Meeting link:
https://unity3d.zoom.us/j/96545674848?pwd=REc5bTVYSGt0UTdMdXRIRFVXMTlHdz09

The meeting notes google doc is here. Feel free to correct or add additional information. The next meeting will be held on Thursday, May 19 at 2pm EST (click for your time zone).

Working group notes for September 1, 2022

Attendees: Mark, Victor

Agenda

Working group notes for September 1, 2022

Attendees: Mark, Victor

Agenda

- Unity update
    - releasing package with fork of 3.0.0-rc4 - Felix backported all relevant bug fixes (I think there was only 1)
    - the python.net API changed (after rc4? I think) and we follow semver so will need to bump our major version before switching to the (eventual) 3.0 release
    - this custom fork will land in Unity 2022.2
    - official docs! https://docs.unity3d.com/Packages/com.unity.scripting.python@6.0/manual/index.html
    - we have to meet a September deadline to land in Unity 2022.2
    - we will update to the official 3.0.0 release later in the fall

Notes

- no quorum

Working group notes for July 21, 2022

Attendees: Mark,

Agenda

-

Notes

- no quorum

Working group notes for July 7, 2022

Attendees: Mark, Victor

Agenda

-

Notes

- no quorum
- Python.NET is part of Unity 2022.1+ and becomes officially released with Unity 2022.2 (in October).

Working group notes for May 12, 2022

Attendees: Mark, Victor

Agenda

Notes

- no quorum

Working group notes for March 31, 2022

Attendees: Mark, Victor

Agenda

Notes

- (Mark) Unity on track to release our Python package with latest 3.0 RC, no problems encountered so far
-

Working group notes for March 17, 2022

Attendees: Victor, Mark

Agenda:

- 3.0 release status & API stability
- Unity status

Notes

- 3.0

        - one bug remaining - resolution agreed on?, not yet merged, could still change
            - only when running .NET from Python, doesn't affect Unity
        -

Action Items:

- Victor will talk to Benedikt about last bug

Working group notes for Feb 17, 2022

Attendees: Victor, Mark

Agenda:

-

Notes

- 3 trivial bugs remaining
- otherwise ready for release
-

Action Items:

-

Working group notes for Feb 3, 2022

Attendees: Victor, Mark

Agenda:

-

Notes

- Unity update:
    - currently moving to master / 3.9.0
        - nothing to report, "no smoke coming out"
- Felix: My only complaint is that I feel the typing is less permissive than before
    - I.E.: object needs to be explicitly iterable to iterate on or must be .ToString() to compare with another string
    - Victor: by design, but could be improved
- no quorum

Action Items:

Working group notes for January 20, 2022

Attendees: Victor, Mark (no quorum)

Agenda:

-

Notes

- Unity update:
  - current Python is 3.7, and they are migrating to 3.9
  - will take a look at https://github.com/pythonnet/pythonnet/pull/1402
  - will move to master / 3.9.0 "soon"
  - if you don't hear from us in the next 2 weeks, go ahead with the release
- no quorum

Action Items:

Working group notes for January 6, 2022

Attendees: Victor

Agenda:

- Review previous action items:
  - Benedikt will review References PR over weekend
  - If no comments Victor will merge References PR
  - Benedikt will release alpha 2 after References PR is merged
- Walk through open issues for 3.0.0 milestone

Notes

- No quorum

Action Items:

-

Working group notes for December 12, 2021

Attendees: Victor, Benedikt, Mark

Agenda:

- Review previous action items:
  - Benedikt to land PR updating version to 3.0-rc.1
- Review References PR
- Review ManagedType rework PR

Notes

- Which shutdown mode(s) does Unity use
    - "Reload" mode - (**not** soft mode)

Action Items:

- Benedikt will review References PR over weekend
- If no comments Victor will merge References PR
- Benedikt will release alpha 2 after References PR is merged

Working group notes for October 28, 2021

Attendees: Victor

Agenda:

- Review previous action items:
    - Benedikt to land PR updating version to 3.0-rc.1

Notes

- No quorum

Working group notes for October 14, 2021

Attendees: Mark, VIctor, Benedikt

Agenda:

- Review previous action items:
    - Victor will go through issues to select ones for 3.0
    - Victor will work on memory leaks
    - Benedikt will look into running 32 bit builds in CI
    - Mention we do not officially support 32 bit yet
- "We'll try to come up with a concrete roadmap for the 3.0 release in one of the next biweekly meetings."

Notes

- goal: 3.0 final by end of 2021
- Status of 3.9 support?
    - clr loader doesn't have any Python dependency
    - only remaining blocker is random CI failures - crashes on 3.9 and 3.10
        - dotnet core crashes in embedding tests
    - not a blocker for 3.0 RC
    - need to transition to safe_variant by end of 2021
- no pending PRs for 3.0, current master can be 3.0.-rc.1

Action Items:

- Benedikt to land PR updating version to 3.0-rc.1

Working group notes for September 2, 2021

Attendees: Victor, Benedikt, Mark

Agenda:
- Review previous action items:
  - Victor will continue working on PRs for breaking changes that could land for 3.0
  - Amos will continue working on porting performance improvements to master
- "We'll try to come up with a concrete roadmap for the 3.0 release in one of the next biweekly meetings."
- Python 3.9 on the mailing list
- Github Discussions feature
- https://github.com/pythonnet/pythonnet/pull/1402 blocked by issue #1412

Notes
- Release candidate for 3.0
  - next step is a preview build
  - need to groom 3.0 milestone
  - need to decide on breaking API changes for 3.0 - last chance to break API
  - python wheel build needs to be fixed
- can't run 32-bit in CI because dotnet clr doesn't support 32-bit
  - Unity does not support 32 bit
- would be nice if we can run ARM in CI
- Python 3.9 is not supported on 2.5 branch
  - if anyone wants to work on this they are welcome to do so (on 2.5 branch) but we have no plans to support on 2.5
  - we know it crashes, non-trivial to debug
  - support will come in 3.0 (master branch)
- Github Discussions
  - let's wait and see
-

Action Items:
- Victor will go through issues to select ones for 3.0
- Victor will work on memory leaks
- Benedikt will look into running 32 bit builds in CI
- Mention we do not officially support 32 bit yet

Working group notes for August 19, 2021

Quorum not met, meeting cancelled.

Working group notes for August 5, 2021

Quorum not met, meeting cancelled.

Working group notes for July 22, 2021

Quorum not met, meeting cancelled.

Working group notes for July 8, 2021

Attendees: Amos, Mark, Victor

Agenda:

- Review last week's action items:
    - Victor to make a list of breaking changes in his branch that could land for 3.0 and discuss with Benedikt
    - Victor to update https://github.com/pythonnet/pythonnet/pull/1133 and check with Benedikt that it's still good to land for 3.0
    - Amos to create a PR to master with bindings performance improvements
    - Benedikt to review https://github.com/pythonnet/pythonnet/pull/1369 Modernize import hook
-

Notes

- Victor opened PRs for breaking changes that could land for 3.0, already in review
    - will continue working on these
- #1133 __name__ and __signature__ for .NET methods will stay as is for now, not important
- #1369 Modernize import hook has been merged
- PR with bindings performance improvements has a lot of conflicts, taking longer than expected
    - Victor suggested porting changes to master one at a time rather than trying to resolve conflicts

Action Items:

- Victor will continue working on PRs for breaking changes that could land for 3.0
- Amos will continue working on porting performance improvements to master

Working group notes for June 17, 2021

Attendees: Amos, Mark, Felix, Victor

Agenda:

- Review last week's action items:

    - Victor to make a list of breaking changes in his branch that could land for 3.0 and discuss with Benedikt
    - Felix to review #1426 & #1441
    - Victor to update https://github.com/pythonnet/pythonnet/pull/1133 and check with Benedikt that it's still good to land for 3.0
- Amos concerned about performance

Notes

- #1426 & #1441 merged
- possibility of performance improvements in bindings
    - https://github.com/topameng/tolua bindings perform much better than Python for .NET
    - Victor: please add any missing coverage to performance tests so we can catch regressions
    - Amos: there is room for improvement in Python .NET performance
        - code gen to improve performance?
            - will likely introduce breaking changes
        - I have a branch already that improves performance
    - Victor: if it's not risky and can be ported to current master it could go into 3.0 (no sooner than September)
    - Victor: make a PR against master branch and describe the breaking changes
- import refactor almost done, waiting for Benedikt to review
-

Action Items:

- Victor to make a list of breaking changes in his branch that could land for 3.0 and discuss with Benedikt
- Victor to update https://github.com/pythonnet/pythonnet/pull/1133 and check with Benedikt that it's still good to land for 3.0
- Amos to create a PR to master with bindings performance improvements
- Benedikt to review https://github.com/pythonnet/pythonnet/pull/1369 Modernize import hook

Working group notes for April 22, 2021

Attendees: Victor, Mark, Felix, Benoit

Agenda:

- Review last week's action items:
    - Victor to make a list of breaking changes in his branch that could land for 3.0 and discuss with Benedikt
    - Felix to review #1426
    - Victor to update Improve Python <-> .NET exception integration
    - Victor to update https://github.com/pythonnet/pythonnet/pull/1133 and check with Benedikt that it's still good to land for 3.0
    - Benoit to test on Apple Silicon & derisk

Notes:

- Improve Python <-> .NET exception integration blocked: waiting for #1426 & #1441 to land
- Python 3.7 doesn't build on Apple Silicon
    - fixes won't be backported to 3.7
    - 3.9 is earliest version that will support Apple Silicon

Action Items:

- Victor to make a list of breaking changes in his branch that could land for 3.0 and discuss with Benedikt
- Felix to review #1426 & #1441
- Victor to update https://github.com/pythonnet/pythonnet/pull/1133 and check with Benedikt that it's still good to land for 3.0

Working group notes for April 8, 2021

Attendees: Victor, Mark, Felix, Benoit

Agenda:

- Review last week's action items:
    - everyone should look into the finalizer issues if they can
- PythonException finalizer update

Notes:

- finalizer issues?
    - not sure what this is referencing, Victor will talk to Benedikt
- PythonException finalizer update blocked due to failure on 3.6
    - Benedikt gave Felix pointers he needed to get unblocked
    - Victor posted a pull that potentially fixes - this PR #1426
- Victor has a bunch of breaking changes he wants to potentially land for 3.0
    - feature: ability to add mix-ins to .NET types with Python
        - solves multiple issues with dictionary conversions
        - avoids having to create a proxy class in C# that doesn't behave like a C# dictionary
        - allows you to create a class that works as a dictionary in both Python and C#
        - fairly mature, possibly can land before 3.0
    - feature: change overload resolution to whatever dynamic does in C#
        - solves problem where overloads behave in a surprising way
        - barely a proof-of-concept (not even tested/validated/etc)
    - feature: ability to change base class
    - others
- Victor has many open PRs
    - https://github.com/pythonnet/pythonnet/pulls/lostmsu
    - please review

