Psi4: THEN AND NOW

ANDREW C. SIMMONETT
JUSTIN M. TURNLEY
LORI A. BURNS
DEVELOPERS MEETING, UGA
4 NOVEMBER 2016
PART I: CONTRIBUTING TO Psi4

ANDY C. SIMMONETT
USING GITHUB’S PULL REQUESTS
DIRECT PUSHES TO MASTER NO LONGER ALLOWED

GitHub
psi4/psi4
USING GITHUB’S PULL REQUESTS
DIRECT PUSHES TO MASTER NO LONGER ALLOWED

GitHub
psi4/psi4

GitHub
developer/psi4
USING GITHUB’S PULL REQUESTS
DIRECT PUSHES TO MASTER NO LONGER ALLOWED
ISSUING A PULL REQUEST
FROM YOUR PERSONAL GITHUB HOMEPAGE

Launching a PR

The PR info screen
INTERACTIVITY OF PULL REQUESTS
GETTING FEEDBACK FROM OTHER DEVELOPERS

The review screen

The conversation screen
AUTOMATIC CODE TESTING
SEEING WHICH COMMITS BREAK THE TESTS

Rebased through USAPTO changes.

loriab and others added some commits 12 days ago

- update samples
- patch up docs build for inversion
- de-magic methods
- INV: Started pure python test cases
- INV: Fixed travis, adds explicit occupation test cases
- INV: Helps to add the changes to runtest
- Initial stab at atomic basis sets for SAD
- mod atomic basis for mints2, x2c1
- INV: SAD guess should be working again.
- INV: Removed old C-side pyconstruct members and fixed datadir input

Getting pretty close to merging this in. Failing tests case should be limited to the following:

- 22 - cc14 (Failed)
- 155 - docs-psimod (Failed)
- 187 - mints9 (Failed)

Docs and mints both require changes to the test case. cc14 remains the only undiagnosed test failure.
CONTINUOUS INTEGRATION TOOLS
TRAVIS / DISTELLI / CONDA-BUILD

psi4 / psi4

Build #581

- #581 passed
- Elapsed time 50 min 50 sec
- Total time 2 hrs 58 min 21 sec
- 13 days ago

Build Jobs

- #581.1
  - Compiler: clang C++
  - CXX_COMPILER='clang++-3.6' C_COMPILER='clai'
  - 35 min 40 sec

- #581.3
  - Compiler: gcc C++
  - CXX_COMPILER='g++-4.9' C_COMPILER='gcc-4.9'
  - 43 min 20 sec

- #581.4
  - Compiler: gcc C++
  - CXX_COMPILER='g++-6.6' C_COMPILER='gcc-6'
  - For
  - 50 min 48 sec

- #581.5
  - Compiler: gcc C++
  - CXX_COMPILER='g++-6.6' C_COMPILER='gcc-6'
  - For
  - 47 min 57 sec

Allowed Failures

- ! #581.2
  - Compiler: clang C++
  - CXX_COMPILER='clang++-3.9' C_COMPILER='clai'
  - 36 sec
ACCEPTING/MERGING A PR
AFTER TEAM APPROVAL

andysim commented 10 days ago
I love the python driver and the extra docs. Fantastic stuff! LGTM

jturney commented 10 days ago
LGTM

loriab commented 10 days ago
LGTM

loriab merged commit cdfd396 into psi4:master 10 days ago

3 checks passed
- approvals/lgtm this commit looks good
- continuous-integration/Distelli
- continuous-integration/travis-ci/pr The Tra...
WORKFLOW FOR CONTRIBUTING
MAKING THE CHANGES/SETTING UP THE PR

1) git clone git@github.com:andysim/psi4
   git remote add upstream git@github.com:psi4/psi4
WORKFLOW FOR CONTRIBUTING
MAKING THE CHANGES/SETTING UP THE PR

1) `git clone git@github.com:andysim/psi4`
   `git remote add upstream git@github.com:psi4/psi4`

2) `git checkout -b fastci`
WORKFLOW FOR CONTRIBUTING

MAKING THE CHANGES/SETTING UP THE PR

1) `git clone git@github.com:andysim/psi4`
   `git remote add upstream git@github.com:psi4/psi4`

2) `git checkout -b fastci`

3) `git push origin fastci`
WORKFLOW FOR CONTRIBUTING
MAKING THE CHANGES/SETTING UP THE PR

1) git clone git@github.com:andysim/psi4
   git remote add upstream git@github.com:psi4/psi4

2) git checkout -b fastci

3) git push origin fastci

4) [New pull request]
WORKFLOW FOR CONTRIBUTING

MAKING THE CHANGES/SETTING UP THE PR

1) git clone git@github.com:andysim/psi4
   git remote add upstream git@github.com:psi4/psi4

2) git checkout -b fastci

3) git push origin fastci

4) [Pull request icon] New pull request

5) Add new code on fastci branch
### WORKFLOW FOR CONTRIBUTING

#### MAKING THE CHANGES/SETTING UP THE PR

1. `git clone git@github.com:andysim/psi4`
   - `git remote add upstream git@github.com:psi4/psi4`

2. `git checkout -b fastci`

3. `git push origin fastci`

4. ![New pull request](https://raw.githubusercontent.com/andysim/psi4/master/images/new_pull_request.png)

5. Add new code on fastci branch

6. `git push origin fastci`
WORKFLOW FOR CONTRIBUTING
SYNCHRONIZING / CLEANING REPOSITORIES

7) New feature added on master by somebody else (via a PR)
WORKFLOW FOR CONTRIBUTING
SYNCHRONIZING / CLEANING REPOSITORIES

7) New feature added on master by somebody else (via a PR)

8) git pull --rebase upstream master
WORKFLOW FOR CONTRIBUTING
SYNCHRONIZING / CLEANING REPOSITORIES

7) New feature added on master by somebody else (via a PR)

8) `git pull --rebase upstream master`

9) `git push origin fastci` (may need to add `--force`)
7) New feature added on master by somebody else (via a PR)

8) `git pull --rebase upstream master`

9) `git push origin fastci` (may need to add `--force`)

