;f58a1388: 8244339: [1world] JIT support for inline types with abstract class supertypes&lt;br&gt;4 lines changed; 4 ins; 0 del; 0 mod; 304 unchg</td>
</tr>
<tr>
<td><strong>Changes</strong></td>
<td><strong>src/hotspot/share/c1/c1/Klass.hpp</strong>&lt;br&gt;f58a1388: 8244339: [1world] JIT support for inline types with abstract class supertypes&lt;br&gt;1 lines changed; 0 ins; 0 del; 1 mod; 143 unchg</td>
<td><strong>src/hotspot/share/opto/graphKit.cpp</strong>&lt;br&gt;f58a1388: 8244339: [1world] JIT support for inline types with abstract class supertypes&lt;br&gt;1 lines changed; 0 ins; 0 del; 1 mod; 4639 unchg</td>
<td><strong>src/hotspot/share/opto/parser2.cpp</strong></td>
</tr>
</tbody>
</table>
7 src/hotspot/share/opto/parse2.cpp

11 src/hotspot/share/opto/type.cpp

```cpp
const TypeOpPtr& TypeOpPtr::make_from_klass_common(ciKlass* kclass, bool kclass_is_exact, NULL, Offset(0), kclass->flatten_array());
if (kclass_is_exact) {
  return TypeInstPtr::make(TypePtr::BotPTR, kclass, kclass_is_exact, NULL, Offset(0), kclass->flatten_array());
}
else if (kclass->is_obj_array_klass()) {
  // Element is an object or value array. Recursively call ourself.
  const TypeOpPtr& etype = TypeOpPtr::make_from_klass_common(kclass->as_array_klass()->element_klass(), false, true, 0);
  const TypeOpPtr& etype = TypeOpPtr::make_from_klass_common(kclass->as_array_klass()->element_klass(), /* kclass */
  bool null_free = kclass->is_loaded() && kclass->as_array_klass()->storage_properties().is_null_free();
  if (null_free) {
    assert(etype->is_valueypeptr(), "must be a valueypeptr");
    etype = etype->join_speculative(TypePtr::NOTNULL) || etype;
  }
  // [V] has a subtype: [V]. So even though V is final, [V] is not exact.
  bool xk = etype->klass_is_exact() && (etype->is_valueypeptr() || null_free);
  bool not_null_free = etype->can_be_value_type() || xk;
  bool not_null_free = etype->can_be_value_type() || not_null_free || (etype->is_valueypeptr() && etype->value_klass()->flatten);
  if (null_free) {
    assert((not_null_free && null_free), "inconsistent null-free information");
  }
  bool not_flat = (ValueArrayFlatten || not_null_free || (etype->is_valueypeptr() && exact_etype->value_klass()->flatten);
  if (null_free) {
    assert((not_null_free && null_free), "inconsistent null-free information");
  }
  bool not_flat = (ValueArrayFlatten || not_null_free || (exact_etype->is_valueypeptr() && !exact_etype->value_klass()->flatten);
  const TypeArty arr0 = TypeArty::make(etype, TypeInt::POS, false, not_flat, not_null_free);
  // We used to pass NotNull in here, asserting that the sub-arrays
  // are all not-null. This is not true in generally, as code can
```
8244339: [lworld] JIT support for inline types with abstract class supertypes (#33)

Closed

TobiHartmann wants to merge changes into openjdk:lworld from TobiHartmann:JDK-8244339

Created 21 hours ago

Reviewers +

Labels +

- integrated

TobiHartmann commented 21 hours ago

C1 and C2 do not correctly handle and optimize inline types with abstract class supertypes and corresponding array covariance. I've fixed and refactored related code. Added lots of tests and IR verification of optimizations.