- Have we tested on Apple Silicon
    - no
    - Benoit can test
    - Travis? do we have automated testing on Mac? we do, AMD64 only
    - we don't have a good debugger for 64-bit ARM
    - Unity will need Apple Silicon support by November to make our Python package officially supported
    - potentially lots of work
- Native code page still there?
    - no, got removed

Action Items:

- Victor to make a list of breaking changes in his branch that could land for 3.0 and discuss with Benedikt
- Felix to review #1426
- Victor to update Improve Python <-> .NET exception integration
- Victor to update https://github.com/pythonnet/pythonnet/pull/1133 and check with Benedikt that it's still good to land for 3.0
- Benoit to test on Apple Silicon & derisk

Working group notes for March 25, 2021

Attendees: Benedikt, Mark

Agenda:

- Review last week's action items:
    - everyone should look into the finalizer issues if they can
- PythonException finalizer update

Notes:

- No quorum, meeting canceled

Action Items:

- Mark will check if dev work on existing Unity PRs is still in progress (if not, please comment on the PR when we expect work to resume)

Working group notes for March 11, 2021

Attendees: Felix, Benoit, Mark

Agenda:

- Review last week's action items:
    - Amos will check if _GC_Calloc can be replaced with public function
    - Felix will continue working on PythonException finalizer

Notes:

- Benedikt agrees it's a good idea to get rid of _GC_Calloc, need to be mindful that allocation works with domain reload

- Technical discussion on debugging
- Current state of tests is that we have pytest and NUnit tests
    - can run pytest through NUnit but not the inverse
    - should we keep both?
- domain reload tests are separate (Re-enable the domain reload tests)
    - could trip up devs if they don't know to run the reload tests
    - same situation right now with pytests & NUnit tests - need to run both locally
    - document? make available via a single front-end?
    - ideally running either pytest or NUnit tests runs all interesting tests
    - most code is in C#, maybe dotnet test should run all tests, not pytest
    - still nice to run pytest, allows us to write pytests in a meaningful way
    - pytests are a bit hacky as they need to build the Python tests DLL
    - ideal to run all tests through dotnet test - big project though
    - create a bridge to run pytests via NUnit?
    - Benedikt needs to think more on how to solve this
    - current setup is fine for now (keep as separate steps)
- Modernize import hook
    - import hook as python code has implications
    - can we embed it instead of having it as a .py file?
- Benedikt has no major changes planned
    - going through PRs to get pythonnet into a stable state
    - dealing with finalizer issues
    - once main performance issues, finalizer fixes & import hook is merged we'll be ready for 3.0
    - the work Victor has started on reworking older components is good
    - look at obsoleting unsafe interfaces before major release
- What does Unity need?
    - import hook is the last thing
    - we'd like to ship with an official 3.0 (currently shipping a git hash build)
    - timeline for that is fall 2021
-

Action items:

- everyone should look into the finalizer issues if they can

Working group notes for February 25, 2021

Attendees: Victor, Amos

Agenda:

- Review last week's action items:
    - Felix will create PR to likely fix Domain reload crashes when .NET object points to PythonException
    - Benedikt will look into automatically setting PYTHON_HOME and/or inferring location of Python DLL

    - Benedikt will comment on the CLR Loader regarding Finalize call from Python on process shutdown and merge the PR

Notes:

- Domain reload crashes, Felix is still working on it. Two steps: ClassObject for Exception finalizer (being tested), PythonException serialization (TBD)
- We still have other private Python functions used. Victor thinks we should get rid of them (related to _GC_Calloc)

Action items:

- Amos will check if _GC_Calloc can be replaced with public function
- Felix will continue working on PythonException finalizer

Working group notes for February 11, 2021

Attendees: Victor, Benedikt, Mark, Felix, Amos

Agenda:

- Review last week's action items:
    - Victor: fix merge conflict and merge PythonDLL PR
    - Victor: create issue for domain reload when .NET objects point to PythonException and/or PyObject
    - Benedikt will check if he can restore perf tests using wheel of 2.5.1

Notes:

- PythonDLL PR was merged
- domain reload when .NET objects point to PythonException and/or PyObject
    - Felix found a problem, will create PR to fix
    - PyValue wasn't reference counted
- PythonDLL work
    - Python usually requires setting PYTHON_HOME - maybe we should try to discover DLL location from PYTHON_HOME instead of starting from the DLL location?
    - PYTHON_HOME doesn't work very well for system environment
    - maybe should try to discover PYTHON_HOME from DLL path instead?
    - need to have both
    - Benedikt will take a look at different OSes and see what could work
- CLR Loader PR
    - The failures are due to Finalize call from Python on process shutdown, which previously was not there
    - Calling Finalize is correct, but exposes preexisting bugs that now cause CI to fail
    - can we ignore the CI failures and fix them as a separate issue?
        - yes
    - Benedikt will move code fixing reference counting (and other changes unrelated to Finalize) and enabling Finalize to a new branch and land CLR loader

    - we'll treat the surfaced Finalize issues separately
    - Felix will need to rebase his PR
    - CI will continue to non-deterministically fail in the interim
- weakref support PR
    - need to fix some bugs to pass CI

Action items:

- Felix will create PR to likely fix Domain reload crashes when .NET object points to PythonException
- Benedikt will look into automatically setting PYTHON_HOME and/or inferring location of Python DLL
- Benedikt will comment on the CLR Loader regarding Finalize call from Python on process shutdown and merge the PR

Working group notes for January 28, 2021

Attendees: Benedikt, Victor, Amos

Agenda:

- Benedikt will fix 32-bit issues
- Felix will complete rewrite of import hook to avoid calling C# unless necessary
    - In review now
- Victor will find out if there will be a Roslyn release fixing the internal compiler error
    - Roslyn has published preview packages, so PythonDLL PR is ready to merge

Notes:

- Victor: might want to replace binding (overload resolution) with one from DLR before 3.0
- We should do some release planning. We can discuss next week with more folks present. Benedikt wants to release soon
- Amos will resume weakref work in Feb

Action items:

- Victor: fix merge conflict and merge PythonDLL PR
- Victor: create issue for domain reload when .NET objects point to PythonException and/or PyObject
- Benedikt will check if he can restore perf tests using wheel of 2.5.1

Working group notes for January 14, 2021

Attendees: Benedikt, Mark, Victor, Benoit, Felix, Amos

Agenda

- Review last week's action items:
  - Benedikt will fix 32-bit issues
  - Benedikt to add test that loads unicode function names
  - Felix will submit PR for #1333 (function delegates)
  - Felix will submit PR that rewrites import hook to use pure Python (see #1096 & #727)

Notes:

- 32-bit issues
  - nothing done yet, Benedikt will look into this
- test that loads unicode function names
  - merged! PR #1329
  - problem was that we need to make sure we always use our own UTF-8 marshaller
    - we can't use the .NET one because it doesn't exist in all .NET frameworks
    - PR fixes the issue where Benedikt saw it, but there are possibly other places where the same issue exists
  - follow-up issue is #1354
    - let's link all encoding problems to this issue
  - in Python 3 it's all UTF-8
  - this particular issue (unicode function names) is fixed
- function delegates (PR #1333)
  - merged!
- rewrite of import hook to use pure Python (see #1096 & #727)
  - in progress (Felix)
  - rewrite avoids entering C# unless we absolutely need to, because there's no way to prevent C# from "owning" a thread that calls C#, and once C# owns it domain reload hangs waiting for the thread to exit
  - fixes a common pattern "try import PySide; except: import PySide2" - previous implementation hits C# on the first import call, even if PySide doesn't exist
- internal compiler error (https://github.com/dotnet/roslyn/issues/49760)
  - still present in latest .NET 5.0.2
  - might still be a Roslyn point release?
  - doesn't help because Roslyn likely frozen until .NET 6
  - Victor will find out when they plan on releasing a fix
  - required for replacing PInvoke delegates for reference types with using unmanaged function pointers (supposed to work in new C#, but triggers an internal compiler error)
  - blocking our ability to do a multi-stage startup and dynamically bind to the right Python dll/dso/dylib anywhere but Mono (which supports __Internal as the DllImport string)

    - helpful for embedding .NET in Python since want to use the statically linked symbols in the python executable - using a dynamic library, even if present, leads to weird problems
    - we'll find out if a fix is forthcoming and if not, use a workaround

Action items:

- Benedikt will fix 32-bit issues
- Felix will complete rewrite of import hook to avoid calling C# unless necessary
- Victor will find out if there will be a Roslyn release fixing the internal compiler error

Working group notes for December 17, 2020

Attendees: Felix, Mark, Amos, Benedikt, Benoit, Victor

Agenda

- Review last week's action items:
    - Benedikt will fix 32-bit issues
    - Victor will sync with Benedikt about minimal version
    - Felix will submit bug for recent fix to keyword argument functions, working on minimal repro

Notes:

- 32-bit issues should be easier now that proper project files have landed
- another odd bug reported (not reproduced) around unicode decode errors when importing one of the .NET modules
    - Benedikt working on reproducing locally
    - have they used the correct installation instructions?
    - really looks like utf-8 vs. another encoding conflict in function names
    - this particular case (using angstrom character in function name) works on master, something weird going on
- minimal version
    - we know .NET 4.7.2 has solid .NET Standard support - we don't support 4.6.x (though nothing is stopping anyone from using it)
    - we should adjust our issue template (already done)
    - clarification: our actual runtime requirement is .NET Standard 2.0, but if you're using .NET (not mono) you need 4.7.2
    - only relevant if you're using Python from inside a .NET runtime
- keyword argument functions bug is fixed
    - still two issues - test cases commented out
    - someone is working on adjusting Amos's operator overload support (also touches this code)
    - actual issue is fixed though
- CLR loader
    - scenario: I'm running Python and I want to import clr
        - spins up .NET runtime and then tries to import Python .NET dll
        - need ability to change Python dll name (e.g. libpython.so.3.7 on linux, python37.dll on windows)