10) PR accepted, after “ready to go” checked, tests, LGTM approval
WORKFLOW FOR CONTRIBUTING
SYNCHRONIZING / CLEANING REPOSITORIES

7) New feature added on master by somebody else (via a PR)

8) `git pull --rebase upstream master`

9) `git push origin fastci` (may need to add `--force`)

10) PR accepted, after “ready to go” checked, tests, LGTM approval

11) `git checkout master`
    `git pull --rebase upstream master`
    `git push origin master`
### WORKFLOW FOR CONTRIBUTING

**SYNCHRONIZING / CLEANING REPOSITORIES**

<table>
<thead>
<tr>
<th>upstream</th>
<th>origin</th>
<th>local</th>
</tr>
</thead>
<tbody>
<tr>
<td>psi4 / psi4</td>
<td>andysim / psi4</td>
<td></td>
</tr>
<tr>
<td>master</td>
<td>master fastci</td>
<td>master fastci</td>
</tr>
</tbody>
</table>

7) New feature added on master by somebody else (via a PR)

8) `git pull --rebase upstream master`

9) `git push origin fastci` (may need to add `--force`)

10) PR accepted, after “ready to go” checked, tests, LGTM approval

11) `git checkout master`

12) `git pull --rebase upstream master`

`git push origin master`

12) `git branch -d fastci`

`git push origin :fastci`
ADVANTAGES OF THE NEW WORKFLOW

• For feature developers:-
  • Feedback from the Psi4 team during development.
  • No fear of breaking the code for others.
  • Easy to see whether each commit breaks anything.
  • Intuitive for those familiar with other GitHub projects.

• For the Psi4 team:-
  • Easy way to track what people are working on.
  • Easy vetting / testing / merging of new contributions.
  • Code review means more consistent style.

• For end users:-
  • Code from GitHub should always work.
PART II: CONDA & BRANCHES
LORI A. BURNS
• API docs like `help(psi4)`
• NumPy docstrings in driver
• Sphinx (and Perl) only build dep
DOCUMENTATION CHALLENGE
A PLOY TO GET RID OF STALE FORKS

KEEP CALM AND BUY A DEVELOPER A BEER

WHAT TO EDIT
HOW TO EDIT
DOCUMENTATION CHALLENGE
A PLOY TO GET RID OF STALE FORKS


1. Delete any existing GH fork

WHAT TO EDIT
HOW TO EDIT
1. Delete any existing GH fork

2. Create a new fork, perhaps by editing a file online (no clone) from docs/sphinxman/source/
DOCUMENTATION CHALLENGE
A PLOY TO GET RID OF STALE FORKS


HOW TO EDIT

1. Delete any existing GH fork

2. Create a new fork, perhaps by editing a file online (no clone) from docs/sphinxman/source/

3. Submit a pull request from a branch of your fork

WHAT TO EDIT
DOCUMENTATION CHALLENGE

A PLOY TO GET RID OF STALE FORKS

1. Delete any existing GH fork
2. Create a new fork, perhaps by editing a file online (no clone) from docs/sphinxman/source/
3. Submit a pull request from a branch of your fork
4. If conflicts arise, clone and rebase to resolve them, independently or with advice

WHAT TO EDIT

HOW TO EDIT
DOCUMENTATION CHALLENGE

A PLOY TO GET RID OF STALE FORKS


1. Delete any existing GH fork
2. Create a new fork, perhaps by editing a file online (no clone) from docs/sphinxman/source/
3. Submit a pull request from a branch of your fork
4. If conflicts arise, clone and rebase to resolve them, independently or with advice
5. Once PR accepted, collect your sticker sheet

WHAT TO EDIT

HOW TO EDIT
A PLOY TO GET RID OF STALE FORKS

1. Delete any existing GH fork
2. Create a new fork, perhaps by editing a file online (no clone) from docs/sphinxman/source/
3. Submit a pull request from a branch of your fork
4. If conflicts arise, clone and rebase to resolve them, independently or with advice
5. Once PR accepted, collect your sticker sheet

WHAT TO EDIT

• Anything that has confused you in narrative docs
• Any broken link
• Any docstring in driver
• Boost used 1-indexing for args, while Pybind11 uses 0-indexing, so many docstrings need adjusting in psi4/src/core.cc

HOW TO EDIT
<table>
<thead>
<tr>
<th>2015</th>
<th>binary</th>
<th>docs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tool</td>
<td>Linux</td>
<td>conda-build</td>
</tr>
<tr>
<td>Resource</td>
<td>Linux</td>
<td>psinet</td>
</tr>
<tr>
<td>Matrix</td>
<td>Linux</td>
<td>1 generic comp/gnu, py 2</td>
</tr>
<tr>
<td>When</td>
<td>nightly</td>
<td>nightly</td>
</tr>
<tr>
<td>Runs</td>
<td>full tests</td>
<td>full code scan</td>
</tr>
</tbody>
</table>
PSINET (AKA. SPINET)