Progress
- Change must not contain extraneous whitespace

Issue
- JDK-8244339: [lworld] JIT support for inline types with abstract class supertypes

Download
16

Copyright © 2020, Oracle and/or its affiliates

[May 7, 2020]
[lworld] RFR: 8244339: [lworld] JIT support for inline types with abstract class supertypes

Roland Westrelin rwestrel at redhat.com
Mon May 4 14:44:58 UTC 2020

- Previous message: [lworld] RFR: 8244339: [lworld] JIT support for inline types with abstract class supertypes
- Next message: [lworld] [Rev 01] RFR: 8244339: [lworld] JIT support for inline types with abstract class supertypes
- Messages sorted by: [ date ] [ thread ] [ subject ] [ author ]

> C1 and C2 do not correctly handle and optimize inline types with abstract class supertypes and corresponding array covariance. I've fixed and refactored related code. Added lots of tests and IR verification of optimizations.

Looks ok to me.

Roland.

- Previous message: [lworld] RFR: 8244339: [lworld] JIT support for inline types with abstract class supertypes
- Next message: [lworld] [Rev 01] RFR: 8244339: [lworld] JIT support for inline types with abstract class supertypes
- Messages sorted by: [ date ] [ thread ] [ subject ] [ author ]

More information about the valhalla-dev mailing list
Webrevs

- 01: Full - Incremental (feefc74)
- 00: Full (f58a138)

Mailing list message from Roland Westrelin on valhalla-dev:

C1 and C2 do not correctly handle and optimize inline types with abstract class supertypes and corresponding array covariance. I've fixed and refactored related code. Added lots of tests and IR verification of optimizations.

Looks ok to me.

Roland.

Thanks Roland!
C1 and C2 do not correctly handle and optimize inline types with abstract class supertypes and corresponding array covariance.
Workflows
Demonstration sources

The screenshots were taken at the following locations:

- Pull request: https://git.openjdk.java.net/valhalla/pull/33
- Mailing list: https://mail.openjdk.java.net/pipermail/valhalla-dev/2020-May/007231.html
- Webrev: https://webrevs.openjdk.java.net/valhalla/33/webrev.00
- JBS: https://bugs.openjdk.java.net/browse/JDK-8244339

Also featured: Visual Studio Code with the C/C++ and GitHub Pull Requests and Issues plugins.
Workflows
Brief Summary of Current Skara Tooling

Server side (bots)

- JBS
  - Add review links to issues
  - Add commit info / update labels (hgupdater)
- JCheck
- Mailing list integration
  - Bi-directional comment bridging
  - Notification emails
  - Webrev generation
- Automated merges
  - Conflict notifications
- CSR enforcement
- OCA checking

Client side (CLI)

Can perform necessary tasks that would otherwise require using a Web UI, such as:

- Create and sync a personal fork
  - Publish local changes
  - Retrieve changes from updates
- Create and interact with pull requests
  - List active
  - Create pull request from local change
  - Mark as reviewed / request changes
Agenda

1. Overview
2. Workflows
3. Status
4. Questions?
### Status

#### Statistics

- 70,000+ commits converted
- 5000+ e-mails sent
- 2100+ webrevs generated
- 1000+ pull requests created
- 1000+ commits integrated
- 65+ unique authors
- 2 EA builds published (Loom)
- 1 GA release (OpenJFX)

- **Largest pull request**
  - 3976 commits
  - 2.1 GB webrev
- **Skara**
  - 15 contributors
  - 98.8% Java
Status
Transitioned projects

- OpenJFX
- OpenJMC
- Mobile
- Loom
- Panama*
- Metropolis
- Valhalla
- Amber
- Tsan

- ZGC
- Lanai
- Skara
- Code Tools
  - JTHarness
  - Jcov
  - Asmtools

(* one or more development branches remain in Mercurial)
Status
What’s next?

Transition more projects
- JDK
- JDK Updates
- Code Tools
- ...

When could the JDK main-line transition? Preferably when most activity happens in a single repository
- After rampdown / feature complete
- Not immediately before an update release
Agenda

1 Overview
2 Workflows
3 Status
4 Questions?
Questions?

If a question should arise after this presentation, the following resources can be consulted:

- Wiki with FAQ: https://wiki.openjdk.java.net/display/SKARA
- Mailing list: skara-dev@openjdk.java.net

The Skara source code can be found here:

- https://git.openjdk.java.net/skara
Thank you!