        - mono has a nice hack where you can use __INTERNAL and then it just looks for symbols in the global space - can just use os library loader to load the dll
        - the hack looks for PInvoke attributes and adjusts the reference using Mono.Cecil without loading into .NET
        - works in (at least) mono on Linux
        - .NET core has no equivalent mechanism
        - modern Python statically links, so doesn't reference libpython - if you try to load libpython from Python.exe, you get conflicts (all symbols you import need to come from the same place)
        - Unity uses the __INTERNAL hack, caveat is that you need to use it before loading any symbols
        - can't do this lazily
    - Benedikt working on integration (PR open)
    - Victor has a better solution than the Cecil hack
        - requires C# 9 compiler feature (function pointers)
        - waiting for bug fix
    - still working on delayed reference resolving
        - problem is that Python dll is already loaded, and once loaded you can't unload it
        - app domains might be a solution?
- Felix discovered (#1333) function to get a delegate from a function pointer
    - doesn't permit pointing to a delegate
    - mono 6.12 problem
    - solution is to store delegates instead of generating them
    - will submit PR
- Mono assumes ownership of Python threads that have called into C#
    - problem because import hook calls C#
    - results in hang on threads on domain unload
    - there's an open issue (#1096 and #727) around replacing the call into C# in the import hook with a pure python implementation
    - there's still a problem if you have a thread that explicitly calls into C#, domain unload hangs
    - researching if possible to tell mono to let go of the thread
    - not sure if there's a solution with .NET
        - .NET 5 drops app domains anyway
        - if people want to use domain reload they should be using Mono
- Felix considering removing unit tests that will be covered by new work
    - fine, just make sure the reason is explained in a comment
- investigated poetry, but it's not ready
    - sticking with setuptools for now
- Amos has work that improves UCS encoding/decoding for 4.7.2
- CI
    - currently doesn't test Mono on Windows
        - if Unity wants to add this permutation, they are welcome to figure out how to convince .NET test to use Mono

    - the new CI with github actions is fast!
    - CI enhancements coming
    - unfortunately github actions it doesn't recognize duplicate warnings across build types - noisy
    -
- we will skip the next meeting (Dec 31st) so next working group meeting will be January 14th
-

Action items:

- Benedikt will fix 32-bit issues
- Benedikt to add test that loads unicode function names
- Felix will submit PR for #1333 (function delegates)
- Felix will submit PR that rewrites import hook to use pure Python (see #1096 & #727)
-

Working group notes for December 3, 2020

Attendees:

Agenda

- Review last week's action items:
    - Felix will reduce the test combinations for soft shutdown on appveyor and travis to reduce build times and make a PR
    - Felix will look at extracting domain reload tests to reduce/simplify build times and report back
    - Benedikt will fix 32-bit issues

Notes:

- Felix still working on soft shutdown PR
    - Travis build still failing due to missing assembly, can't reproduce locally
    - TestRunner executable is not building
    - Felix will flag the issues in the PR
- Viktor worked on removing the conditional compilation and Python version fixed dll name
- Benedikt is working on dropping everything except .NET standard
- Unity will use 2.x versions for temp releases of Python for .NET to avoid conflicting with upcoming releases
- 3.x won't happen earlier than summer 2021
- Unity found two regressions
    - UCS-related missing method, could be on Unity's side
    - bug related to recent fixes to keyword argument functions

Action items:

- Benedikt will fix 32-bit issues
- Victor will sync with Benedikt about minimal version

- Felix will submit bug for recent fix to keyword argument functions, working on minimal repro

Working group notes for November 19, 2020

Attendees: Victor, Benedikt, Mark, Amos, Benoit, Felix

Agenda

- Review last week's action items:
  - Felix & Benoit to add tests to master in preparation for the soft reload fix PR - will be ignored for now (or expected to fail - ideal if that's possible)
  - Victor to review these to understand the need for isolated testing environment for each test
- discuss soft init/reload/shutdown modes: what are the use cases for each? are they all necessary? need to document if so. (Benoit/Felix)
-

Notes:

- PR for soft reload tests is failing, hasn't run on travis
  - coverage reported is not accurate
  - let's not worry too much about the coverage numbers and try to provide good tests for new features
  -
- build times have gotten slower over time due to testing all shutdown modes
  - do we need to run full tests for all modes?
  - some of this is likely redundant
  - let's remove unnecessary tests from travis
    - additional run for every mode
  - github actions?
  - we should keep at least one version of python testing all shutdown modes
  - drop everything except 3.7 from testing shutdown since that's the version we support in Unity?
  - still want to catch regressions caused by new Python versions (e.g. API changes)
  - long term we should look at moving from travis to github actions
- Unity shutdown/domain reload is using reload mode. do we need other two modes? (soft / none)
  - we never use reloading when embedding C# in Python - just kill workers/instances and restart
  - can we remove soft & none modes?
  - what are the use cases for the other two modes?
  - reload mode requires implementing a customer serializer
    - for CLR objects

        - Feilx is working on making the custom serializer optional so it still works without it
        - with this update reload won't fail without a serializer
        - does this make soft mode redundant?
    - there are a lot of places where there are three code paths depending on the shutdown mode - can we simplify
    - soft mode has no such requirement
    - none - used for shutting down Python and not restarting it - makes no attempt to save state
    - soft - does not maintain connections between CLR and Python objects, but saves state of Python interpreter
    - reload - also maintains connections between CLR and Python objects
    - all three modes are useful as they facilitate testing different parts of the system
        - e.g. if something fails on only one of the modes, it's useful data
    - currently all tests are run with all modes in all versions
- Travis problems
    - jobs take forever to start
    - job status not reporting
    - at least partly connected to new free tier limitations
    - https://blog.travis-ci.com/2020-11-02-travis-ci-new-billing
    - we will need to move things off travis-ci.org by the end of the year
    - travis is shutting down free travis access due to abuse
    - move to travis-ci.com, still potential for free use but will be potentially limited
    - longer term move to github actions needs to be evaluated
- Benedikt's SDK-style build system branch working well, except for some unexpected crashes
    - using poetry instead of setuptools - much simpler than current
    - PRs to come

Action items:

- Felix will reduce the test combinations for soft shutdown on appveyor and travis to reduce build times and make a PR
- Felix will look at extracting domain reload tests to reduce/simplify build times and report back
- Benedikt will fix 32-bit issues

Working group notes for November 5, 2020

Attendees: Mark, Victor, Felix, Benoit, Amos

Agenda

- Review last week's action items
    - Mohamed - try PR with latest master
    - Benoit will summarize what soft reload should reload (and how?)
        - Discussion here: https://github.com/pythonnet/pythonnet/issues/1268
    -

Notes:

- Soft reload
    - Felix & Benoit working on figuring out which of the three options (see issue #1268) is the best approach
        - all three seem to be equally complex
        - still working on derisking
    - what are the requirements?
        - if I import a module in Python that is actually on the C# side, after domain reload I shouldn't have to explicitly reload it
        - if it's not a module, the user is responsible for handling refresh - we guarantee not to crash, but you will get exceptions
        - tests
            - how do we handle tests that currently fail, but will pass once the above is fixed?
            - include as disabled? expected to fail?
            - these tests need to run in their own environment because initialization of Python.NET and shutdown is part of the test - each test needs to run in an isolated environment
            -
            -
- Victor reviewing Amos' PR#1267 for fixing CLR handle location
- When will version 3 come out?
    - not before 2021 - March or even April
    - Unity will release (in beta) our package (including Python.NET) before 3 comes out, so we all will need to be careful to call out any breaking changes in the Python.NET changelog
- discussion of Intern string PR
-

Action items:

- Felix & Benoit to add tests to master in preparation for the soft reload fix PR - will be ignored for now (or expected to fail - ideal if that's possible)
- Victor to review these to understand the need for isolated testing environment for each test

Working group notes for October 22, 2020

Attendees:Victor, Amos, Benoit, MarkV, Mohamed, Felix

Agenda

- **Review Action Items**
    - Amos:
        - Set test (in TestFinalizer) to ignore.
        - Create a ticket
        - Push the updated soft-reload branch (today!)
    - Then Victor or Benedikt can merge the Amos branch. Do not squash

    - Victor: take a look at Mohamed's PR issue with Exception (next time if Mohamed is there)
    - Benoit will create issue for delegates/assemblies/classes
        - https://github.com/pythonnet/pythonnet/issues/1250
    - Benoit to prototype solution for deserialization, report on it next time

Notes:

- Amos branch
    - PR merged, failing test (unrelated to soft reload) disabled for now
- Benoit - deserialization prototype in progress. When (de-)serialization fails, reload stops (fatal failure). Custom serializer, trying to serialize field by field. Issue preserving object referential identity. Suggestion so far is to throw in Python if the CLR object is missing after reload.
- Amos suggests to check the list of types available after reloading before trying to deserialize.
- Unity reload: compiler recompiles, then Unity initiates shutdown, so at shutdown time it is already known which types are gone (Precompile the libraries to a temprary folder, then use Mono.Ceil to read the metadatas for checking which methods, fields, properties would be gone). Amos suggests using this to warn user.
- Victor suggests exposing an interface from Python.NET, that would shift the concern about serializing internal objects to the consumer of the library. Benoit: this would need to produce internal Python.NET objects one by one rather than the entire list.
- Mohamed is checking how soft-reload affects planned PyType_FromSpec changes
- Amos, Victor: GitHub is a better communication media for internal stuff

Action items:

- Mohamed - try PR with latest master
- Benoit will summarize what soft reload should reload (and how?)
    - Discussion here: https://github.com/pythonnet/pythonnet/issues/1268

Working group notes for October 8, 2020

Attendees: Benedikt, Victor, Amos, Benoit, MarkV

Agenda

- **Review Action Items**
    - Felix to implement a couple of comments by Victor
        - Then Victor to merge to master
    - Victor: take a look at Mohamed's PR issue with Exception (next time if Mohamed is there) #1219
    - Benedikt: will write ticket for WITH_DEBUG, let someone else implement it.
    - Felix to create github issue to track the delegate/assembly/class/member so we can discuss technical solutions on it. Include test cases to demonstrate each specific issue.