MACPSINET
<table>
<thead>
<tr>
<th>2016</th>
<th>CI</th>
<th>binary</th>
<th>installer</th>
<th>stats</th>
<th>docs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tool</td>
<td>L</td>
<td>Travis</td>
<td>conda-build</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>M</td>
<td>Distelli</td>
<td></td>
<td>conda constructor</td>
<td></td>
</tr>
<tr>
<td></td>
<td>L</td>
<td>cloud</td>
<td>psinet</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>M</td>
<td>mac-psinet</td>
<td>mac-psinet, NYI</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>L</td>
<td>5 comp/gnu, py 2</td>
<td>1 generic comp/gnu, py 2</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>M</td>
<td>1 comp/gnu, py 2</td>
<td></td>
<td>1 comp/gnu, py 2</td>
<td></td>
</tr>
<tr>
<td>Resource</td>
<td>L</td>
<td>commit</td>
<td>nightly</td>
<td>occasionally</td>
<td>occasionally</td>
</tr>
<tr>
<td></td>
<td>M</td>
<td>every latest commit</td>
<td>crontab at 2AM</td>
<td>manually at new tag</td>
<td>manually</td>
</tr>
<tr>
<td>Matrix</td>
<td>L</td>
<td>quicktests</td>
<td>full tests</td>
<td>full tests</td>
<td>full + processing</td>
</tr>
<tr>
<td></td>
<td>M</td>
<td>AM 4</td>
<td>AM 6; all must pass</td>
<td>AM 6; all must pass</td>
<td>AM 5</td>
</tr>
<tr>
<td>When</td>
<td>runs</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Produces</td>
<td>Pass/Fail notice on GitHub commit</td>
<td>Psi4 core conda package</td>
<td>Installer: Miniconda + P4core + Addons</td>
<td>Coverage Dashboard, Sunburst</td>
<td>Sphinx &amp; Doxygen Autodocumenting</td>
</tr>
<tr>
<td>POST-PROCESSING</td>
<td>QC METHODS</td>
<td>INTEGRALS</td>
<td>SOLVATION</td>
<td>STRUCTURE &amp; VIS</td>
<td></td>
</tr>
<tr>
<td>----------------</td>
<td>------------</td>
<td>-----------</td>
<td>------------</td>
<td>----------------</td>
<td></td>
</tr>
<tr>
<td>DFTD3 <strong>Grimme</strong></td>
<td><strong>Ambit</strong> Turney</td>
<td><strong>ERD</strong> Flocke</td>
<td><strong>ADF</strong> Visscher</td>
<td><strong>MOLDEN</strong> Schaftenaar</td>
<td></td>
</tr>
<tr>
<td>dispersion correction</td>
<td>tensor manipulations</td>
<td>2e⁻ integrals</td>
<td>FDE solvation</td>
<td>orbital/density visualization</td>
<td></td>
</tr>
<tr>
<td><em>Fortran executable</em></td>
<td><em>Python/C++ library</em></td>
<td><em>Fortran library</em></td>
<td><em>Fortran/C executable</em></td>
<td><em>Fortran executable</em></td>
<td></td>
</tr>
<tr>
<td><strong>GCP</strong> <strong>Grimme</strong></td>
<td><strong>Cfour</strong> Stanton &amp; Gauss</td>
<td><strong>libint</strong> Valeev</td>
<td><strong>libefp</strong> Kaliman &amp; Slipchenko</td>
<td><strong>MOLDEN</strong> Schaftenaar</td>
<td></td>
</tr>
<tr>
<td>small-basis correction</td>
<td>CC/MBPT/properties/etc.</td>
<td>2e⁻ integrals</td>
<td>fragment potentials</td>
<td>orbital/density visualization</td>
<td></td>
</tr>
<tr>
<td><em>Fortran executable</em></td>
<td><em>Fortran executable</em></td>
<td><em>C library</em></td>
<td><em>C library</em></td>
<td><em>Fortran executable</em></td>
<td></td>
</tr>
<tr>
<td><strong>GDMA</strong> <strong>Stone</strong></td>
<td><strong>CheMPS2</strong> Wouters</td>
<td><strong>DKH</strong> Wolf &amp; Reiher</td>
<td><strong>PCMSolver</strong> DiRemigio</td>
<td><strong>MOLDEN</strong> Schaftenaar</td>
<td></td>
</tr>
<tr>
<td>multipole analysis</td>
<td>DMRG theory</td>
<td>relativistic correction</td>
<td>solvation</td>
<td>orbital/density visualization</td>
<td></td>
</tr>
<tr>
<td><em>Fortran library</em></td>
<td><em>Python/C++ library</em></td>
<td><em>Fortran library</em></td>
<td><em>Fortran library</em></td>
<td><em>Fortran executable</em></td>
<td></td>
</tr>
<tr>
<td><strong>JANPA</strong> Nikolaienko</td>
<td><strong>Gaussian</strong></td>
<td><strong>resp2</strong> McGibbon</td>
<td><strong>v2rdm_casscf</strong> DePrince</td>
<td><strong>MOLDEN</strong> Schaftenaar</td>
<td></td>
</tr>
<tr>
<td>NBO bond analysis</td>
<td>many-body expansion</td>
<td>RESP charges</td>
<td>active-space SCF</td>
<td>orbital/density visualization</td>
<td></td>
</tr>
<tr>
<td><em>Java executable</em></td>
<td><em>Fortran executable</em></td>
<td><em>C++ Psi4 plugin</em></td>
<td><em>Fortran/C++ Psi4 plugin</em></td>
<td><em>Fortran executable</em></td>
<td></td>
</tr>
<tr>
<td><strong>Pulsar</strong> Pritchard &amp; Richards</td>
<td><strong>MRCC</strong> Kállay</td>
<td><strong>v2rdm_casscf</strong> DePrince</td>
<td><strong>libxc</strong> Marques</td>
<td><strong>MOLDEN</strong> Schaftenaar</td>
<td></td>
</tr>
<tr>
<td>datastructure interoperability</td>
<td>arbitrary order CC/CI</td>
<td>exchange-correlation fnctls.</td>
<td><em>exchange-correlation fnctls.</em></td>
<td><em>orbital/density visualization</em></td>
<td></td>
</tr>
<tr>
<td><em>C++ framework; calls Psi4</em></td>
<td><em>Fortran executable</em></td>
<td><em>C library</em></td>
<td><em>C library</em></td>
<td><em>Fortran executable</em></td>
<td></td>
</tr>
<tr>
<td><strong>WebMO</strong> Polik &amp; Schmidt</td>
<td><strong>WebMO</strong> Polik &amp; Schmidt</td>
<td><strong>MRCC</strong> Kállay</td>
<td><strong>v2rdm_casscf</strong> DePrince</td>
<td><strong>MOLDEN</strong> Schaftenaar</td>
<td></td>
</tr>
<tr>
<td>GUI/web server</td>
<td>WebMO Polik &amp; Schmidt</td>
<td>MRCC Kállay</td>
<td><em>active-space SCF</em></td>
<td>orbital/density visualization</td>
<td></td>
</tr>
<tr>
<td><em>Perl/Java executable; calls Psi4</em></td>
<td><em>MRCC executable</em></td>
<td><em>Fortran/C++ Psi4 plugin</em></td>
<td><em>Fortran/C++ Psi4 plugin</em></td>
<td><em>orbital/density visualization</em></td>
<td></td>
</tr>
<tr>
<td><strong>UPSTREAM</strong></td>
<td><strong>UPSTREAM</strong></td>
<td><strong>UPSTREAM</strong></td>
<td><strong>UPSTREAM</strong></td>
<td><strong>UPSTREAM</strong></td>
<td></td>
</tr>
<tr>
<td><strong>CHARMMing</strong> Woodcock</td>
<td><strong>CHARMMing</strong> Woodcock</td>
<td><strong>CHARMMing</strong> Woodcock</td>
