GraalVM Advisory Board | November 2020

- Project updates & upcoming release (Alina Yurenko)
- GraalVM Developer survey (Alina Yurenko)
  - The survey is available here, please share;
- Security Group update (Roxana Bradescu)
  - the mailing list has been set up;
- Red Hat Feedback
  - Foivos, Red Hat: “I would personally like you to bring up how happy I (or "we" if others feel the same) am with all the help they have been providing in various issues and PRs I have opened recently. I personally feel they often go out of their way (by not just reviewing but also suggesting solutions and alternatives) to help.”
  - Stuck issues (Max, Red Hat)
    - 175 days since last update: https://github.com/oracle/graal/pull/2067 - **C2 style range check smearing**
      - Thomas: has been updated; the best is to ping the person working on it;
  - Still seeing things go into master with zero context, for example https://github.com/oracle/graal/commit/2d5eee0
    - Can GraalVM team give status on Public PR workflow ?
      - we will check;
  - Improve trackability of issues/work related to past versions
    - https://oss.oracle.com/pipermail/graalvm-dev/2020-November/000093.html
      - we discussed it internally and will follow up;
  - Project roadmap items ?
    - https://github.com/oracle/graal/projects is more worklists; any way to tag or target epics/roadmap style items ? highlevel wiki doc ?
      - to-do: share the list of more high-level that we prepared for native image;
        - Added: https://github.com/oracle/graal/issues/2762
  - Arch64 - on github action move
    - get access to an AArch64 GH runner in the graalvm-org team (this would benefit both oracle/graal and graalvm/mandrel repos when it comes to testing on Aarch64)
      - we are a bit short on ARM testing resources; could probably use help in this area;
      - Fabio: in theory we can integrate our OSS solution into travis.com;
      - Roxana: we would need an ARM runner somewhere;
      - Security concerns? Running a merged PR should be fine;

        - Thomas: we look into this on our side to get those resources;
- Project Leyden
    - Can GraalVM team give update on Project Leyden and what (if any) plans around it ? (Max, Red Hat)
        - no news at the moment;
- Sanhong Li, Alibaba Feedback
    - https://bugs.openjdk.java.net/browse/JDK-8255616, AOT/Graal have been disabled since OpenJDK16. It would be helpful to give us any guidance on the future development on AOT/Graal, how we can align these independent projects with OpenJDK?
        - Thomas: our position is the best solution for Java AOT compilation is native image;
        - regarding native image working with dynamic class loading etc - there could be a way via having Java implemented as a Truffle language; there are plans to have dynamic class loading on SVM;
    - Will each release of GraalVM do the 'full' test on AOT/Graal features after the integration with OpenJDK base? e.g run all OpenJDK related jtreg test cases?
    - How does the GraalVM evolve towards Metropolis & Leyden project? do you have any update on this front?
        - regarding project Metropolis - with the approach mentioned above we will have kind of a "Java on Java" VM; Long-term vision here is to be able run Java on GraalVM like you would run e.g. JavaScript on GraalVM.
    - (Alan Hayward) Is Graal in OpenJDK16+ still supported at all? If not, how should Graal be tested with jtreg?
        - we are working on supporting the latest version of JDK;
        - Alan: will it still be possible to integrate GraalVM compiler into the latest JDK?
            - yes, since JVMCI stays;
    - Issues follow-up
        - https://github.com/oracle/graal/pull/2730 , target next release ?
            - Thomas: we are happy about this contribution and want to contribute it soon; we are also waiting for changes to be made;
- Sébastien Deleuze: VMware feedback
    - Switch Netty to runtime init by default and sync with those are using it with build time init, see https://github.com/netty/netty/issues/10797#issuecomment-727888871
        - Netty is still using build-time init; not very consistent with GraalVM defaults; need to ask for the default change;
    - Sébastien Deleuze: Is there a global bigger footprint on Java 11 compared to Java 8 issue to tackle?
        - Max: we also see it; https://github.com/oracle/graal/issues/2129
        - would be good to collaborate on this and resolve

        - as more classes are added this also doesn't scale well;
        - there's an option that should help with that: https://github.com/oracle/graal/issues/2384 - let us know if that's still a problem
        - Analysis & compilation times could be improved too - there are plans how to make it happen; Also frameworks could help
    - Sébastien Deleuze: Looking for feedback on Project Leyden as well
- Alan Hayward: Autovectorisation
    - What are the current vectorisation plans for CE post the merge of the RangeCheckElimination patch (https://github.com/oracle/graal/pull/2153). Not sure if https://github.com/oracle/graal/issues/2703 is still in progress.
        - we have plans to add it, most likely in 21.X;
        - Thomas: would be good to coordinate with our team on this;
        - there's also an issue from Red Hat about it;
    - Any other BCE work in progress or planned?
        - Thomas: there was an issue that is closed; there's one more in progress; needs more investigation;
- Uma Srinivasan: Twitter requests/proposals
    - Co-ordinate Graal releases with Oracle & OpenJDK update releases every quarter.
        - also what kind of testing is done before the releases?
        - Thomas: we have our LabsJDK on 11; our intention is to keep it in sync with OpenJDK; we test heavily with LabsJDK. Can Mandrel help with this? No, since it doesn't use the Graal compiler.
            - We are testing GraalVM builds and making sure JVMCI works;
            - Max: we follow the OpenJDK cadence;
                - Uma: are those published? Max: the best would be to ping the team;
                - Gilles: if you want to know which versions of LabsJDK are tested, there's a file in the repo specifying that;
    - We depend on "mx" for builds and tests, any reason this repo is not branched & released like the "graal" repo?
        - mx should be compatible, so are fixing the version of `mx` used for certain builds;
    - Ran into test failures with OpenJDK 11.0.7+GraalVM version 20.1, had to backport fixes from 20.2. Would be great to understand test versions being used and a list of known test failures with the testing done associated with a release.
            - This line contains the LabsJDK version used to build the GraalVM:
                - JDK8: https://github.com/oracle/graal/blob/master/common.json#L5, the corresponding tag can be found in https://github.com/graalvm/graal-jvmci-8. Right now for example we have version "8u272+10-jvmci-20.3-b05" and the tag is jvmci-20.3-b05

        - JDK11: https://github.com/oracle/graal/blob/master/common.json#L11, the corresponding tag can be found in https://github.com/graalvm/labs-openjdk-11, Right now for example we have version “ce-11.0.9+10-jvmci-20.3-b05" and the tag is jvmci-20.3-b05
    - For the mx version used to build/test:
    - It is available in the common.hocon file but this is only set on release branches. For example on the 20.3 branch:
    - https://github.com/oracle/graal/blob/release/graal-vm/20.3/common.hocon#L17
    - (On master the version is set to “HEAD” which means just the latest version from the mx repo)