Notes:

- Amos branch:
    - Amos merged master.
    - One new test case fails.
    - Unrelated to soft reload.
- Mohamed PR issue: punt
- Benedikt: created #1238
- Felix: did not create, Benoit and Felix talked at length
    - ClassObject serializes members of Type
    - CLRObject serializes object
    - Potential solution: custom serializer for those references

Action items:

- Amos:
    - Set test (in TestFinalizer) to ignore.
    - Create a ticket
    - Push the updated soft-reload branch (today!)
- Then Victor or Benedikt can merge the Amos branch. Do not squash
- Victor: take a look at Mohamed's PR issue with Exception (next time if Mohamed is there)
- Benoit will create issue for delegates/assemblies/classes
- Benoit to prototype solution for deserialization, report on it next time

Working group notes for September 24, 2020

Attendees: Benedikt, Victor, Amos, Felix, Benoit

Agenda

- **Review Action Items**
    - Felix: move some changes from PR to upstream; CI fails - needs fixes
    - Victor: take a look at Mohamed's PR issue with Exception
    - Benedikt to confirm WITH_DEBUG builds need to be updated for 3.9 (low priority) and will create issue to document for later
- Soft shutdown: next steps
    - What's left to merge
    - Some further fixes for removing assemblies, classes, members; dealing with delegates

Notes

- Benedikt: Python 3.8 is indeed broken with WITH_DEBUG. Memory layout has changed. Shouldn't be complicated. Benedikt will write a ticket.
- Felix: seems to be one last issue with borrowed reference GetDangerousAddress.
    - Amos: I think merge to master is good. Remaining issues are trivial.
    - Victor: minor comments left. Then I can merge. But the build is failing. (benedikt, felix: oh that's a trivial issue, should be fixed momentarily)
- Victor: wait on Mohamed's PR (issue with Exception) until Amos branch is merged

    - Benedikt: Might be the same issue as WITH_DEBUG? (actually no)
    - Victor thinks probably not. Seems to be related to the space for the gchandle to put into the type handle.
    - Amos: we're relying on the cpython code which isn't using the fields we're using for pyobject (not documented). There's a PR which doesn't rely on that: https://github.com/amos402/pythonnet/pull/5
- Amos branch has some conflicts, seems to be about dropping py-3.4/3.5
- Ensure CI testing of reload:
    - Net standard: can't run tests but we can reload. Conclusion: rely on user reports if there's a problem.
    - Net core: can't run tests, can't reload (there's no such things as domains)
    - Mono: are we testing? (seems like yes)
- Reload in the face of delegates, and in the face of assemblies, classes, members that are deleted.

Action items

- Felix to implement a couple of comments by Victor
    - Then Victor to merge to master
- Victor: take a look at Mohamed's PR issue with Exception (next time if Mohamed is there)
- Benedikt: will write ticket for WITH_DEBUG, let someone else implement it.
- Felix to create github issue to track the delegate/assembly/class/member so we can discuss technical solutions on it. Include test cases to demonstrate each specific issue.

Working group notes for September 10, 2020

Attendees: Felix, Mohamed, Victor, Amos

Agenda

- Review action items from last meeting:
    - Benedikt to confirm WITH_DEBUG builds need to be updated for 3.9 (low priority) and will create issue to document for later
    - Victor will check out PyType_FromSpec branch on the issue with Python types deriving from System.Exception #1196
- Soft shutdown

Notes

- Codecs: performance discussion requires tests
- Soft Shutdown: main comments are pending Amos review
    - Type serialization: fixed if a base type not serializable it will crash
    - Refactoring is not related (Victor is ok with it - cleans code a bit; would be better a separate PR)
    - PR CI is currently failing

Action Items

- Felix: move some changes from PR to upstream; CI fails - needs fixes
- Victor: take a look at Mohamed's PR issue with Exception

Working group notes for August 27, 2020

Attendees: Felix, Mohamed, Victor, Amos, Benedikt

Agenda

- Benedikt to confirm WITH_DEBUG builds need to be updated for 3.9 (low priority) and will create issue to document for later
- Benedikt - Add reference to backports-2.5 branch in README
- Benedikt - Switch to .NET Standard for everything but ClrModule
- Benedikt - will look at Amos's fix for #891 issue in PR #958
- Benedikt - will look into Python Buffer branch again
- Benedikt to look into weirdness with pycparser in appveyor.yml (see PR #1175)
- Felix to create test case for Amos that demonstrates the "Python function as delegate fails on reload" (and others) and log issue - will create unit tests to land after main PR lands
- Benoit/Felix to create doc outline for reload & soft shutdown modes (developer docs)
- Felix will fix remaining domain reload issues raised by Victor
- Amos will fix new issues to land after main PR lands

Notes

- Reference to backports-2.5 added
- .NET Standard/SDK Style in work in https://github.com/pythonnet/pythonnet/pull/1209
- Work on Soft Shutdown ongoing
- Mohamed stuck on PyType_FromSpec because of our handling of exception types (https://github.com/pythonnet/pythonnet/pull/1196#issuecomment-670956945)
- Soft shutdown uses pycparser too

Action Items

- Benedikt to confirm WITH_DEBUG builds need to be updated for 3.9 (low priority) and will create issue to document for later
- Victor will check out PyType_FromSpec branch on the issue with Python types deriving from System.Exception #1196

Working group notes for August 13, 2020

Attendees: Mark, Benoit, Felix, Mohamed, Victor, Amos

Agenda

- Review action items from last meeting:
  - Benedikt to confirm WITH_DEBUG builds need to be updated for 3.9 (low priority) and will create issue to document for later
  - Benedikt - Add reference to backports-2.5 branch in README

    - Benedikt - Switch to .NET Standard for everything but ClrModule
    - Benedikt - will look at Amos's fix for #891 issue in PR #958
    - Benedikt - will look into Python Buffer branch again
    - Benedikt to look into weirdness with pycparser in appveyor.yml (see PR #1175)
- Domain reload update

Notes

- Felix has been testing domain reload branch
    - mostly success with a few fixes needed
    - issues
        - if you register a Python function as a delegate directly into a C# event, it is not retained - currently throws an exception due to leaving the Python function in a bad state
            - would be ideal to bring back the connection
            - Acceptable to lose the connection, but must leave python.net in a good state otherwise
        - if you maintain a proxy to a C# object in Python, on domain reload the proxy usually gives several different exceptions and sometimes crashes
            - correct behaviour is to raise an exception, ideally the same one with no crashes
        - if an assembly/type/class member/etc is missing after domain reload, causes problems on the Python side
            - needs to be gracefully handled
        - finalizers at exit fix done by Felix
            - PR not yet made
    - other than these issues reload mode works really well
    - soft shutdown mode doesn't work well for domain reload
        - it unloads all modules, which doesn't work for native modules (by design)
        - intended for use in the game, not the editor
    - Victor approved landing it to master once remaining issues raised in PR are fixed
        - Felix/Benoit to fix the minor issues in the PR and land the patch
    - If it can't land to master, Victor will create a branch for it in the repo
    - Remaining issues found by Felix can land to master after the main PR lands to master. Process:
        - Fixes: Felix submits a PR for the fix (ideally with test case)
        - Issues: create an issue, Felix submits PR with test case, invites Amos to fix. Amos fixes it. Land PR with test case + fix to master.
    - Documentation would be good
        - Proposal:
            - Unity/imaginary spaces writes an outline,
            - amos fills in outline,
            - Unity/imaginary spaces revises it

    - Python 2 support for soft shutdown / reload is not necessary.

Action Items

- Benedikt to confirm WITH_DEBUG builds need to be updated for 3.9 (low priority) and will create issue to document for later
- Benedikt - Add reference to backports-2.5 branch in README
- Benedikt - Switch to .NET Standard for everything but ClrModule
- Benedikt - will look at Amos's fix for #891 issue in PR #958
- Benedikt - will look into Python Buffer branch again
- Benedikt to look into weirdness with pycparser in appveyor.yml (see PR #1175)
- Felix to create test case for Amos that demonstrates the "Python function as delegate fails on reload" (and others) and log issue - will create unit tests to land after main PR lands
- Benoit/Felix to create doc outline for reload & soft shutdown modes (developer docs)
- Felix will fix remaining domain reload issues raised by Victor
- Amos will fix new issues to land after main PR lands

Working group notes for July 30, 2020

Attendees: Amos, Victor, Mark, Benedikt, Mohamed

Agenda

- Review action items from last meeting
    - Benedikt - Check whether we need to change the `WITH_DEBUG` variant of PyObject_TYPE for newer versions of Python (> 3.8?)
    - Benedikt - Add reference to backports-2.5 branch in README
    - Benedikt - Switch to .NET Standard for everything but ClrModule
    - Benedikt - Change project files to new format
    - Victor - Consider adding a StolenReference type
    - Benedikt - will look at Amos's fix for #891 issue in PR #958
    - Benedikt - will look into Python Buffer branch again
- Soft shutdown PR

Notes

- Check whether we need to change the `WITH_DEBUG` variant of PyObject_TYPE for newer versions of Python (> 3.8?)
    - in newer versions of Python WITH_DEBUG builds have the same layout
    - we need to remove the special case
    - not high priority right now since it's 3.9+
- Switch to .NET Standard for everything but ClrModule
    - Benedikt will complete today to unblock Mohamed
    - Change project files to new format
    - need to get rid of old project files and replace by new ones
    - CLR module needs to stay on old format
- StolenReference type

    - created issue for StolenReference type
- Python Buffer branch again
    - up for grabs if someone else wants to contribute
    - is working fine - was held back until release because it's a Python 3-only branch
    - should be ready to be merged once rebased
- Soft shutdown PR
    - Unity will test before fall
- Python 3.9
    - multiphase initialization for modules
        - used for subinterpreter support among other things
        - every module is an object instead of a singleton
        - can have multiple instances of a module
        - thus there can be multiple instances of the clr module
        - can be opted out of - see PEP 489 (other PEPs are based on it as well)
        - may incur work on Python.NET side to support or disable
    - PEP 554 also relevant (subinterpreters python API for 3.10)
        - interesting for embedding use case of embedding Python into .NET
            - nicer API
        - seems like a good idea to leverage this for our own embedding API
        - needs investigation
- Discussion on broken appveyor.yml fixed in PR #1175

Action Items

- Benedikt to confirm WITH_DEBUG builds need to be updated for 3.9 (low priority) and will create issue to document for later
- Benedikt - Add reference to backports-2.5 branch in README
- Benedikt - Switch to .NET Standard for everything but ClrModule
- Benedikt - will look at Amos's fix for #891 issue in PR #958
- Benedikt - will look into Python Buffer branch again
- Benedikt to look into weirdness with pycparser in appveyor.yml (see PR #1175)

Working group notes for July 16, 2020

Attendees: Benedikt, Mark, Amos

Agenda

- Review action items from last meeting
    - Amos will try to fix soft shutdown 3.8 CI
    - Victor will try to reproduce soft shutdown 3.8 CI failure
    - Since the failure appears to be in codecs tests, Victor will try to check if any codecs fixes were not backported from Lost Tech branch
-