<td><strong>CHARMMing</strong> Woodcock</td>
<td><strong>CHARMMing</strong> Woodcock</td>
<td></td>
</tr>
<tr>
<td>QM/MM web server</td>
<td>QM/MM web server</td>
<td>QM/MM web server</td>
<td>QM/MM web server</td>
<td>QM/MM web server</td>
<td></td>
</tr>
<tr>
<td><em>Python/HTML executable; calls Psi4</em></td>
<td><em>Python/HTML executable; calls Psi4</em></td>
<td><em>Python/HTML executable; calls Psi4</em></td>
<td><em>Python/HTML executable; calls Psi4</em></td>
<td><em>Python/HTML executable; calls Psi4</em></td>
<td></td>
</tr>
<tr>
<td><strong>Pulsar</strong> Pritchard &amp; Richards</td>
<td><strong>Pulsar</strong> Pritchard &amp; Richards</td>
<td><strong>Pulsar</strong> Pritchard &amp; Richards</td>
<td><strong>Pulsar</strong> Pritchard &amp; Richards</td>
<td><strong>Pulsar</strong> Pritchard &amp; Richards</td>
<td></td>
</tr>
<tr>
<td>datastructure interoperability</td>
<td>datastructure interoperability</td>
<td>datastructure interoperability</td>
<td>datastructure interoperability</td>
<td>datastructure interoperability</td>
<td></td>
</tr>
<tr>
<td><em>C++ framework; calls Psi4</em></td>
<td><em>C++ framework; calls Psi4</em></td>
<td><em>C++ framework; calls Psi4</em></td>
<td><em>C++ framework; calls Psi4</em></td>
<td><em>C++ framework; calls Psi4</em></td>
<td></td>
</tr>
<tr>
<td><strong>WebMO</strong> Polik &amp; Schmidt</td>
<td><strong>WebMO</strong> Polik &amp; Schmidt</td>
<td><strong>WebMO</strong> Polik &amp; Schmidt</td>
<td><strong>WebMO</strong> Polik &amp; Schmidt</td>
<td><strong>WebMO</strong> Polik &amp; Schmidt</td>
<td></td>
</tr>
<tr>
<td>GUI/web server</td>
<td>GUI/web server</td>
<td>GUI/web server</td>
<td>GUI/web server</td>
<td>GUI/web server</td>
<td></td>
</tr>
<tr>
<td><em>Perl/Java executable; calls Psi4</em></td>
<td><em>Perl/Java executable; calls Psi4</em></td>
<td><em>Perl/Java executable; calls Psi4</em></td>
<td><em>Perl/Java executable; calls Psi4</em></td>
<td><em>Perl/Java executable; calls Psi4</em></td>
<td></td>
</tr>
<tr>
<td><strong>INTEGRALS</strong></td>
<td><strong>INTEGRALS</strong></td>
<td><strong>INTERNET</strong></td>
<td><strong>INTERNET</strong></td>
<td><strong>INTERNET</strong></td>
<td></td>
</tr>
<tr>
<td><strong>ERD</strong> Flocke</td>
<td><strong>libint</strong> Valeev</td>
<td><strong>PCMSolver</strong> DiRemigio</td>
<td><strong>MOLDEN</strong> Schaftenaar</td>
<td><strong>MOLDEN</strong> Schaftenaar</td>
<td></td>
</tr>
<tr>
<td><strong>libint</strong> Valeev</td>
<td><strong>simint</strong> Pritchard</td>
<td><strong>MOLDEN</strong> Schaftenaar</td>
<td><strong>MOLDEN</strong> Schaftenaar</td>
<td><strong>MOLDEN</strong> Schaftenaar</td>
<td></td>
</tr>
<tr>
<td><strong>simint</strong> Pritchard</td>
<td><strong>MOLDEN</strong> Schaftenaar</td>
<td><strong>MOLDEN</strong> Schaftenaar</td>
<td><strong>MOLDEN</strong> Schaftenaar</td>
<td><strong>MOLDEN</strong> Schaftenaar</td>
<td></td>
</tr>
<tr>
<td><strong>INTEGRALS</strong></td>
<td><strong>QC METHODS</strong></td>
<td><strong>POST-PROCESSING</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------------</td>
<td>----------------</td>
<td>---------------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ERD Flocke</td>
<td>Ambit Turney</td>
<td>DFTD3 Grimme</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2e^- integrals</td>
<td>tensor manipulations</td>
<td>dispersion correction</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fortran library</td>
<td>Python/C++ library</td>
<td>Fortran executable</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>libint Valeev</td>
<td>Cfour Stanton &amp; Gauss</td>
<td>GCP Grimme</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2e^- integrals</td>
<td>CC/MBPT/properties/etc.</td>
<td>small-basis correction</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C library</td>
<td>Fortran executable</td>
<td>Fortran executable</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>simint Pritchard</td>
<td>CheMPS2 Wouters</td>
<td>GDMA Stone</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2e^- integrals</td>
<td>DMRG theory</td>
<td>multipole analysis</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C++ library</td>
<td>Python/C++ library</td>
<td>Fortran library</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Gaussian</td>
<td>DKH Wolf &amp; Reiher</td>
<td>JANPA Nikolaienko</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>many-body expansion</td>
<td>relativistic correction</td>
<td>NBO bond analysis</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fortran executable</td>
<td>Fortran library</td>
<td>Java executable</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MRCC Kállay</td>
<td>v2rdm_casscf DePrince</td>
<td>resp2 McGibbon</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>arbitrary order CC/Cl</td>
<td>active-space SCF</td>
<td>RESP charges</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fortran executable</td>
<td>Fortran executable</td>
<td>C++ Psi4 plugin</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>v2rdm_casscf DePrince</td>
<td>Gaussian</td>
<td>resp2 McGibbon</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>active-space SCF</td>
<td>many-body expansion</td>
<td>RESP charges</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fortran executable</td>
<td>Fortran executable</td>
<td>C++ Psi4 plugin</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>libxc Marques</td>
<td>libxc Marques</td>
<td>WebMO Polik &amp; Schmidt</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>exchange-correlation fnctls.</td>
<td>exchange-correlation fnctls.</td>
<td>GUI/web server</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C library</td>
<td>C library</td>
<td>Perl/Java executable; calls Psi4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Psi4 actively performs build and integration testing</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
**POST-PROCESSING**