Notes

- Soft shutdown -
    - CI failure related to issue #891 see https://github.com/pythonnet/pythonnet/blob/master/src/runtime/runtime.cs#L311
    - soft mode finished from Amos's side
    - still need test cases for reload mode - may have stability issues
    - Benedikt still reviewing
    - hold off merging until we're sure it's stable
    - Unity will do additional QA (use branch updated to master first)
    - Still some polish to be done
    - Some parts have already been merged to master via other PRs
- small PR #1175 - Remove unnecessary `CheckExceptionOccurred` calls
    - looks good
    - comments from Victor to address
- quantconnect branches
    - will close

Action Items

- Benedikt - Check whether we need to change the `WITH_DEBUG` variant of PyObject_TYPE for newer versions of Python (> 3.8?)
- Benedikt - Add reference to backports-2.5 branch in README
- Benedikt - Switch to .NET Standard for everything but ClrModule
- Benedikt - Change project files to new format
- Victor - Consider adding a StolenReference type
- Benedikt - will look at Amos's fix for #891 issue in PR #958
- Benedikt - will look into Python Buffer branch again

Working group notes for July 2, 2020

Attendees: Victor, Amos

Agenda

- Review action items from last meeting
    - Check whether we need to change the `WITH_DEBUG` variant of PyObject_TYPE for newer versions of Python (> 3.8?)
    - Add reference to backports-2.5 branch in README
    - Switch to .NET Standard for everything but ClrModule
    - Change project files to new format
    - Consider adding a StolenReference type
- PR #958 - what's left, how can we (Unity) help bring this to master?

Notes

- StolenReference - no progress, not gonna work until the next 2 weeks
- PR #958 - still fails tests on Python 3.8

- Codecs use too much reflection, potentially bad for performance. Victor suggests to add a perf test. Alternative approach at https://github.com/joonhwan/pythonnet/blob/fast-call/src/runtime/converter.cs Invocation improment: https://github.com/joonhwan/pythonnet/blob/56e9b1eca1b6a8ed05a77ebd716e68a428be2ce3/src/runtime/methodobject.cs#L289
-

Action Items

- Amos will try to fix soft shutdown 3.8 CI
- Victor will try to reproduce soft shutdown 3.8 CI failure
- Since the failure appears to be in codecs tests, Victor will try to check if any codecs fixes were not backported from Lost Tech branch

Working group notes for June 18, 2020

Attendees:

Agenda

- Review action items from last meeting
    - Benedikt/Mohamed: extract “Moving files around” PR
    - Benedikt/Mohamed: extract “Support for dotnet SDK” PR
- PR #958 - what's left, how can we (Unity) help bring this to master?

Notes

- No new NuGet for 2.x releases due to many packages needed
- We will start releasing NuGet for 3.x dev builds as soon as they are available
- PR opened to drop Python 2 support (1158)
- CLR/CoreCLR loading is striving to be pure Python with no C code to simplify build and deployment
- PR #958 : Amos is syncing it to the latest master. Victor: still quite a few comments pending
- On PyHandle (#1087): Integrate with the existing NewReference and BorrowedReference types, which should be used on the interface to Python

Action Items

- Check whether we need to change the `WITH_DEBUG` variant of PyObject_TYPE for newer versions of Python (> 3.8?)
- Add reference to backports-2.5 branch in README
- Switch to .NET Standard for everything but ClrModule
- Change project files to new format
- Consider adding a StolenReference type

Working group notes for June 4, 2020

Attendees: Benedikt, Victor, Mark

Agenda

- Review action items from last meeting
    - Benedikt/Mohamed: extract “Moving files around” PR
    - Benedikt/Mohamed: extract “Support for dotnet SDK” PR

Notes

- PR 1094 reviewed by Amos (thanks!)
- 2.5 release in progress
    - PR 824 will go into 2.5
    - RC will be released this week - taking a while due to combination of platforms & python versions
    - github master branch will be 2.5 release and only land bug fixes
    - will only publish 2.5 to PyPI - not Nuget
    - Benedikt will start new develop branch after next meeting (June 18th)
    - once that's done we'll start integrating big PRs that break things (including dropping Python 2.x)
- other PRs on hold until after 2.5 release
- Victor's branch (for after 2.5 release)
    - worked on providing Python interfaces to existing .NET collections
    - implemented and injected mixins for stuff from collections.abc into reflected classes implementing interfaces from System.Collections.Generic
    - also implemented buffer interface for arrays
    - depends on features that haven't yet landed upstream
- Python for .NET now part of the .NET foundation!
    - no change to the license
    - after the 2.5 release we can improve our appveyer setup

Action Items

- Benedikt/Mohamed: extract “Moving files around” PR
- Benedikt/Mohamed: extract “Support for dotnet SDK” PR

Working group notes for May 21, 2020

Attendees: Victor, Amos, Mark

Agenda

- Review action items from last meeting
    - Amos to review PR 1094
    - Benedikt/Mohamed: extract “Moving files around” PR
    - Benedikt/Mohamed: extract “Support for dotnet SDK” PR
-

Notes

- Benedikt has been preparing for a release of 2.5

    - basically ready
    - Python 3.8 support has landed
    - updating change log and pip packages left to do
- PR #1094 has landed
- no progress on other two PRs
-

Action Items

- Benedikt/Mohamed: extract “Moving files around” PR
- Benedikt/Mohamed: extract “Support for dotnet SDK” PR
-

Working group notes for May 7, 2020

Attendees: Victor, Amos, Mark

Agenda

- Review action items from last meeting
    - Victor: clean 2.x milestones
    - Benedikt/Mohamed: extract “Moving files around” PR
    - Benedikt/Mohamed: extract “Support for dotnet SDK” PR
    - Add new items to 2.x milestones. If no new items are added, we start cutting release after the first May meeting.
    - Amos: take a look at Finalizer PR: either resolve comments, or reply; accept/decline

Notes

- clean 2.x milestones done, one task left waiting for Amos' review
- Benedikt will try to find time to finish 3.8 support s.t. we can cut the release.
- Unity working on packaging Python 3.8 + Python.NET for Unity package manager
    - will not be discoverable for everyone, but anyone will be able to access it by editing their package manifest
    - there will be a Unity forum announcement sometime in June

Action Items

- Amos to review PR 1094
- Benedikt/Mohamed: extract “Moving files around” PR
- Benedikt/Mohamed: extract “Support for dotnet SDK” PR
-

Working group notes for April 23, 2020

Attendees: Victor, Benedikt, Amos

Agenda

- Review action items from last meeting

    - [x] Review/progress Codec PR on main branch
    - [x] review/progress Codec PR in Codecs
    - [x] Tests are failing in Codecs
    - [x] Mohamed will join modernisation effort
    - [+-] Benedikt will publish PRs for modernisation
    - [+-] Need to determine breakin parts for modernisation
- Approaches for PInvoke: Cecil, GetDelegateForFunctionPointer

Notes

- Dynamic dll: try Cecil to modify assembly in-memory before loading?
- Cecil for now happens only in loader (see modernization PR)
- .NET Core support in 3.0 (e.g. not in 2.x)

Action Items

- Victor: clean 2.x milestones
- Benedikt/Mohamed: extract “Moving files around” PR
- Benedikt/Mohamed: extract “Support for dotnet SDK” PR
- Add new items to 2.x milestones. If no new items are added, we start cutting release after the first May meeting.
- Amos: take a look at Finalizer PR: either resolve comments, or reply; accept/decline

Working group notes for April 9, 2020

Attendees: Victor, Benedikt, Amos, Mohamed, Mark

Agenda

- Review action items from last meeting
    - [x] Viktor to create issue to track removing old solutions
    - [ ] Continue PyHandle/References discussion on GitHub
    - [ ] Amos will create a test for WrapperStorer demonstrating it
    - [x] Amos please review PR #1094
    - [ ] Benedikt will review codec groups PR #1085
    -
-

Notes

- .NET Foundation joining is going forward; signed CLAs
- We should strive for P.NET to be compileable by dotnet CLI; Mono supports it

Action Items

- Review/progress Codec PR on main branch
- review/progress Codec PR in Codecs
- Tests are failing in Codecs
- Mohamed will join modernisation effort
- Benedikt will publish PRs for modernisation
- Need to determine breakin parts for modernisation

- Benedikt will review codec groups PR #1085
- Victor - Reproduce #939

Working group notes for March 26, 2020

Attendees: Victor, Benedikt, Amos, Mohamed, Mark

Agenda

- Review action items from last meeting:
    - [x] Mohamed will work on list and action codec PRs
    - [x] Victor will ask Benedikt about older solution files/create an issue
    - [x] Victor will look again at Mohamed's issue with VS 2019
    - [ ] Continue PyHandle/References discussion on GitHub
    - [x] Victor will start a discussion about serializing user objects - it might be highly misleading
    - [ ] Amos will create a test for WrapperStorer demonstrating it
- release plan?

Notes

- List & action codec PRs
    - working through comments
    - will put function codec into codec repo
- older solution files
    - everything has not been converted to new project format
    - old solution files still used by one of the appveyor builds
        - Mono build uses xbuild?
        - we should use .NET CLI for building
        - what if users only have mono?
        - building by yourself is painful
    - older solutions should be removed
    - Benedikt has removed it in a branch
- serializing user objects
    - discussion will continue in PR
- private mailing list has been created
    - contact Benedikt/Victor if you want to be on the mailing list
    - use this for any future security issues / contacts
- joining .NET foundation
    - got a new questionnaire, upcoming boarding meeting discussing new applications
- please review PR #1094 on finalize that Victor created
- what's going in the next release?
    - finalizing Python 3.x support should go in
    - memory dump can go in
    - nested namespaces?