- **DFTD3** Grimme
  - dispersion correction
  - Fortran executable

- **GCP** Grimme
  - small-basis correction
  - Fortran executable

- **GDMA** Stone
  - multipole analysis
  - Fortran library

- **JANPA** Nikolaenko
  - NBO bond analysis
  - Java executable

- **resp2** McGibbon
  - RESP charges
  - C++ Psi4 plugin

**UPSTREAM**

- **CHARMMing** Woodcock
  - QM/MM web server
  - Python/HTML executable; calls Psi4

- **Pulsar** Pritchard & Richards
  - datastructure interoperability
  - C++ framework; calls Psi4

- **WebMO** Polik & Schmidt
  - GUI/web server
  - Perl/Java executable; calls Psi4

**QC METHODS**

- **Ambit** Turney
  - tensor manipulations
  - Python/C++ library

- **Cfour** Stanton & Gauss
  - (CC|CC)
  - CC/MBPT/properties/etc.
  - Fortran executable

- **CheMPS2** Wouters
  - DMRG theory
  - Python/C++ library

- **DKH** Wolf & Reiher
  - relativistic correction
  - Fortran executable

- **Gaussian**
  - many-body expansion
  - Fortran executable

- **MRCC** Kállay
  - arbitrary order CC/CI
  - Fortran executable

- **v2rdm_casscf** DePrince
  - active-space SCF
  - Fortran/C++ Psi4 plugin

- **libxc** Marques
  - exchange-correlation fnctls.
  - C library

**INTEGRALS**

- **ERD** Flocke
  - 2e⁻ integrals
  - Fortran library

- **libint** Valeev
  - 2e⁻ integrals
  - C library

- **simint** Pritchard
  - 2e⁻ integrals
  - C++ library

**SOLVATION**

- **ADF** Visscher
  - FDE solvation
  - Fortran/C executable

- **libefp** Kaliman & Slipchenko
  - fragment potentials
  - C library

- **PCMSolver** DiRemigio
  - solvation
  - Fortran library

**STRUCTURE & VIS**

- **MOLDEN** Schaftenaar
  - orbital/density visualization
  - Fortran executable

- **PubChem** NIH
  - structure look-up
  - HTML

**Psi4 actively performs build and integration testing**
# INTEGRATIONS FOR ALL

## CORE DEVELOPERS

```bash
>>> # dir with only Psi4 source
>>> # all non-required add-ons disabled by default
>>> cmake

-DCMAKE_PREFIX_PATH="/path1;/another/path2"
-DENABLE_CheMPS2
-DENABLE_gdma

... # CMake downloads external projects’ source from internet
>>> make

>>> ctest -L addons
Psi4 core.........................................................PASSED
Psi4 + DFTD3 executable...........................................PASSED
Psi4 + CheMPS2 library............................................PASSED
Psi4 + PCMSolver library..........................................PASSED
Psi4 + V2RDM_CASSCF plugin........................................PASSED
Psi4 + Ambit library..............................................PASSED
Psi4 library......................................................PASSED

# working Psi4 executable complete with add-ons, customized for local hardware & software
```
INTEGRATIONS FOR ALL

seeking *built* in **CMAKE_PREFIX_PATH**

**CORE DEVELOPERS**

```bash
>>> # dir with only Psi4 source
>>> # all non-required add-ons *disabled* by default
>>> cmake
    -DCMAKE_PREFIX_PATH="/path1;/another/path2"
    -DENABLE_CheMPS2
    -DENABLE_gdma
    ...
    # CMake downloads external projects’ source from internet
>>> make
>>> ctest -L addons
Psi4 core ............................................................... PASSED
Psi4 + DFTD3 executable ............................................... PASSED
Psi4 + CheMPS2 library ................................................ PASSED
Psi4 + PCMSolver library ............................................... PASSED
Psi4 + V2RDM_CASSCF plugin ........................................... PASSED
Psi4 + Ambit library .................................................... PASSED
Psi4 library ............................................................... PASSED
>>> # working Psi4 executable complete with add-ons, customized for local hardware & software
```
```bash
>>> # dir with only Psi4 source
>>> # all non-required add-ons disabled by default
>>> cmake
   -DCMAKE_PREFIX_PATH="/path1;/another/path2"
   -DENABLE_CheMPS2
   -DENABLE_gdma
   ...
   # CMake downloads external projects' source from internet
>>> make
>>> ctest -L addons
 Psi4 core........................................................................PASSED
 Psi4 + DFTD3 executable...............................................PASSED
 Psi4 + CheMPS2 library................................................PASSED
 Psi4 + PCMSolver library...........................................PASSED
 Psi4 + V2RDM_CASSCF plugin....................................PASSED
 Psi4 + Ambit library................................................PASSED
 Psi4 library.........................................................PASSED
>>> # working Psi4 executable complete with add-ons, customized for local hardware & software
```
# dir with only Psi4 source
>>> # all non-required add-ons disabled by default
>>> cmake \
   -DCMAKE_PREFIX_PATH="/path1;/another/path2" \
   -DENABLE_CheMPS2 \
   -DENABLE_gdma \
   ...
   # CMake downloads external projects’ source from internet
>>> make
>>> ctest -L addons
Psi4 core..........................................................PASSED
Psi4 + DFTD3 executable...........................................PASSED
Psi4 + CheMPS2 library.............................................PASSED
Psi4 + PCMSolver library.........................................PASSED
Psi4 + V2RDM_CASSCF plugin....................................PASSED
Psi4 + Ambit library..............................................PASSED
Psi4 library..........................................................PASSED
>>> # working Psi4 executable complete with add-ons, customized for local hardware & software
cmake -DCMAKE_INSTALL_PREFIX="/path/to/install" -DENABLE_CheMPS2 -DENABLE_gdma -DENABLE_Linux64 -DENABLE_MacOSX -DENABLE_Win64
make
ctest -L addons
Psi4 core..........................................................PASSED
Psi4 + DFTD3 executable...........................................PASSED
Psi4 + CheMPS2 library.............................................PASSED
Psi4 + PCMSolver library.........................................PASSED
Psi4 + V2RDM_CASSCF plugin....................................PASSED
Psi4 + Ambit library..............................................PASSED
Psi4 library..........................................................PASSED
>>> ctest -L addons NYI
nothing so far has saved compile time, unless you built all those pkgs separately (and compatibly)
• you’re probably not using many of the addons, so why bother maintaining their build
• conda’s already got them built. why not reuse?
INTEGRATIONS FOR ALL

```bash
>>> # install a mini/anaconda
>>> # update it (conda update conda)
>>> # if “conda env create” below fails, conda install anaconda-client –or–
>>> # download file from site and save as environment.yml
>>> cat environment.yml
name: linux_psi4_devel
channels: !!python/tuple
  - !!python/unicode 'defaults'
dependencies:
- mkl=11.3.3=0
- numpy=1.11.2=py27_0
- psi4/label/test::gdma=2.2.06=0
- psi4/label/test::psi4=1.1a2.dev3+20605d7=py27_0
- psi4::dftd3=3.2.0=7
- psi4::gcc-5=5.2.0=1
- psi4::libint=1.1.6=0
- python=2.7.12=1
...
>>> conda env create psi4/linux_psi4_devel (or macos_psi4_devel)
>>> source activate linux_psi4_devel
>>> # working Psi4 executable complete with add-ons
```
# dir with only Psi4 source

```bash
export PREFIX=/home/miniconda/envs/linux_psi4_devel

cmake -DCMAKE_PREFIX_PATH="${PREFIX}" -DENABLE_CheMPS2=ON -DENABLE_gdma=ON -DMAX_AM_ERI=6 -DPYTHON_EXECUTABLE=${PREFIX}/bin/python2.7...
```

# CMake collects mostly built conda libraries, internet src as fallback

```
make
ctest -L addons
Psi4 core.........................................................PASSED
Psi4 + DFTD3 executable...........................................PASSED
Psi4 + CheMPS2 library............................................PASSED
Psi4 + PCMSolver library..........................................PASSED
Psi4 + V2RDM_CASSCF plugin...................................PASSED
Psi4 + Ambit library.............................................PASSED
Psi4 library......................................................PASSED
```

# working Psi4 executable complete with add-ons, Psi4 core customized for local hardware & software
# dir with only Psi4 source

export PREFIX=/home/miniconda/envs/linux_psi4_devel

cmake

-DCMAKE_PREFIX_PATH="${PREFIX}" 
-DENABLE_CheMPS2=ON 
-DENABLE_gdma=ON 
-DMAX_AM_ERI=6 

-DPYTHON_EXECUTABLE="${PREFIX}/bin/python2.7"

... 

CMake collects mostly built conda libraries, internet src as fallback

make

Psi4 core.........................................................PASSED
Psi4 + DFTD3 executable...........................................PASSED
Psi4 + CheMPS2 library............................................PASSED
Psi4 + PCMSolver library..........................................PASSED
Psi4 + V2RDM_CASSCF plugin........................................PASSED
Psi4 + Ambit library..............................................PASSED
Psi4 library......................................................PASSED

# working Psi4 executable complete with add-ons, Psi4 core customized

for local hardware & software
INTEGRATIONS FOR ALL

USERS

```bash
>>> # download Psi4 installer
>>> bash Psi4conda-latest.sh
...
Psi4 core.........................................................PASSED
Psi4 + DFTD3 executable...........................................PASSED
Psi4 + CheMPS2 library............................................PASSED
Psi4 + PCMSolver library..........................................PASSED
Psi4 + V2RDM_CASSCF plugin........................................PASSED
Psi4 + Ambit library..............................................PASSED
Psi4 library......................................................PASSED
>>> # working Psi4 executable complete with add-ons
```

PERIPHERAL DEVELOPERS

```bash
>>> # working Psi4 executable complete with add-ons (from above)
>>> conda install gcc
>>> psi4 --new-plugin mygreatcode
>>> cd mygreatcode
>>> make
Attention! This SCF may be density-fitted.
>>> # working compiling development environment ready for extension
```
VERSIONING
PRAY I DON’T ALTER IT ANY FURTHER

- DEVELOPMENT SNAPSHOT
  - on master; passes quicktests, may break features;
  - FORMAT M.m+1.devN
  - UPDATE VIA conda update psi4 -c psi4/label/level

- PRE-RELEASE
  - on master; passes quicktests; conda devel channel
  - ALPHA much of the milestone is in
  - RC much a plausible release candidate
  - conda devel channel

- RELEASE
  - on master or maintenance; all tests pass, never less capable than its predecessor
  - FORMAT master (M.m) or maintenance (M.m.p)
  - UPDATE VIA conda update psi4

- want to be able to break things on psi4/psi4 master occasionally (e.g., plugins, mints9) without those commits being a released version (implied by 1.0.101)
- want to be able to do bug fixes on releases (pulled off as 1.0.x) without naming conflicts
- want to accommodate those who only want stable versions
- you can ignore tags if you like. but locally computed v will be wrong
equivalent

1.0.0

1.0

RELEASE

A

B

C

D

E

F

G

H

PRE-RELEASE

I

PRE-RELEASE

J

K

L

RELEASE

M

Development master not readily accessed

Maintenance 1.0.x

• always has a tag at its head
• only bug fixes

1.0.0

1.0

1.0.1

1.0.2

1.1.dev2 1.1.dev3

1.1.dev5

1.1a1.dev1

1.1rc1.dev2

1.1a1

1.1rc1

1.1.dev3

1.1.dev2

1.1.dev5

1.1a1.dev1

1.1rc1.dev2

1.2.dev1

Maintenance 1.1.x
PART III: PLUGINS & LAYOUT

JUSTIN M. TURNNEY
Plugins continue to be the best way to add new functionality to Psi4.