        - Benedikt: deprecation warning about implicit loading is wrong and we should remove it
    - codec groups?
- re: IronPython compatibility
    - import system vs. import clr
    - current system replaces __import__ and is overly invasive

Action Items

- [ ] Viktor to create issue to track removing old solutions
- [ ] Continue PyHandle/References discussion on GitHub
- [ ] Amos will create a test for WrapperStorer demonstrating it
- [ ] Amos please review PR #1094
- [ ] Benedikt will review codec groups PR

Working group notes for March 12, 2020

Attendees: Victor, Amos, Mohamed

Agenda

- Review action items from last meeting:
    - Victor will create codecs repo on github
    - Amos to create issue or PR to make implicit operator for reference types to convert reference to IntPtr
    - Amos will create wiki page to explain soft shutdown
    - Benedikt will look into creating a private mailing list for maintainers only
    - Everyone - tag PRs you want to land in 2.4.1, we'll try to get merged before next release
    - Benedikt to check that modernization branch will work with 3.8
-

Notes

- Codecs repo with stub solution created
- VS requirement - 2015 no longer works since C# 7.3 requirement
- WrapperStorer - facility for user to preserve objects linked from Python after domain reloads
- Action passing needs more work, might need IL generation, List codec OK

Action Items

- Mohamed will work on list and action codec PRs
- Victor will ask Benedikt about older solution files/create an issue
- Victor will look again at Mohamed's issue with VS 2019
- Continue PyHandle/References discussion on GitHub
- Victor will start a discussion about serializing user objects - it might be highly misleading

- Amos will create a test for WrapperStorer demonstrating it

Working group notes for Feb 27, 2020

Attendees: Victor, Mark, Benedikt, Benoit, Amos, Felix

Agenda

- Review action items from last meeting:
    - Benoit to finish wiki page for threading
    - Victor will talk to Benedikt regarding creating a lib/separate repo for codecs (this also applies to QuantConnect)
    - Quantconnect PR - Victor to ensure Amos commits are mirrored in master
    - Amos should take over PR #1016
    - Victor will try to review #958 instead
    - See if Benedikt needs better time or we can make decisions while he's off
    - Unity is settled on 2.4.0, and might reduce presence
- Create NuGet organization and transfer package ownership for NuGet package
    - At least 1 more person to admin NuGet org
- Mailing list for maintainers only
- A plan to replace IntPtr with the new reference types
- Upstreaming Lost Tech (Victor's) fork
    - Python 2 support?
    - DllImport -> GetDelegateForFunctionPointer

Notes

- Wiki page for threading is up in tutorials section
- Victor & Amos discussing failing multithreading test case in PR #1022
- Looks like Amos's commits in the Quantconnect PR are already in master
- PR #1016 closed in favour of using PR #958 instead
    - Slow going on PR #958 review
    - needs some description to explain the changes - a few sentences that describe the intended effects of the changes: what actions are necessary to implement soft shutdown?
    - hard to get overall picture of what the code is expected to do
- Regarding the time slot
    - this is the best we can get that balances China, North America and Germany time zones
    - it's OK for Victor make decisions in the meeting - unless Benedikt is completely against the decision, it will be OK
- NuGet org
    - Victor has access
    - problem is that registered e-mail is the mailing list e-mail, security hole
- A plan to replace IntPtr with the new reference types
    - we should avoid using IntPtr in our own pull requests

    - should use proper pointer types ourselves, once we've got good practices established can be documented
    - belongs with Benedikt's work on loader PR
- Upstreaming Lost Tech (Victor's) fork
    - Python 2 support not handled, should we drop support from master (and move 2.x to branch)?
    - DllImport -> GetDelegateForFunctionPointer change require dropping 2.x
    - in general dropping 2.x helps with a lot of things
    - not a problem for Unity, also moving to 3.x
- No arguments against dropping support for 2.x in master were raised
    - is master in a good state for a last release of 2.x?
    - have one last push to get any changes not specific to 3.x into master before branch-off / tag
        - codecs
        - other minor changes
    - stability of current release? random CI failures
        - we added more tests recently - e.g. domain reload tests unstable
        - one of the domain reload tests often fails on Python 2.7
        - Amos' changes for normal mode shutdown should solve these failures - should we land them first? no, let's only backport them if there is demand
    - master is currently green
- As soon as we have all PRs we want in, we will tag 2.4.1 (if no breaking changes, 2.5 otherwise) and branch if necessary
    - will use a github milestone to track PRs for 2.4.1
- Still failing tests on 3.8, but likely a dependency on 3.7 - next release should have 3.8 support
- Lost Tech branch has support for Python version independence (except for 3.8)
    - complete version independence requires dynamic tp slot handling
    - will need runtime logic for handling different versions of Python to handle differing slot locations
    - also need to remove PInvoke usage
    - solving for these two issues will allow any Python version to be used at runtime instead of requiring pre-build for specific versions
    - Lost Tech branch doesn't handle embedding C# in Python, but CLR loader PR should take care of this - already working, and should work with Lost Tech branch changes
- Dynamic CLR loading won't support 2.4

Action Items

- Victor will create codecs repo on github
- Amos to create issue or PR to make implicit operator for reference types to convert reference to IntPtr
- Amos will create wiki page to explain soft shutdown
- Benedikt will look into creating a private mailing list for maintainers only

- Everyone - tag PRs you want to land in 2.4.1, we'll try to get merged before next release
- Benedikt to check that modernization branch will work with 3.8

Working group notes for Feb 13, 2020

Attendees: Victor, Benoit, Amos, Mohammed

Agenda

- Review action items from last meeting:
    - Benoit will write github issue documenting GIL best practices when running multithreaded code and possible solutions [in progress, wiki page + tests]
    - Benedikt will look into splitting up CLR loader into multiple work items to speed up .NET Core support
    - Victor will address comments on codec PR - ready to check in
    - Victor - will work on breaking up Quantconnect PR
    - Mohamed will rebase #974 on top of Victor's codec PR
    - ? - github actions
    - Victor will send Benoit & Felix build that failed on Linux 3.7
    - Benoit will continue working on PR #1016
    - Amos is having connection issues and will send an update email to the mailing list

- Create NuGet organization and transfer package ownership for NuGet package

Notes

- Benoit will work on wiki page + tests for GIL
- Quantconnect PR
    - Converter-related stuff depends on codecs repository
    - Performance in separate PR
- Mohamed will rebase #974 - handle generic parameters on action type, needs repo for codecs
- Victor made repo + NuGet with packaged Python

Action Items

- Benoit to finish wiki page for threading
- Victor will talk to Benedikt regarding creating a lib/separate repo for codecs
- Quantconnect PR - Victor to ensure Amos commits are mirrored in master
- Amos should take over PR #1016
- Victor will try to review #958 instead
- See if Benedikt needs better time or we can make decisions while he’s off
- Unity is settled on 2.4.0, and might reduce presence

Working group notes for January 30, 2020

Attendees: Benedikt, Victor, Amos, Benoit, Mark

Agenda

- Review action items from last meeting:
    - Benedikt will continue working on .NET foundation application
    - Victor - will continue review of CLR loader PR
    - Benedikt - review codec PR please
    - Benoit - multithreaded python code
    - Victor - will work on breaking up Quantconnect PR
    - Mohamed will rebase #974 on top of Victor's codec PR
    - ? - github actions
    - Amos - will take changes made to PR #958 and push to #1016, address comments
-

Notes

- Benoit looked at releasing threads on initialization return for multithreaded python code
    - use case is initializing Python.NET in Unity, on return GIL is held, workaround doesn't need Python.NET change (just release GIL from C#)
    - should the solution be in user code or Python.NET? discussion around tradeoff between complexity for users and correct behaviour
    - Could this be opt-in via e.g. attributes? So existing code continues to work without change.
    - Currently PyGIL attribute doesn't do anything, purely documentation
    - Current behaviour is somewhat surprising
    - First step is to document behaviour
- Benedikt still working on .NET foundation application
    - seems to be in some triage stage, not rejected but not progressing either
    - will continue with personal membership and keep application moving
- CLR loader
    - still WIP, can load any CLR into Python
    - dynamic loading of Python.NET is messy,
        - problems around PInvoke and content string, works in Mono and .NET framework
        - should work in .NET core but first function call that hits Python function fails with missing DLL from the PInvoke call
        - we want PInvoke to call into symbols in the existing namespace so can use current Python exe (without Python so/dll)
        - very WIP
    - could this be split into multiple work items?
    - goal is to have a single deployment, but maybe this work can be split up

    - there are a lot of people asking for .NET core support, splitting would help that
- Codec PR
    - target a specific version or just merge into master once it passes CI?
        - merge into new branch for next version?
        - or make functions internal and use [InternalsVisibleTo] in master?
        - just land in master
    - tuple converter looks more like an example of how code should be done - maybe move out of main assembly to an example project?
    - internal conversions should be rewritten using codec API
- PR #1016
    - Amos merged changes to the PR
    - Some test failures need to be addressed
        - Linux 3.7 crash
        - not sure if errors are legitimate or test framework failure
        - seems to be infrastructure instability? Victor saw same crash on other branch
        - try again?
- Is there a release schedule? not really
    - want to get Amos' big change into master before next release so we can update Python for Unity (sometime in 2020)
    - needs input from Benedikt

Action Items

- Benoit will write github issue documenting GIL best practices when running multithreaded code and possible solutions
- Benedikt will look into splitting up CLR loader into multiple work items to speed up .NET Core support
- Victor will address comments on codec PR
- Victor - will work on breaking up Quantconnect PR
- Mohamed will rebase #974 on top of Victor's codec PR
- ? - github actions
- Victor will send Benoit & Felix build that failed on Linux 3.7
- Benoit will continue working on PR #1016
- Amos is having connection issues and will send an update email to the mailing list

Working group notes for January 16, 2020

Attendees: Felix, Mark, Victor, Mohamed, Amos

Agenda

- Review action items from last meeting:
    - Benedikt will continue working on .NET foundation application
    - Victor will take a look at the CLR loader PR, looks big though
    - Victor will look into if we can generate perf stats through AppVeyor and Travis
    - Victor will publish a PR for conversion config

    - Mohamed will split PR #974
    - Benoit - multithreaded python code
    - Victor - breaking up Quantconnect PR
    - ? - github actions
    - Benedikt - will try to reproduce PyBuffer crash locally

Notes

- Victor took a look at CLR loader PR, review in progress
- Perf stats on hold as they aren't supported by Appveyor/Travis, would need an external server to record stats
- Codec PR published, needs review from Benedikt
- PR #1016 - some bugs in this PR have been fixed in #958 by Amos
- Felix working through domain reload changes, updating to master and reviewing PR
    - plan is to break up PR #958

General note: please participate in PR reviews and/or take a look at the PyBuffer crash.