Your code is developed as a standalone entity, which is compiled independently of Psi4, but can still make sure of Psi4’s vast library.

Plugins are loaded at runtime by Psi4 from any location.
PLUGINS

% psi4 --help
   --new-plugin NEW_PLUGIN
       Creates a new directory with files for writing a new
       plugin.
   --new-plugin-template
       {aointegrals,basic,dfmp2,mointegrals,scf,sointegrals,wavefunction}
       New plugin template to use.

% psi4 --new-plugin mydfmp2 --new-plugin-template dfmp2
   -- Creating "mydfmp2" with "dfmp2" template. -------------------
   ==> Created plugin files:
       __init__.py, CMakeLists.txt, doc.rst, input.dat, plugin.cc, pymodule.py

The ability to create just a Makefile for an existing plugin as been removed.
Plugins now make use of CMake for configuring and building.

If you install Psi4 into a non-standard location you will likely need to define `CMAKE_PREFIX_PATH`
% cd mydfmp2
% cmake .
-- The CXX compiler identification is AppleClang 8.0.0.8000042
-- Check for working CXX compiler: /usr/bin/c++
-- Check for working CXX compiler: /usr/bin/c++ -- works
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info -- done
-- Detecting CXX compile features
-- Detecting CXX compile features -- done
-- Psi4 include directory: /Users/jtturney/Install/psi4/include
-- Psi4 executable: /Users/jtturney/Install/psi4/bin/psi4
-- Psi4 core library: /Users/jtturney/Install/psi4/lib/psi4/core.so
-- Python executable: /Users/jtturney/anaconda3/bin/python
-- Configuring done
-- Generating done
-- Build files have been written to: /Users/jtturney/mydfmp2

% make

% psi4
PART IV: DIRECTORY LAYOUT

JUSTIN M. TURNEY
**BINARIES AND LIBRARIES Merged**

- **WHAT** modules/libraries of C++ code, written by us
- **WHY** bin/lib distinguished by names; mirror installed header paths
- **NEW** binaries get built; libraries get their headers installed
### C++ CORE
RAISED ABOVE MODULES AND CLEANED

**src/bin/psi4/**
- clean.cc
- export_blas_lapack.cc
- export_functional.cc
- export_mints.cc
- export_oeprop.cc
- ...
- psi4.cc
- psi4.h
- psi_start.cc
- psi_stop.cc
- python.cc
- read_options.cc
- script.cc
- script.h
- set_memory.cc

**psi4/src/**
- core.cc
- export_blas_lapack.cc
- export_functional.cc
- export_mints.cc
- export_oeprop.cc
- ...
- psi4/
- read_options.cc

---

**C. 2015**
- **WHAT** module-running code
- **WHY** sanity and CMake like dependencies as subdirectories
- **NEW** `python.cc` (`BOOST_PYTHON_MODULE(psi4)`) becomes `core.cc` (`PYBIND11_PLUGIN(core)`); most else gone

**NOW**
PSIDATADIR
SPLIT BY LANGUAGE, PURPOSE, & PROJECT

- **WHAT** the pieces that didn’t need compilation
- **WHY** separate python and data; mirror GNU install directory structure in source
- **NEW** `python/` becomes `driver/`; envvar PSI4DATADIR defunct; efp evicted
HEADERS
WINNOWED AND NAMESPACE

- **include/**
  - Z_to_element.h
  - chkpt_params.h
  - compiler.h
  - cov_radii.h
  - element_to_Z.h
  - exception.h
  - fragment.h
  - masses.h
  - molecular_system.h
  - physconst.h
  - process.h
  - psi4-dec.h
  - psi4-def.h
  - psifiles.h
  - psitypes.h
  - ...

- **psi4/include/psi4/**
  - masses.h
  - physconst.h
  - pragma.h
  - psi4-dec.h
  - psifiles.h
  - pybind11.h

- **psi4/src/psi4/libmints/**
  - element_to_Z.h

- **psi4/src/psi4/optking/**
  - cov_radii.h

- **psi4/src/psi4/libpsi4util/**
  - exception.h

- **psi4/src/psi4/libparallel/**
  - process.h

**C. 2015**

- **WHAT** global (supra-module) headers
- **WHY** mirror GNU install directory structure in source
- **NEW** now actually installed! usable by plugins
EXTERNAL PROJECTS
TENTACLES SEVERED

WHAT everything we didn’t write that we depend on or use as enhancement

WHY good practice, sanity for maintainers (LAB), and avoid possessing others’ code

NEW no external code stored locally, even tarballs & share; “config” CMake proj., not “module”

boost/
boost_1_57_0.tar.bz2
cmake/
FindCHEMPS2.cmake
FindPCMSolver.cmake
ConfigPCMSolver.cmake
ConfigChemps2.cmake
ConfigBoost.cmake
interfaces/
libefp/
lib/
efpfrag/
src/lib/
libderiv/
liberd/
libint/
libmints/dkh2-dkh4_main.F90

external/
ambit/CMakeLists.txt
chemps2/CMakeLists.txt
dkh/CMakeLists.txt
gdma/CMakeLists.txt
gtfock/CMakeLists.txt
libefp/CMakeLists.txt
liberd/CMakeLists.txt
libint/CMakeLists.txt
pcmsolver/CMakeLists.txt
pybind11/CMakeLists.txt
• **WHAT** initial run-time probes

• **WHY** everyone from Py, CM, GNU has versioning prefs; easier to control env through Py