Action Items

- Benedikt will continue working on .NET foundation application
- Victor - will continue review of CLR loader PR
- Benedikt - review codec PR please
- Benoit - multithreaded python code
- Victor - will work on breaking up Quantconnect PR
- Mohamed will rebase #974 on top of Victor's codec PR
- ? - github actions
- Amos - will take changes made to PR #958 and push to #1016, address comments

Working group notes for December 19, 2019

Attendees: Benedikt, Victor, Amos, Mohamed, Mark

Agenda

- Review action items from last meeting:
    - Benedikt - continue working on joining .NET foundation
    - Benoit - multithreaded python code
    - ? - breaking up Quantconnect PR
    - ? - github actions
    - Amos - continue breaking up pull 958 into smaller PRs to make it easier to review and merge. e.g. have a single PR that gets reference counts right and doesn't touch GC chains. See Benoit's example of the __import__ restore PR for best practices.
    - Amos - same for other older PRs
    - Benedikt - will try to reproduce PyBuffer crash locally
    - Benedikt - will check if there are any breaking changes since 2.4.0
    - Benedikt - continue exploring CLR loader, splitting parts into separate Python interop DLLs, have main Python runtime implement layers on top

    - Victor - Perf runs in CI
- Review PRs
- Mailing list
    - currently moderated - @markv asked for ownership to be transferred to Benedikt

Notes

- joining .NET
    - process had changed, Benedikt working on it
- Pull 958 (domain reload handling)
    - split PRs being approved and merged, thanks Amos!
- github actions
    - if we join .NET we can stay on current setup
    - would be nice to have Darwin CI working
- PyBuffer
    - nothing done yet
- CLR loader
    - modernization PR has a single build
    - Python 2 should still work, but Benedikt won't spend a lot of time on Python 2, 3.5+ is current focus
    - moved code and tests into separate directories
    - removed old csproj file in favour of small project files generated using .NET CLI
    - goal is a fully building project using .NET CLI that is operating system independent
    - made a small pythonnet module that will be the future entry point
    - still rough, testing currently with core CLR and Mono on Linux
        - loading works, P/Invoke __internal breaks on .NET core
    - Python functions have been split out into an interface
    - using T4 templates
    - ready for early review if you can take a look
- Performance tests and CI
    - not very stable if we put in CI, fail depending on weather conditions
    - no way to put in CI without random failures
    - maybe makes sense to keep it as an ongoing statistic on master and not blocking PRs
    - useful for running locally on branch vs. master to check for regressions
    - still informative for detecting regressions - we can run it on every PR, but it shouldn't result in a hard error - can we report it on the PR?
    - perf comparison is relative, so different machines shouldn't affect the result
- Mailing list
    - keep it, but remove moderation, transfer ownership to Benedikt
- Improve method binding PR #974 discussion
- Meeting on January 2 is cancelled, next meeting will be in 4 weeks on Thursday, January 16, 2020

Action Items

- Benedikt will continue working on .NET foundation application
- Victor will take a look at the CLR loader PR, looks big though
- Victor will look into if we can generate perf stats through AppVeyor and Travis
- Victor will publish a PR for conversion config
- Mohamed will split PR #974
- Benoit - multithreaded python code
- ? - breaking up Quantconnect PR
- ? - github actions
- Benedikt - will try to reproduce PyBuffer crash locally

Working group notes for December 5, 2019

Attendees: Mohamed, Amos, Mark

Agenda

- Review action items from last meeting:
    - Benedikt - continue working on joining .NET foundation
    - Benoit - multithreaded python code
    - ? - breaking up Quantconnect PR
    - ? - github actions
    - Amos - continue breaking up pull 958 into smaller PRs to make it easier to review and merge. e.g. have a single PR that gets reference counts right and doesn't touch GC chains. See Benoit's example of the __import__ restore PR for best practices.
    - Amos - same for other older PRs
    - Benedikt - will try to reproduce PyBuffer crash locally
    - Benedikt - will check if there are any breaking changes since 2.4.0
    - Benedikt - continue exploring CLR loader, splitting parts into separate Python interop DLLs, have main Python runtime implement layers on top

Notes

- Quorum not reached, meeting cancelled

Action Items

Working group notes for November 21, 2019

Attendees: Victor, Benoit, Mark, Amos, Benedikt

Agenda

- Review action items from last meeting:
    - Mark - got in touch with .NET foundation

- Benoit - document how to use pythonnet when you expect to run multithreaded python code (BeginAllowThreads/EndAllowThreads) in an embedded interpreter
- Victor - will create Github issues for items in the Quantconnect PR and figure out a plan for bringing more generic solutions to master.
- ? - can someone research how hard it would be to move the CI to Github Actions?
- Benoit - will break out SlotsHolder and __import__ restore into smaller PRs
- Amos - will continue to break https://github.com/pythonnet/pythonnet/pull/958 into smaller PRs following Benoit's example
- All - please discuss changing the meeting time on the list

Notes
.NET foundation - Benedikt working on it with contact & maintainer of IronPython

Multithreaded python code - Benoit still working on it, has a deadlock issue that's probably unrelated. WPF has a similar issue? There are old tickets around where people tried this before. Possibility of making it automatic, but requires a major version bump. https://github.com/pythonnet/pythonnet/issues/990 for example.

Quantconnect PR - perf PR was OK, but is a fairly small part of the larger PR. Need to get .NET standard / core stuff on the way so that they don't fork off.

Github actions - currently have appveyer and travis CI implementations, but in theory github actions can do both and also work on mac. Could cover all platforms.
One of the unit tests crashes on Mac - ongoing problem, would be good to have CI for that.
Possible plan is to start with github actions on Mac, then extend to Linux, finally Windows.

__import__ restore - merged (moved out of pull 958)

SlotsHolder in progress - goal is to make it so that PyFinalize doesn't need to be called to disconnect C# from Python (moved out of pull 958)

General note: please make PRs as small and targeted as possible - large PRs are very difficult to review and judge the impact and risk. Large PRs won't likely be merged.
Discussion around time to accommodate Amos - we will move the meeting 3 hours earlier.

Need to start planning for a point release (2.4.1) including Python 3.8 support.
PyBuffer change - good addition but is crashing in a non-obvious way. Maybe not a candidate for the point release. If someone needs it for the next release, they could try to fix it in branch. We shouldn't have any breaking changes since 2.4.0, only bug fixes.

Benedikt - Haven't been able to integrate CLR loader yet (ripping out old stuff). Will try to have the interface ready for the CLR loader. Python.NET exposes a clr module inside Python which exposes .NET VM startup functions. Mono vs. Core have different code paths currently. Current approach in latest .NET core seems to be stable. New CLR loader

abstracts three different ways of loading .NET. Technical discussion. Proposal: pass parameters as JSON. Alternatives? Would also enable choosing runtime version of Python.

Benchmarking - Still in progress. PR will be open today.

Could CLR loader be completely separate from Python.NET?
Benedikt - sure, it's useful on its own as a low-level mechanism to talk to .NET. On the other hand Python interop is interconnected with how Python.NET works and should stay in the project. No plans to have more libraries than CLR loader (which isn't actually needed if you're embedding Python within .NET) and pure interop + abstractions on top.

Action Items
Benedikt - continue working on joining .NET foundation
Amos - continue breaking up pull 958 into smaller PRs to make it easier to review and merge. e.g. have a single PR that gets reference counts right and doesn't touch GC chains. See Benoit's example of the __import__ restore PR for best practices.
Amos - same for other older PRs
Mark - move meeting 3 hours earlier
Benedikt - will try to reproduce PyBuffer crash locally
Benedikt - will check if there are any breaking changes since 2.4.0
Benedikt - continue exploring CLR loader, splitting parts into separate Python interop DLLs, have main Python runtime implement layers on top

Working group notes for November 7, 2019

Attendees: Benoit Hudson (ImgSpc), Amos Li, Mark Visser (Unity)

Agenda
- Review action items from last meeting

Notes
Update from Victor: The only thing I've done though is reviewing PyBuffer request PR: https://github.com/pythonnet/pythonnet/pull/980, for which, sadly, tests are still not passing.

Update from Benedikt: I've been mostly working on clr_loader, will open a PR on pythonnet for this weekend.

.NET foundation:
- Mark finally got a reply back from his contact at Unity
- He will loop Benedikt in on the e-mail as soon as he's introduced to someone on the .NET foundation

Soft restart:
- PR is large and hard to review
- https://github.com/pythonnet/pythonnet/pull/958#issuecomment-551230878

- documented here: https://docs.google.com/document/d/1a9OLsdKHXJ6MxHjo0WlVRcfjAiaP5hONNKUKKH6Tr-o
- Benoit will break out SlotsHolder and __import__ restore into smaller PRs
- Amos can you continue to break https://github.com/pythonnet/pythonnet/pull/958 into smaller PRs to land individually?
- discussion about system modules restoration - clarity found
- split removal of native code page into another PR after everything else has landed

Regarding the meeting time:
- Amos pointed out that the meeting is at 4am his local time - could we move this meeting to a better time for him?
- possible times:
    - 4 hours earlier (12pm EST, 8am PST, 12am China)
    - 2 hours later (5pm EST, 11am PST, 6am China)
    - 3 hours later (6pm EST, 12pm PST, 7am China)
    - please discuss!

Action Items
Mark - still working on getting in touch with .NET foundation
Benoit - document how to use pythonnet when you expect to run multithreaded python code (BeginAllowThreads/EndAllowThreads) in an embedded interpreter
Victor - will create Github issues for items in the Quantconnect PR and figure out plan for bringing more generic solutions to master.
? - can someone research how hard it would be to move the CI to Github Actions?
Benoit - will break out SlotsHolder and __import__ restore into smaller PRs
Amos - will continue to break https://github.com/pythonnet/pythonnet/pull/958 into smaller PRs following Benoit's example
All - please discuss changing the meeting time on the list

Working group notes for October 22, 2019

Attendees: Benedikt Reinhartz (Project Lead), Benoit Hudson (ImgSpc), Felix Bourbonnais (ImgSpc), Mohamed Kouba (ANSYS Inc.), Victor Milovanov, Amos Li, Mark Visser (Unity)

Agenda

1. (Mark) I got a reply from Jared Broad (QuantConnect): His team is unable to devote resources to anything other than the PR they agreed to open (#895). He wanted to remind us that he proposed making a $500/month contribution, which combined with other sponsors could have a full-time person working on pythonnet.

   I took a look through their PR and the changes seem to roughly fit into one of these categories:
    - stylistic (e.g. int vs. var)
    - version change (1.0 vs. 2.4)

- correctness checks & better support for debugging (e.g. TRACE_ALLOC, FINALIZER_CHECK)
- code removal due to earlier branch point (domain reload work)
- additional type support (datetime, timespan)
- changes to support Python 3?
- unclear modifications (e.g. removal of initSigs parameter)

Are any of these non-contentious? If so, could we cherry-pick those changes at least?