• **NEW** wait for versioning lightning talk
DIRECTORY STRUCTURE

**psi4/**
- src/
  - core.cc (make py-mod, top exports)
  - export_mints.cc (module exports)
  - export_blas_lapack.cc ("")
  - export_functional.cc ("")
  - export_oeprop.cc ("")
- ...
- psi4/ (module source)

**driver/**
- import_core.py (import C- details)

**include/**
- (C-side headers)
- psi4-config.in (build config help)
- psi4Config.cmake.in (CM detection)
- run_psi4.py.in (simulate psi4 exe)

**share/**
- (text data library)

**src/**
- version.py (header & versioning)

**psi4+ext/**
- superbuild
- LICENSE (GPLv2+)
- cmake/ (superbuild build support)
- doc/ (documentation)
- external/ (external projects)
- media/ (logos)
- plugins/ (model plugins)
- psi4/ (Psi4 proj. in superbuild)
- samples/ (tests less CMake)
- tests/ (Psi4 core & add-on tests)

**CONCERNING C-SIDE CORE**

**psi4/src/**
- core.cc (make py-mod, top exports)
- export_mints.cc (module exports)
- export_blas_lapack.cc ("")
- export_functional.cc ("")
- export_oeprop.cc ("")
- ...

**CONCERNING PSI4 WHOLE**

** psi4/ **
- __init__.py (import Py- & C-sides)
- driver/ (Py-side Psi4)
- import_core.py (import C- details)
- include/ (C-side headers)
- psi4-config.in (build config help)
- psi4Config.cmake.in (CM detection)
- run_psi4.py.in (simulate psi4 exe)

**CONCERNING PSI4+EXT**

**SUPERBUILD**

DIRECTORY STRUCTURE

**psi4/**
- src/
  - core.cc (make py-mod, top exports)
  - export_mints.cc (module exports)
  - export_blas_lapack.cc ("")
  - export_functional.cc ("")
  - export_oeprop.cc ("")
- ...
- psi4/ (module source)

**driver/**
- import_core.py (import C- details)

**include/**
- (C-side headers)
- psi4-config.in (build config help)
- psi4Config.cmake.in (CM detection)
- run_psi4.py.in (simulate psi4 exe)

**share/**
- (text data library)

**src/**
- version.py (header & versioning)

**psi4+ext/**
- superbuild
- LICENSE (GPLv2+)
- cmake/ (superbuild build support)
- doc/ (documentation)
- external/ (external projects)
- media/ (logos)
- plugins/ (model plugins)
- psi4/ (Psi4 proj. in superbuild)
- samples/ (tests less CMake)
- tests/ (Psi4 core & add-on tests)
DIRECTORY STRUCTURE
NOT MEANT TO BE CRUEL

- constrained by GitHub repository name
- contains the CMake project
- constrained by Python module name
- contains the Python project
- constrained by GNU project name
- contains the C++ project
- new namespace pad dirs, but tab-able
PART V: A BRIEF HISTORY OF VERSIONING Psi4

OR, WHY THINGS KEEP GETTING MORE COMPLICATED

LORI A. BURNS
IN THE BEGINNING
JET NEEDED NO VERSIONING

- CONSTRAINTS None: only one person. Plus, SVN numbers sequentially.
THE INTERREGNUM
TRUE DEVELOPERS ONLY NEED GIT HASHES

• CONSTRAINTS Minimal: only need to ID the code so git hash suffices.
**VERSIONING CIRCA 2015**
Enter Conda & Sortability

- **CONSTRAINTS** conda update psi4 needs a sortable version to act upon
• CONSTRAINTS `conda update psi4` needs a sortable version to act upon
VERSIONING IN THE AGE OF CHANGE
I HAVE ALTERED THE VERSIONING

- CONSTRAINTS
  - want to break things on master occasionally (e.g., plugins, cc14) without implying a released version
  - want to do bug fixes on tagged releases without naming conflicts
  - want to accommodate users who only want stable versions

THE FINE PRINT: you can ignore tags if you like. but locally computed version will be undefined
Pray I don’t alter it any further

**Development Snapshot**
- on master; passes quicktests, may break features;
- **format** `M.m+1.devN`
- **update via** `conda update psi4 -c psi4/label/level`

**Pre-Release**
- on master; passes quicktests; conda devel channel
- **alpha** much of the milestone is in
- **rc** much a plausible release candidate
- conda devel channel

**Release**
- on master or maintenance; all tests pass, never less capable than its predecessor
- **format** `master(M.m)` or `maintenance(M.m.p)`
- **update via** `conda update psi4`

---

**A** 1.1.dev2
**B**
**C**
Development master not readily accessed

Maintenance 1.0.x
- always has a tag at its head
- only bug fixes

Maintenance 1.1.x

• conda update psi4 –c psi4/label/devel

on master; passes quicktests, may break features; M.m+1.devN format; conda update psi4 –c psi4/label/devel

on master; passes quicktests; conda devel channel

on master (M.m) or maintenance (M.m.p) branch; all tests pass, always excepting deprecations, never less capable than its predecessor; conda update psi4
on master; passes quicktests, may break features; M.m+1.devN format; conda update psi4 -c psi4/label/devel

on master; passes quicktests; conda devel channel

on master (M.m) or maintenance (M.m.p) branch; all tests pass, always excepting deprecations, never less capable than its predecessor; conda update psi4

equivalent

1.0.0

RELEASE

v1.0

A

Development master not readily accessed

V DEVELOPTMENT SNAPSHOT

PRE-RELEASE

1.1.dev2 1.1.dev3

1.1.dev5

1.1a1.dev1

1.1rc1.dev2

RELEASE

v1.1

M

Maintenance 1.0.x

• always has a tag at its head
• only bug fixes

Maintenance 1.0.x

A

D' E' F'

H'

PRE-RELEASE

v1.1a1

v1.1rc1

1.1.a1.dev1

1.1rc1.dev2

RELEASE

v1.0.1

v1.0.2

PRE-RELEASE

v1.0.1

v1.0.2

PRE-RELEASE

v1.1a1

v1.1rc1

RELEASE

v1.1

L

M.0.1