Notes

Benoit - Felix & Amos are talking on domain reload - next step - write up possible solutions. Working towards consensus, Amos is solving the problem more generally than we are. Likely will be two patches.
Victor - If there will be multiple patches, can we split into two PRs?
Benoit - that's the goal.
Victor - Have two PRs that are ready for review, please review. New PR that introduces performance tests, not yet working with CI. Needs to be split into two PRs: test framework and actual tests. This will enable adding performance tests for any feature.

Victor - re Quantconnect PR, can we make some of these changes generic and bring them to master? Victor will create issues and the plan for PRs to bring them to master. Goal is for Quantconnect to not have to modify base source code.

Mark - re Jared's suggestion of $500/month: Unity can't pay someone to maintain Python for .NET without a lot of legal overhead (master service agreement, statement of work, etc). Joining the .NET foundation may help with this.

Mark - will move meeting to Thursday to assist Benedikt joining

Benedikt- Still didn't hear anything from the .NET Foundation even though I filled their questionnaire now twice
Mark - trying to contact .NET foundation from inside Unity, but my contact is on vacation. Will continue when he returns.

Benedikt:
- I started work on separating the CLR loading out into a separate Python package with implementations for Framework, Core and Mono. This should simplify our build-process a lot.
- I started (again) to drop the old csproj files in favour of proper Sdk-style files, will try to finish that after the loader is done
- I really like Victor's work on the benchmark, will look into the PR tomorrow. If this turns out well, I'd really like to see where we would get by replacing the P/Invoke style by explicit delegates, as this would finally open up a proper route towards a single Nuget package

- After the build-system improvements I'll go forward on the IEnumerable changes started in PR 963. The idea is to drop (almost) all implicit conversions in favour of implementing the correct protocols for .NET objects (leveraging duck typing).
- Since I don't use embedding very much, please implement your proposals for domain reload etc. as you please and I'll just have a final look on the respective PRs. I don't have an informed opinion on these cases.

Discussion about performance. For good performance you'd need nice C# access to the Buffer protocol: https://docs.python.org/3/c-api/buffer.html (also available in Python 2.7: https://docs.python.org/2/c-api/buffer.html)

Mohamed- C# method that takes IEnumerable can't be called with Python lists and tuples. Should that work? Worked this way in IronPython, but not ideal.
Victor- Have been working on another approach with static list of conversion codecs. You can add your own codec to the list to handle arbitrary conversions. Still WIP, needs design discussion.
Benedikt- What about filtering by assembly or class in registration?
Victor- could achieve that by subclassing a codec and adding a filter for assembly/class.
Discussion on corner cases around automatic conversion of parameters. Implicit conversion is tricky.
Mohamed- is there a way to register at runtime? (with your conversion API)
Victor- yes, registration is via singleton method. Worth some discussion.
Benedikt- would it be better to always require function-based registration? i.e. argument & return type conversion?
Discussion on parameter conversion between Python and C#. Non-trivial to do automatic conversion, too many corner cases.
Amos- showed proposal for conversion API
Victor- how to handle generic types? will send Amos example.

Benedikt- if anyone has time to look into Github Actions, they are a potential way to have a single way to run CI on both Windows and *nix.

Action Items
Benoit - document how to use pythonnet when you expect to run multithreaded python code (BeginAllowThreads/EndAllowThreads) in an embedded interpreter
~~Benoit - document my understanding of soft restart, work with Felix & Amos to get to a shared understanding. Goal is two PRs for domain reload.~~
Victor - will create Github issues for items in the Quantconnect PR and figure out plan for bringing more generic solutions to master.
~~Mark - move meeting to Thursday, same time to accommodate Benedikt's schedule~~
? - can someone research how hard it would be to move the CI to Github Actions?
~~Victor - send complex type conversion example to Amos~~

Working group notes for October 8, 2019

Attendees: Benoit Hudson (ImgSpc), Mohamed Kouba (ANSYS Inc.), Victor Milovanov, Amos Li, Mark Visser (Unity)

Agenda

1. (Mark) Suggest we use a "needs discussion" label on issues and PRs for next time. Then we can just go through them on the call. Who can tag issues?
2. (Mark) Can we get a status update on joining the .NET Foundation? Github issue: Joining the .NET Foundation - Any response?
3. (Mark) PR review
4. (Mark) Issues review

Notes

Mark - no news on .NET foundation progress, Mark is investigating internally through Unity if there's some way to get the process moving.

Benoit - Amos & Felix still need to connect and chat re: PR#957
Benoit - re: grabbing the GIL in issue#964 - this is by design and should stay that way. Need to document implications of multithreaded environment when Python is embedded in a C# application. GIL needs to be explicitly released and regrabbed via BeginAllowThreads/EndAllowThreads otherwise the process will crash. No change is proposed to this behaviour.

Victor - will move PRs that aren't ready for merge to draft state so we have a clearer idea of what's actually in process.

Mohamed - in IronPython you can have more than one instance running, i.e. more than one scope.
Victor - Python 3.7 or 3.8 has/will/may have sub-interpreters working..
Mark/Benoit - RpyC + subprocesses can achieve the same effect but has marshalling/IPC overhead
Mohamed - what's the process for supporting a feature like sub-interpreters in pythonnet
Victor - basically propose it at a meeting, get alignment on design, write the code

Amos - is there any plan for improving performance?
Victor - need performance tests before we check in any PRs around performance, no time at the moment. Hopefully in time for the next meeting or shortly after.

Action Items

Mark - see if there's a path to the .NET foundation internally and get our application unblocked.
Benoit - coordinate discussion between Amos & Felix on #957

Benoit - document how to use pythonnet when you expect to run multithreaded python code (BeginAllowThreads/EndAllowThreads) in an embedded interpreter
Victor - move PRs that aren't ready for merging to draft status
Mohamed - update #808 according to feedback, time permitting

Working group notes for September 24, 2019

Attendees: Mark Visser (Unity), Benoit Hudson & Felix Bourbonnais (ImgSpc), Victor Milovanov, Mohamed Koubaa (ANSYS Inc), Amos Li

Agenda

1. (Mark) Suggest we use a "needs discussion" label on issues and PRs for next time. Then we can just go through them on the call.
2. (Mark) Can we get a status update on joining the .NET Foundation? Github issue: Joining the .NET Foundation
3. (Benoit) Python can't reliably re-initialize; new proposal for Unity's domain reload
4. (Benoit) What's the policy with the GIL? Main thread appears to be grabbing the lock and not releasing it.
5. (Mohamed) Can I get clarification on IWrapper vs. reflection for PR #808?

Notes

Welcome to new contributor Mohamed Koubaa (ANSYS), who is investigating moving his company's platform from IronPython to CPython.

1 & 2) need input from Benedikt, Mark will follow up with Benedikt.

4)
Benoit: GIL - Should we expect the GIL to be held after initialization?
Victor: I wouldn't expect it to be held. [PyGIL] attribute is informational.

3)
Benoit: Python can't reliably re-initialize.
- last fall's work - force PyFinalize to be called
    - unfortunately doesn't work for native extensions - can't reliably reinitialize
- proposal
    - instead of calling PyFinalize, delete all links from Python to C#
    - Felix prototyped a solution that tracks all links during proxy generation and cuts all links during domain shutdown to prevent dereference of dead pointers
    - result is just AttributeError instead of crash
    - Amos proposed a different solution which seems to be targeting memory leaks
    - **Victor**: what about memory leaks?
    - C# side deletes all of its memory

    - If C# holds a reference to Python, it will still exist when C# shuts down - no problem
    - **Amos**: has a branch that solves one of the problems
    - Benoit & Felix will investigate his branch and see where it intersects with their solution
- the reason we're using pythonnet is to enable PySide/PyQt and numpy among others

5) **Mohamed**: Working on a small change to implement the wrapper function for console output via __repr__ https://github.com/pythonnet/pythonnet/pull/808

- inheritance over reflection, IWrapper vs. method with a signature Pythonnet can reflect
- what are the pros and cons? should we go with a convention?
- **Victor**: you can take the reflection approach, but need to cache the delegates
    - more of an implementation detail, not actually necessary

Action Items:

Mark

- follow up with Benedikt about Python foundation and using issues to track discussion items

Benoit & Felix

- investigate why GIL is held indefinitely
- test out Amos's branch and connect over gitter to discuss what parts of the problem each approach is solving

Thank you all for attending! The next call will be held at 3pm EST on Tuesday, October 8th, 2019, see you then. Please e-mail me if you're not on the attendee list and I will add you.

Working group notes for September 10, 2019

Here are the minutes from Tuesday's meeting.

Present: Victor Milovanov, Amos Li, Mark Visser

Mark mentioned that his team has found a solution to C# domain reload that doesn't crash the Unity Editor. Benoit Hudson will join the next meeting to discuss with the group.
Victor has been working on the ability to inherit C# classes from Python classes and override Python methods in C# using attributes.

The next meeting will held on Tuesday, September 24th at 3pm EST. If you have not received an invite, please e-mail me directly and I'll add you to the event.

Also please note that I went ahead and switched to Zoom for hosting meetings as Google Hangouts is blocked in the PRC.

I'll ask the list for agenda items the week before the meeting. Hope to see you all online next time.

Working group notes for June 25, 2019

Present:
denis.akhiyarov@gmail.com
amos_402@msn.com
jared@quantconnect.com
alex@quantconnect.com
lostfreeman@gmail.com
filmor@gmail.com
markv@unity3d.com

(sorry if I missed anyone)

- Denis (denis.akhiyarov@gmail.com) stepped down from project ownership
- Benedict (filmor@gmail.com) is most senior python net developer so will take over project ownership
- Mark (markv@unity3d.com) offered to take care of administrative stuff (meetings, etc) and will schedule next meeting 2 hours earlier on Tuesday July 2
- should pythonnet join the .NET Foundation? AppVeyor account might make it worth it
  - vote next meeting
- development resources for the future?
  - performance improvements
  - missing features
  - maintenance
- merging QuantConnect fork into main repo? Benedict will review QC's fork with Jared (jared@quantconnect.com) and merge or suggest changes
- Benedict will trim admin rights to only active committers
- Please use issues for proposed work to be done and use "Epic" tag. suggested:
  - .NET core work
  - Python 3.8 work
  - other refactoring or features
- Victor (lostfreeman@gmail.com) will make PRs for key parts from his fork
- Benedict suggested that we should move as much work as possible into the main repo
  - concern from QuantConnect that their velocity might be impacted
  - mitigated by keeping work flowing from QC fork to pythonnet/pythonnet?